Основи інформаційних технологій

Видавництво Літера

З М І С Т

ВСТУП 6

ЧАСТИНА І. ОСНОВИ ІНФОРМАТИКИ

Розділ 1. Загальні відомості про інформатику

1.1. Загальне поняття про інформатику 8
1.2. Сигнали 11
1.3. Перетворення інформації в універсальну форму двійкових чисел 16
Тестові завдання 22

Розділ 2. Комп'ютер — основний засіб оброблення інформації

2.1. Принципи роботи комп'ютера 23
2.2. Ієрархічні рівні комп'ютера 26
2.3. Архітектура фон Неймана 29
2.4. Організація взаємодії пристроїв комп'ютера під час виконання поточної команди 30
Тестові завдання 32

Розділ 3. Логічні й арифметичні основи комп'ютера

3.1. Логічні основи комп'ютерної техніки 33
3.2. Логічні сигнали 35
3.3. Логічні елементи 37
3.4. Поняття про числа й системи числення 39
3.5. Числа з фіксованою точкою і арифметичні дії з ними 44
3.6. Числа з рухомою точкою і арифметичні дії з ними 47
Тестові завдання 49

ЧАСТИНА ІІ. АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРІВ

Розділ 4. Пристрої введення і виведення інформації

4.1. Пристрої введення інформації 54
4.2. Пристрої виведення інформації 56
4.3. Інформаційні магістралі (шини) 64
Тестові завдання 66

Розділ 5. Процесори (мікропроцесори)

5.1. Історія розвитку процесорів 68
5.2. Класифікація процесорів 69
5.3. Будова процесорів 70
Тестові завдання 77

Розділ 6. Пам’ять комп’ютера

6.1. Загальні відомості 78
6.2. Основна (оперативна) пам’ять 81
6.3. Кеш-пам’ять 85
Тестові завдання 91

ЧАСТИНА III. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП’ЮТЕРІВ

Розділ 7. Програмне забезпечення. Основні поняття

7.1. Види програмного забезпечення 93
7.2. Алгоритми 95
7.3. Мови програмування 98
7.4. Операційні системи 100
Тестові завдання 101

Розділ 8. Мови програмування низького рівня

8.1. Мікрокоманди й мова міжрегістрових пересилань RTL 102
8.2. Машинні команди й архітектура системи команд 102
8.3. Режими адресації 104
8.4. Асемблер 106
Тестові завдання 107

Розділ 9. Мови програмування високого рівня

9.1. Трансляція програм 109
9.2. Види мов програмування 111
Тестові завдання 114

Розділ 10. Операційні системи комп’ютерів

10.1. Коротка історія розвитку операційних систем 115
10.2. Призначення операційних систем 117
10.3. Режими роботи комп’ютера 119
10.4. Структури операційних систем 119
10.5. Основні функції операційних систем. 121
Тестові завдання 135

ЧАСТИНА IV. НАЙПОШИРЕНІШІ КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

Розділ 11. Оброблення текстової інформації. Програма Microsoft Word

11.1. Текстова інформація 137
11.2. Засоби оброблення текстової інформації 138
11.3. Текстовий процесор Microsoft Word 140
Тестові завдання 161

Розділ 12. Оброблення графічної інформації. Програма Microsoft Visio

12.1. Основні поняття комп’ютерної графіки 162
12.2. Графічний редактор Microsoft Visio і його призначення 164
12.3. Графічні об’єкти в MS Visio 177

12.4. Створення тексту 185
Тестові завдання 194

Розділ 13. Оброблення інформації за допомогою електронних таблиць. Програма Microsoft Excel
13.1. Основні елементи електронних таблиць 195
13.2. Оброблення інформації в Excel 202
13.3. Подання інформації за допомогою діаграм і графіків. 209
Тестові завдання 214

Розділ 14. Бази даних і системи керування базами даних. Програма Microsoft Access
14.1. Основні поняття баз даних 215
14.2. Система Microsoft Access 217
14.3. Основні елементи СКБД MS Access 219
14.4. Таблиці й робота з ними 221
14.5. Створення зв'язків між таблицями 227
14.6. Застосування запитів для оброблення і подавання інформації . . 229
14.7. Застосування форм для роботи з базою даних 234
14.8. Створення звітів 237
Тестові завдання 239

Розділ 15. Створення презентацій. Програма Microsoft PowerPoint
15.1. Основні поняття презентації 240
15.2. Етапи створення презентації та вимоги до її оформлення 241
15.3. Основні елементи PowerPoint 242
15.4. Робота зі слайдами презентації 246
Тестові завдання 251

Розділ 16. Система оброблення математичної інформації MathCAD
16.1. Особливості оброблення математичної інформації 252
16.2. Загальні відомості про систему MathCAD 252
16.3. Засоби оброблення математичної інформації MathCAD 257
16.4. Створення графіків засобами MathCAD 267
16.5. Використання текстової інформації в MathCAD 268
Тестові завдання 269

Розділ 17. Комп'ютерні мережі. Інтернет
17.1. Комп'ютерні мережі 270
17.2. Глобальна комп'ютерна мережа Інтернет 277
17.3. Служби Інтернету 279
17.4. Хмарні комп'ютерні технології 283
17.5. Інтернет речей. 285
17.6. Штучний інтелект 287
Тестові завдання 288

Список використаної та рекомендованої літератури 288

ВСТУП

Сучасний стан суспільства характеризується різким зростанням ролі інформації у всіх сферах людської діяльності. Сучасне виробництво неможливо уявити без застосування комп'ютерів — основного технічного засобу оброблення інформації. Постійно зростає роль інформації та інформаційних технологій у науці, культурі, побуті. Якщо донедавна рівень розвитку суспільства визначали за ступенем його індустріалізації, то сьогодні — за ступенем інформатизації.

Високий рівень інформатизації всіх сфер людської діяльності, надзвичайно швидкі темпи розвитку комп'ютерної і цифрової техніки ставлять підвищені вимоги до підготовки фахівців, зокрема в закладах професійної (професійно-технічної) освіти.

У навчальному посібнику висвітлено фундаментальні принципи інформатики, основні тенденції і перспективи розвитку інформаційних технологій. Для наближення до практики, наповнення курсу конкретним змістом описано пристрої для оброблення інформації і найпоширеніші програмні продукти.

Видання підготовлено відповідно до програми з інформатики, затвердженої Міністерством освіти і науки для навчальних закладів професійної (професійно-технічної) освіти.

Розділ 1
ЗАГАЛЬНІ ВІДОМОСТІ ПРО ІНФОРМАТИКУ

1.1. ЗАГАЛЬНЕ ПОНЯТТЯ ПРО ІНФОРМАТИКУ

Інформатика — це наука, яка досліджує процеси отримання, передавання, оброблення, зберігання, подання інформації та застосування її в усіх сферах людської діяльності. Назва походить від французьких слів *information* — «інформація» та *automatioque* — «автоматика». У англомовній літературі використовують термін *Computer science* — комп'ютерні науки.

Інформатика має великий вплив на розвиток людського суспільства загалом. Недарма наш час називають інформаційною ерою. Цей період розпочався після інформаційної революції — якісного стрибка в розвитку людського суспільства, що стався після аграрної (8000–5000 рр. до н. е.) і промислової (1750–1850 рр.) революцій.

Інформатика — це дуже широке поняття, яке означає і сферу людської діяльності, і галузь науки, і навчальну дисципліну, яку викладають у школах, закладах середньої і вищої освіти. Перший факультет інформатики було засновано 1962 р. в університеті Пердью в США.

В Україні першу електронно-обчислювальну машину — МЕСМ було створено в 1951 р. у Києві в Інституті електротехніки (з 1953 р. — Інститут електродинаміки). Це була перша робоча електронно-обчислювальна машина континентальної Європи.

Велике значення для розвитку інформатики мало створення у 1957 р. Обчислювального центру Академії наук Ук-раїни, який у 1962 р. перетворено на Інститут кібернетики. Сьогодні цей заклад має ім'я його засновника — академіка Віктора Михайловича Глушкова. У 1962 р. В. М. Глушков видав монографію «Синтез цифрових автоматів», яка мала великий вплив на розвиток інформатики не тільки в Україні, а й в усьому світі.

Міжнародні організації з інформатики

Для сприяння розвитку інформатики було створено міжнародні організації. Асоціація обчислювальних машин (англ. Association for Computing Machinery, ACM) — одна з найбільших і найавторитетніших міжнародних організацій зі штаб-квартирою в Нью-Йорку, об'єднує понад 100 тисяч членів і має філіали в 56 країнах. Про результати наукової і дослідницької діяльності доповідають на міжнародних наукових конференціях, які проводять під егідою ACM, і публікують у спеціальних наукових журналах, які ця організація видає. ACM має філіали в більш ніж 500 університетах і коледжах різних країн, де навчають комп'ютерних наук.

Інститут інженерів з електротехніки та електроніки (англ. Institute of Electrical and Electronics Engineers, IEEE) — міжнародна некомерційна органі-

зація фахівців, світовий лідер у розробленні стандартів з електротехніки й електроніки. Головна мета IEEE — інформаційна і матеріальна підтримка фахівців для організації і розвитку наукової діяльності в електротехніці, електроніці, комп'ютерній техніці та інформатиці. IEEE об'єднує понад 400 тисяч індивідуальних членів зі 170 країн, видає третину світової технічної літератури, 122 реферативні журнали, проводить за рік близько 300 наукових конференцій. З інформатики IEEE видає науковий журнал «IEEE Transactions on Information Theory».

Міжнародна організація технології навчання (англ. International Society for Technology in Education, ISTE) має у своєму складі спеціальний підрозділ (англ. Special Interest Group for Computer Science, SIGCS), який сприяє вивченню інформатики, розробляє методичні рекомендації і стандарти.

Галузі інформатики

Інформатику застосовують у багатьох галузях людської діяльності, це багатопрофільна й багатогалузева наука. У навчальних закладах доцільно вивчати найважливіші галузі інформатики, з огляду на обмежений час, виділений на опанування цього предмета. ACM і IEEE, що є найавторитетнішими міжнародними організаціями з інформатики, створили спільний комітет — ACM K–12 Task Force Curriculum Committee, який після ретельного вивчення рекомендував внести до навчальної програми такі галузі інформатики:

- Алгоритми і їхня складність (Algorithms and Complexity);
- Архітектура комп'ютерів (Architecture);
- Дискретні структури (Discrete Structures);
- Комп'ютерна графіка і візуалізація (Graphics and Visual Computing);
- Взаємодія людини і комп'ютера (Human-Computer Interaction, HCI), зокрема побудова графічного інтерфейсу користувача (Graphical User Interface, GUI);
- Інформаційний менеджмент (Information Management);
- Інтелектуальні системи (Intelligent Systems);
- Комп'ютерні мережі (Net-centric Computing);
- Операційні системи (Operating Systems);
- Основи програмування (Programming Fundamentals);
- Мови програмування (Programming Languages);
- Соціальні та професійні проблеми інформатики (Social and Professional Issues);
- Програмна інженерія (Software Engineering).

Поняття інформації

Інформація (від лат. informatio — «роз'яснення, викладення, подання») — це дуже широке поняття, що має багато визначень. У цей термін вкладають різний зміст, залежно від сфери застосування. У філософії інформацію розглядають як категорію, що відображає структуру матерії і є її атрибутом. На за-

гальному рівні під інформацією розуміють факти, новини, знання, відомості про об'єкти і явища навколишнього світу та їхні властивості. Інформація є об'єктом вивчення і дослідження спеціальної науки — *теорії інформації*. У загальнішому вигляді інформацію вивчає також *кібернетика*.

Інформацію можна поділити на види за різними критеріями.

За істинністю:

- *істинна;*
- *хибна;*

за способом сприйняття:

- *візуальна* (зорова) — сприймають органами зору;
- *аудіо* (слухова) — сприймають органами слуху;
- *тактильна* — сприймають на дотик тактильними рецепторами;
- *смакова* — сприймають смаковими рецепторами;
- *нюхова* — сприймають нюховими рецепторами;

за формою подання:

- *текстова* — за допомогою букв (літер) та інших спеціальних символів (крапка, кома тощо);
- *числова* — за допомогою цифр;
- *графічна* — у вигляді зображень, графіків, креслень, фотографій тощо;
- *звукова* — у вигляді звуків;
- *відео* (анімаційна) — у вигляді рухомого зображення (кіно, телебачення тощо);

за призначенням:

- *масова* — призначена для всього населення;
- *спеціальна* — зрозуміла лише фахівцям у певній галузі;
- *секретна* — призначена лише для певного кола осіб і захищена від доступу сторонніх осіб;
- *особиста* (приватна) — набір відомостей про певну особу;

за значенням:

- *актуальна* — цінна в цей час;
- *достовірна* — без спотворень;
- *зрозуміла* — виражена так, що адекватно сприймається тим, для кого вона призначена;
- *повна* — достатня для розуміння і прийняття правильного рішення;
- *корисна* — визначається можливістю її використання конкретним суб'єктом.

Із поняттям інформації тісно пов'язане поняття «*повідомлення*», під яким розуміють певну кількість (порцію) інформації, оформлену в певний спосіб для передавання і сприйняття. Повідомлення, як і інформація, можуть бути передані у звуковій, візуальній та інших формах. Повідомлення завжди має передавача і приймача інформації.

У результаті пізнавальної діяльності людина отримує інформацію і систематизує її. Таку форму існування інформації називають **знаннями** (англ.

knowledge), тобто це суб'єктивний образ реальності через поняття та уявлення. Знання фіксуються в образах і символах природних та штучних мов. У теорії штучного інтелекту знання трактують як сукупність інформації і правил виведення.

У технічних системах, зокрема в комп'ютерах і мікропроцесорах, інформацію подають у формалізованій формі, зручній для оброблення такими системами. Таку інформацію зазвичай називають **даними** (англ. data). У комп'ютерній техніці під даними розуміють інформацію, що підлягає обробленню, на відміну від адресної інформації і команд.

Отримання, передавання, перетворення, оброблення, зберігання, поширення, пошук і використання інформації — **інформаційні процеси**.

Комплекс заходів, пристроїв, процесів для проведення (здійснення) інформаційних процесів — це **інформаційні технології (ІТ)**. На сучасному етапі розвитку під інформаційними технологіями розуміють комп'ютерні технології, а фахівців у галузі комп'ютерної техніки і програмування називають ІТ-фахівцями.

1.2. СИГНАЛИ

Сигнал — це фізичний процес, що несе інформацію, тобто матеріальний носій інформації.

За *природою фізичного процесу* сигнали поділяють на електромагнітні, електричні, світлові, звукові, пневматичні, гідравлічні тощо. У комп'ютерній техніці використовують здебільшого сигнали *електричної природи*.

Останнім часом інтенсивно розвиваються галузі, де застосовують *світлові сигнали*, які зручно передавати за допомогою оптичних волоконних кабелів. Це зумовлено такими позитивними якостями цих кабелів, як велика пропускна здатність, висока завадостійкість, відсутність взаємного впливу між каналами.

Люди, спілкуючись між собою, використовують *звукові сигнали*. *Пневматичні й гідравлічні сигнали* застосовують у певних пристроях автоматики.

Параметри сигналу, що несуть інформацію, називають інформативними. Як такі часто застосовують рівень постійної напруги або струму; амплітуду, частоту або фазу гармонійного сигналу; амплітуду, тривалість, частоту повторення імпульсів тощо. Крім інформативних параметрів сигнали мають *неінформативні* параметри, які не несуть інформації. Наприклад, якщо інформацію несе амплітуда гармонійного сигналу, то частота і фаза цього сигналу будуть неінформативними.

Види сигналів

Однією з найважливіших ознак сигналу є характер його зміни в часі й за інформативним параметром. За характером зміни в часі сигнали поділяють

на *неперервні*, або *аналогові*, й *дискретні*. У сучасних засобах комп'ютерної техніки (комп'ютерах, мікропроцесорах, контролерах) інформативним параметром є рівень електричної напруги, тому відповідну зміну сигналу часто називають зміною за рівнем сигналу. Дискретні за рівнем сигнали мають назву *квантових*.

У технічних засобах оброблення інформації використовують такі основні види сигналів.

1. Сигнали, неперервні як за інформативним параметром, так і за часом (рис. 1.1). Такі сигнали визначені в будь-який момент часу і можуть мати довільне значення в діапазоні значень, несуть інформацію про властивості об'єктів матеріального світу, як-от температура, тиск, переміщення і швидкість фізичних тіл та інші фізичні величини. У технічних засобах оброблення інформації такі сигнали сприймаються датчиками (сенсорами) і передаються далі для оброблення. Традиційно такі сигнали називають *аналоговими*.

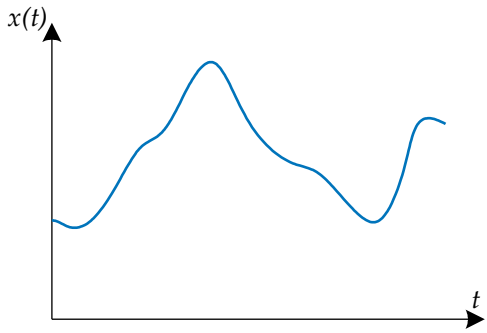


Рис. 1.1

2. Сигнали, неперервні за інформативним параметром і дискретні за часом, визначені тільки в окремі моменти часу (рис. 1.2). Дискретні за часом сигнали зручно обробляти сучасними вимірювальними та іншими пристроями для оброблення інформації, тому аналогові сигнали досліджуваних об'єктів здебільшого перетворюють на дискретні сигнали. Такий процес називають *дискретизацією* (англ. sampling) сигналу. Інтервал часу між сусідніми значеннями дискретного сигналу — це інтервал, або період дискретизації (T_s). Величину, обернену до періоду дискретизації, називають частотою дискретизації (f_s). Якщо інтервали часу між миттєвими значеннями сигналу однакові, така дискретизація є *рівномірною*, якщо неоднакові — *нерівномірною*.

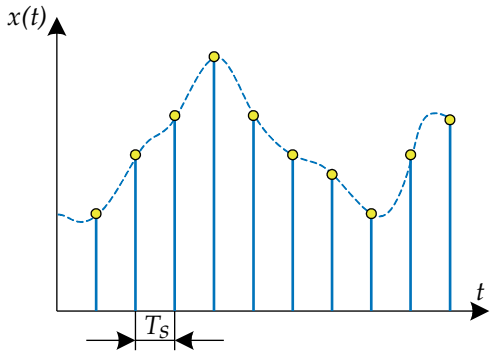


Рис. 1.2

3. Сигнали, неперервні в часі і квантовані (дискретні) за інформативним параметром, визначені в будь-який момент часу, а інформативний параметр може мати не всі значення, а тільки певну кількість дозволених значень (рівнів) (рис. 1.3). Перетворення аналогових сигналів на квантовані називають *квантуванням* сигналу. Інтервал між двома сусідніми дозволеними рівня-

ми — це *квант*. Якщо інтервали між сусідніми дозволеними рівнями однакові, то таке квантування є рівномірним, а якщо неоднакові — нерівномірним.

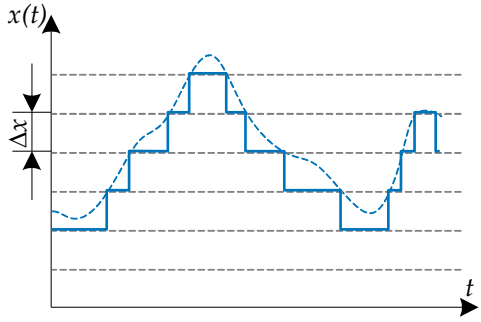


Рис. 1.3

4. У сучасних технічних засобах оброблення інформації, як правило, здійснюють як дискретизацію аналогових сигналів, так і їх квантування за допомогою спеціальних пристроїв — аналого-цифрових перетворювачів (АЦП, англ. analog digital converter, ADC). Сигнали, дискретизовані в часі і квантовані за рівнем із виходів аналого-цифрових перетворювачів, надходять на подальше оброблення. Ці сигнали визначені в певні моменти часу і можуть мати тільки певні дозволені рівні (рис. 1.4). Кількість квантів, які має сигнал у дискретні моменти часу, кодується в аналого-цифрових перетворювачах у двійковій системі числення. Саме такі дискретизовані сигнали, кількість квантів яких закодовано у двійковій системі числення, обробляються сучасними засобами оброблення інформації і мають назву *цифрових*.

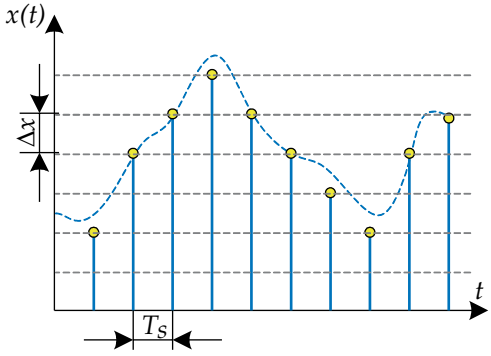


Рис. 1.4

Шуми

Шумами називають небажані, шкідливі сигнали, які перешкоджають сприйняттю корисних сигналів. У електронних пристроях розрізняють такі види шумів:

- тепловий шум, зумовлений хаотичним тепловим рухом носіїв заряду (електронів);
- дробовий шум, пов’язаний із дискретністю електричного заряду;
- флікер-шум, зумовлений наявністю домішок у провідниковому матеріалі, процесами генерації і рекомбінації носіїв заряду в напівпровідниках;
- вибуховий шум, пов’язаний із поверхневими дефектами напівпровідника;
- лавиноподібний шум, який виникає у *p-n* переході за напруги, близької до напруги пробою.

Крім шумів, на сприйняття корисних сигналів негативно впливають завади, зумовлені дією через паразитні електричні й магнітні зв’язки сигналів сусідніх каналів. Особливою інтенсивністю вирізняються завади від джерел живлення змінного струму.

Ступінь зашумленості сигналів оцінюють відношенням сигнал / шум (англ. signal-to-noise ratio, SNR), яке визначають відношенням наявного значення сигналу до середнього квадратичного значення шуму.

Найпоширеніші способи зменшення негативного впливу шумів і завад на сигнал такі: електричне й магнітне екранування пристроїв, якісне заземлення, застосування екранованих кабелів, скручування проводів тощо.

Оброблення сигналів

Оброблення сигналів (англ. signal processing) — це певні дії (операції) з сигналами і перетворення сигналів. Розглянемо найпоширеніші операції з сигналами (рис. 1.5).

Сприйняття сигналів. Людина сприймає інформацію, яку несуть сигнали, органами зору, слуху, нюху тощо. Для сприйняття сигналів у технічних, зокрема комп'ютерних і мікропроцесорних, системах використовують спеціальні пристрої — *датчики*, або *сенсори, первинні вимірювальні перетворювачі*. Датчики сприймають сигнали різної фізичної природи (температуру, тиск, переміщення, світловий потік, магнітний потік тощо) і перетворюють їх на електричну напругу, або струм. Прикладом датчика є термопара, яка перетворює різницю температур на електричну напругу.

Підсилення аналогових сигналів. Вихідні сигнали датчиків здебільшого слабкі, тож їх необхідно підсилювати. Цю операцію виконують за допомогою електронних підсилювачів. Основними параметрами підсилювачів є коефіцієнт підсилення, динамічний і частотний діапазони, рівень частотних і фазових спотворень.

Модуляція і демодуляція аналогових сигналів. *Модуляція* — це процес зміни параметра одного сигналу за законом зміни інформативного параметра іншого сигналу. Сигнал, параметр якого змінюють, називають *сигналом-носієм* (англ. carrier signal), а сигнал, що несе інформацію, — *модулювальним*.

Як сигнал-носій найчастіше використовують високочастотний синусоїдний радіосигнал або імпульсний сигнал. Залежно від того, який параметр синусоїдного сигналу змінюють — амплітуду, частоту чи фазу, розрізняють *амплі-*

Рис. 1.5

тудну, *частотну* і *фазову модуляції*. Якщо ж сигналом-носієм є імпульсний сигнал, то розрізняють *широтно-імпульсну*, *амплітудно-імпульсну*, *частотно-імпульсну*, *фазо-імпульсну модуляцію* відповідно до того, який параметр імпульсів змінюється: тривалість, амплітуда, частота або фаза.

Спектр сигналів, що несуть інформацію, здебільшого розміщується в ділянці низьких частот. Наприклад, спектр людського голосу має частотний діапазон від 20 Гц до 20 кГц; низькочастотний спектр мають також більшість вихідних сигналів датчиків. Щоб передати інформацію, яку такі сигнали несуть на велику відстань, необхідно перенести їхній спектр у смугу частот радіохвиль, що можуть поширюватися без значного ослаблення на далеку відстань. Завдання з перенесення низькочастотного спектра інформаційного сигналу в частотний діапазон радіохвиль виконує модуляція сигналів.

Демодуляція — це зворотний до модуляції процес, що полягає в перенесенні спектра інформативного сигналу з ділянки високих частот у низькочастотну ділянку.

Пристрій, який здійснює модуляцію сигналів, називають *модулятором*, демодуляцію — *демодулятором*, а пристрій, який виконує модуляцію і демодуляцію, — *модемом*.

Фільтрація аналогових і цифрових сигналів — це процес усунення або зменшення впливу небажаних чи шкідливих компонент у складі сигналу. *Фільтри* — це пристрої, які здійснюють фільтрацію сигналів. Фільтри поділяють на такі види:

- аналогові і цифрові;
- лінійні і нелінійні; пасивні і активні;
- низькочастотні, високочастотні, смугові, загороджувальні;
- для неперервних і дискретизованих сигналів;
- із нескінченною (англ. infinite impulse response, IIR) і скінченною (англ. finite impulse response, FIR) імпульсною характеристикою.

Слід зазначити, що фільтрують не тільки сигнали електричної природи, а й зображення.

Математичні операції з аналоговими і цифровими сигналами. Із сигналами можна здійснювати математичні операції: додавання, віднімання, множення, ділення, логарифмування, потенціювання (антилогарифмування), інтегрування, диференціювання тощо. Для аналогових сигналів використовують електронні пристрої на основі операційного підсилювача, охопленого петлею зворотного зв'язку, а для цифрових — спеціальні цифрові сигнальні процесори (англ. digital signal processor, DSP).

Аналого-цифрове перетворення сигналів. Комп'ютерні й мікропроцесорні пристрої сприймають і обробляють інформацію лише у формі двійкових чисел, а реальні сигнали навколишнього світу здебільшого аналогові, тому важливим є завдання з перетворення аналогових сигналів на цифрові, точніше — на послідовність двійкових чисел. Таке перетворення здійснюють за допомогою *аналого-цифрових перетворювачів* (АЦП, англ. analog-to-digital converter, ADC).

Аналого-цифрове перетворення охоплює такі операції: дискретизацію вхідного аналогового сигналу з частотою f_s, яку називають частотою дискретизації; квантування на N рівнів зі ступенем Δ_x дискретизованих сигналів; подання кількості рівнів миттєвого значення сигналу двійковим числом.

Аналого-цифрові перетворювачі мають такі основні параметри:

- роздільна здатність (англ. resolution), яка визначається кількістю рівнів квантування діапазону сигналу або кількістю двійкових розрядів;
- швидкодія, що характеризується періодом дискретизації T_S перетворюваного сигналу;
- точність, яка характеризується найменшим значущим розрядом (англ. least significant bit, LSB). Наприклад, для 8-розрядного АЦП найменший значущий розряд становить $1/2^8 = 1/256$ від повної шкали (діапазону), або 0,4 %.

На практиці використовують багато типів АЦП, найпоширенішими з яких є АЦП подвійного інтегрування, Сигма-дельта АЦП, АЦП порозрядного врівноважування, паралельні АЦП.

Цифро-аналогове перетворення сигналів. Інформація у формі двійкових чисел, з якою оперує комп'ютер, не завжди є зручною для сприйняття людиною, і тому є потреба перетворити двійковий код на аналоговий сигнал. Якщо комп'ютер або процесор використовують для керування технічними пристроями або технологічними процесами, таке перетворення також потрібне. Пристрої, які здійснюють перетворення послідовності двійкових чисел на аналоговий сигнал, називають цифро-аналоговими перетворювачами (ЦАП, англ. digital-to-analog converter, DAC).

Параметри цифро-аналогових перетворювачів аналогічні параметрам аналого-цифрових перетворювачів.

1.3. ПЕРЕТВОРЕННЯ ІНФОРМАЦІЇ В УНІВЕРСАЛЬНУ ФОРМУ ДВІЙКОВИХ ЧИСЕЛ

Людина сприймає різноманітну інформацію про навколишній світ через органи чуття, тимчасом як комп'ютер може сприймати й обробляти лише цифрові, тобто дискретизовані сигнали. Кожна дискрета квантується, і кількість квантів кодується у двійковій системі числення. Іншими словами, на вхід пристроїв оброблення інформації надходить послідовність двійкових чисел. Тому постає завдання перетворити різні форми інформації — текстову, графічну, звукову та інші — у форму двійкових чисел.

Перетворення текстової інформації у форму двійкових чисел

Як відомо, текстову інформацію подають за допомогою спеціальних символів: букв (літер), цифр, розділових і діакритичних знаків, знаків математичних операцій тощо. Щоб ці символи стали «зрозумілими» для комп'ютера, кожний символ увідповіднюють до двійкового числа — *коду* символу.

Уперше кодування літер і цифр було здійснено у 1840 р. з використанням двох сигналів: довгого «тире» і короткого «крапка». Така система кодування дістала назву «азбука Морзе». За її допомогою зручно було передавати повідомлення на великі відстані, здійснюючи довгі й короткі натискання на простий телеграфний ключ.

Досконаліший 5-розрядний код запропонував у 1870 р. французький телеграфний інженер Бодо (Émile Baudot). Код Бодо згодом став стандартом у телеграфії.

Для задоволення вимог комп'ютерної техніки, яка інтенсивно розвивалась, Американська асоціація стандартів (American Standards Association, ASA) розробила стандарт ASCII (American Standard Code for Information Interchange), за яким кодували 26 великих і малих літер англійського алфавіту, цифри, розділові знаки за допомогою 7-розрядних двійкових чисел. Крім того, стандарт охоплював керівні символи, що не підлягали друкуванню.

Поширення комп'ютерів у всьому світі, а не лише в англомовних країнах, зумовило потребу відображати національні алфавіти. Оскільки в комп'ютерах широко використовують байтове, тобто 8-розрядне подання двійкових чисел, кодування ASCII було розширено до восьми розрядів. Молодші 128 чисел відповідали попередній версії стандарту (за значення 0 найстаршого розряду), а старші 128 чисел відводили для кодування національних алфавітів (рис. 1.6). Міжнародна організація ISO (International Standardization Organization, Міжнародна організація зі стандартів) прийняла групу стандартів ISO 8859, які ви-

b_7 →					0	0	0	0	1	1	1	1
b_6 →					0	0	1	1	0	0	1	1
b_5 →					0	1	0	1	0	1	0	1
Bits	b_4 ↓	b_3 ↓	b_2 ↓	b_1 ↓	Row↓ / Column →	0	1	2	3	4	5	6	7
	0	0	0	0	0	NUL	DLE	SP	0	@	P		P
	0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
	0	0	1	0	2	STX	DC2	"	2	B	R	b	r
	0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
	0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
	0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
	0	1	1	0	6	ACK	SYN	&	6	F	V	f	V
	0	1	1	1	7	BEL	ETB	'	7	G	w	g	w
	1	0	0	0	8	BS	CAN	(	8	H	X	h	X
	1	0	0	1	9	HT	EM	)	9	I	Y	i	Y
	1	0	1	0	10	LF	SUB	*	:	J	z	j	2
	1	0	1	1	11	VT	ESC	+	;	K	[	k	{
	1	1	0	0	12	FF	FC	.	<	L	\	l	l
	1	1	0	1	13	CR	GS	–	=	Ml	]	m	}
	1	1	1	0	14	SO	RS	-	>	N	A	n	~
	1	1	1	1	15	SI	US	1	?	0	_	0	DEL

Рис. 1.6

значають 8-розрядні коди для різних груп національних мов. Наприклад, ISO 8859-1 — для США і Західної Європи, ISO 8859-5 — для кодування кирилиці. Однак така система кодування в нашій країні з історичних причин не прижилась, натомість використовували:

- Code Page 866 (CP866);
- KOI-8U — для кодування українського алфавіту;
- Code Page 1251, CP1251, Windows-1251 — систему кодування, розроблену компанією Microsoft для підтримки кирилиці в Інтернеті.

Із розвитком комп'ютерної техніки й Інтернету 8-розрядне кодування вже не відповідало новим вимогам. У 1991 р. некомерційна організація Консорціум Юнікоду (англ. Unicode Consortium) запропонувала новий стандарт кодування **Юнікод (Unicode)**, тобто універсальний код, який забезпечить двійкове подання символів усіх писемностей світу, зокрема ієрогліфічної писемності й мертвих мов. Юнікод використовує 16-розрядне, тобто двобайтне кодування символів, що дасть змогу однозначно подавати символи всіх відомих графічних знаків. Важливо зазначити, що *Юнікод* дає змогу подавати символи мов різних рас і національностей, а також *код Брайля* для людей з обмеженими фізичними властивостями (незрячих).

Перетворення графічної інформації у форму двійкових чисел

До графічної інформації належать малюнки, картини, креслення, географічні й топографічні карти, графіки, діаграми, фотографії та інші зображення предметів навколишнього світу. Цікаво, що людина для передавання інформації спочатку застосовувала графічну форму і лише набагато пізніше — текстову.

Графічна форма інформації в комп'ютерах є об'єктом дослідження **комп'ютерної графіки** — комплексу принципів, методів, засобів і технологій створення й оброблення графічної інформації. Одним із найважливіших завдань комп'ютерної графіки є перетворення графічної інформації у форму двійкових чисел.

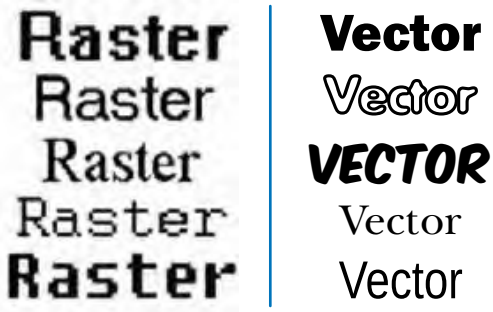


Рис. 1.7

Існує декілька способів такого перетворення (рис. 1.7).

За векторного способу графічне зображення подають як сукупність елементарних графічних об'єктів (графічних примітивів): ліній, ламаних ліній, багатокутників (полігонів), сплайнів, кривих Безьє (Bézier curves) тощо. Кожен такий графічний об'єкт описують математичним виразом (формулою). У пам'яті комп'ютера зберігаються коефіцієнти формули, дані про колір лінії, її товщину тощо. Наприклад, щоб зобразити коло, потрібно знати: тип цього елементарного графічного об'єкта, його радіус і координати (абсцису й ординату) його центра.

Математичний опис графічних об'єктів дає змогу просто й ефективно виконувати перетворення графічних об'єктів: переміщення, зміну масштабу, поворот, розтягування (стиснення), скошування тощо за допомогою аналітичного перетворення (так зване *афінне перетворення*) математичних виразів, що описують ці графічні об'єкти.

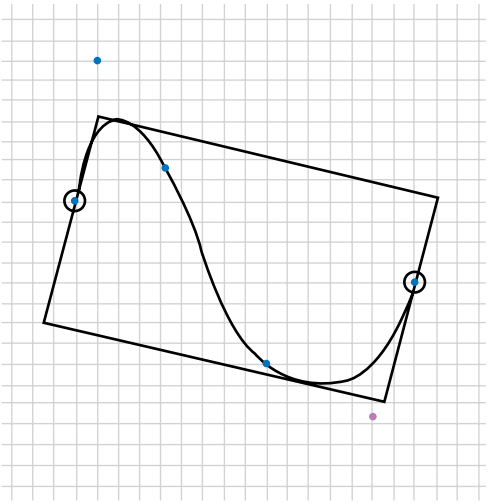


Рис. 1.8

Дуже багато можливостей при зображенні складних графічних об'єктів надає крива, яку запропонував у 1964 р. французький інженер П'єр Безьє (Pierre Bézier). Крива Безьє (рис. 1.8) описується поліномом третього порядку, хоча іноді застосовують і поліноми першого та другого порядку. Змінюючи значення коефіцієнтів полінома, можна змінити форму кривої. Таку зміну значень коефіцієнтів здійснюють опосередковано за допомогою керівних (вузлових) точок і відрізків, дотичних до кривої в початковій і кінцевій точках. З'єднуючи послідовно криві Безьє, можна апроксимувати замкнутий і розімкнутий контури будь-якого ступеня складності.

За растрового способу графічне зображення подають як прямокутну матрицю елементів — пікселів (англ. pixel — picture element), на які поділяється початкове зображення. На якість зображення впливають такі параметри і характеристики.

Розмір зображення залежить від кількості пікселів. Визначають або загальну кількість пікселів (зазвичай у мегапікселях; 1 мегапіксель = мільйон (10^6) пікселів), або окремо кількість пікселів за горизонталлю й вертикаллю, наприклад: 1600×1200,1400×1050, 1280×1024, 1280×960, 1152×864, 1024×768, 800×600.

Роздільна здатність — це кількість точок на дюйм (англ. dpi — dots per inch) або кількість пікселів на дюйм (англ. ppi — pixels per inch). Чим більша роздільна здатність, тим реалістичніше виглядає зображення.

Кожен піксель має колір. Відомо, що будь-який колір або відтінок можна виразити через систему трьох основних кольорів — *модель кольору* (англ. color model). Сітківка людського ока також складається з трьох типів кольорових рецепторів (колбочок). Найпоширеніші моделі кольорів у комп'ютерній техніці — RGB і CMYK.

Абревіатура *RGB* складається з перших літер назв трьох основних кольорів англійською: червоний (англ. red), зелений (англ. green), блакитний (англ. blue). Система RGB належить до адитивного типу моделей, тобто будь-який колір створюється додаванням у різних пропорціях трьох основних кольорів — червоного, зеленого й блакитного. RGB застосовують для виведення кольорових зображень на екрани моніторів.

У поліграфії використовують субтрактивну (англ. subtract — віднімати) систему кольорів *CMYK* (основні кольори: синьо-зелений (англ. cyan), пурпурний (англ. magenta), жовтий (англ. yellow) і додатковий чорний (англ. black)). Оскільки в поліграфії при нанесенні зображення на папір або інший носій важливо знати, який колір відбивається, а не який поглинається, то три основні кольори CMYK віднімаються від білого кольору.

Отже, колір кожного пікселя в растровому зображенні кодується трьома двійковими числами, які відповідають частці кожного з основних кольорів. Точність відтворення кольору визначається кількістю градацій кожного з основних кольорів, тобто діапазоном двійкових чисел, якими кодуються основні кольори. Крім того, точність відтворення кольору можна характеризувати кількістю розрядів цих двійкових чисел. Такий параметр часто називають *глибиною кольору*.

Кодування зображення матрицею чисел за растрового способу дає широкі можливості для оброблення таких зображень, оскільки є багато математичних методів оброблення числових масивів. Застосовуючи такі математичні методи, можна усувати дефекти зображень, коригувати кольорову гаму, покращувати реалістичність зображень.

Позитивною властивістю растрового способу є здатність створювати зображення будь-якого ступеня складності, наприклад картини живої природи, портрети людей тощо. Крім того, багато пристроїв видають зображення у растровому вигляді: цифрові фотоапарати, відеокамери, сканери тощо.

Основним недоліком растрового способу є значний обсяг цифрових даних порівняно з векторним способом. Для зберігання, оброблення і передавання растрових зображень потрібні значні апаратурні затрати. Неможливо також масштабувати зображення без втрати його якості (рис. 1.9).

Рис. 1.9

Перетворення звукової інформації у форму двійкових чисел

Людський голос, звук музичних інструментів, звуки в живій природі — всі ці звукові сигнали поширюються в пружному середовищі (газах, рідинах, твердих тілах) у вигляді хвиль і переносять інформацію від джерела до приймача.

Людина сприймає звукову інформацію органами чуття — вухами. У технічних пристроях звуковий сигнал, що несе інформацію, сприймається спеціальним датчиком — *мікрофоном*, який перетворює коливання тиску пружного середовища на переміщення мембрани, яке своєю чергою перетворюється на електричний сигнал — коливання електричної напруги або струму.

Вихідний сигнал мікрофонів надто слабкий, і його потрібно підсилити. Це завдання виконує *електронний підсилювач*, який стоїть на виході мікрофона. Після підсилення сигнал фільтрується за допомогою аналогового фільтра,

який зменшує рівень шумів і усуває явище накладання спектрів за дискретизації (англ. aliasing). Далі аналого-цифровий перетворювач перетворює сигнал на послідовність двійкових чисел, які може обробляти комп'ютер.

Звукова інформація, перетворена в цифрову форму, зберігається в пам'яті у вигляді звукових файлів (аудіофайлів) певного формату. *Формат* звукового файлу визначається способом, структурою і особливостями подання звукової інформації для її збереження в пам'яті комп'ютера або на інших цифрових носіях (*CD*, *DVD*). Найпоширенішими є такі групи форматів звукових файлів:

- без стиснення інформації;
- зі стисненням інформації без її втрат;
- зі стисненням інформації із втратами.

Звукова інформація зберігається без стиснення як послідовність двійкових чисел, яка надходить з аналого-цифрового перетворювача, тобто в необробленому «сирому» вигляді. Найвідомішими форматами такого виду звукових файлів є WAV і AIFF.

Стиснення звукової інформації — це перетворення її з однієї форми на іншу з метою зменшення обсягу. Цей процес ґрунтується на усуненні статистичної надлишковості, яка міститься у вихідних даних. Ступінь стиснення характеризується *коефіцієнтом стиснення*, який дорівнює відношенню обсягу вихідних даних до обсягу стиснених даних.

Стиснення відбувається *без втрат*, якщо можливо відновити вихідні дані без спотворень, і з втратами, якщо під час відновлення вихідних даних стаються спотворення. Найпоширеніші формати стиснення без втрат — FLAC, APE, а із втратами — MP3.

У стисненні без втрат використовують апріорну інформацію про те, які дані трапляються частіше, а які — рідше. Тоді дані, які трапляються частіше, кодують двійковими числами з меншою кількістю розрядів, а які рідше — більшою.

Принцип стиснення звукової інформації з втратами полягає в тому, що істотна інформація зберігається, а неістотна, яку все одно не можна сприйняти через психофізіологічні особливості слуху, — відкидається. Наприклад, слабкий звуковий сигнал на фоні сильного на слух не сприймається, тому його можна відкинути без істотного погіршення якості звучання.

Перетворення відеоінформації у форму двійкових чисел

Відеоінформація (від лат. video — «дивлюся, бачу») — інформація у формі рухомих зображень. Сприйняття людиною рухомих зображень ґрунтується на *інерції зору*, або персистенції (від лат. persist — «постійно перебувати, залишатися») — психофізичній особливості зорового сприйняття дискретних послідовних зображень як неперервних. Око людини має інерційність, яка полягає в тому, що після початку впливу світла на зоровий аналізатор відчуття наростає приблизно за 0,1–0,25 секунди. Якщо змінювати зображення положення предмета в просторі за менший інтервал часу, це сприйматиметься оком як неперервний рух.

Одним із найважливіших параметрів якості відеоінформації є *частота кадрів* (англ. frame rate) — кількість зображень за одиницю часу (секунду) (англ. frames per second, FPS). У разі збільшення частоти кадрів збільшується *плавність руху*. Частота кадрів у німому кінематографі була 16 кадрів за секунду, а із розвитком звукового кіно зросла до 24 кадрів за секунду.

Кожен кадр відеоінформації перетворюють у форму двійкових чисел як графічне зображення і далі формують послідовність кадрів в інформаційний потік. Обсяг відеоінформації набагато перевищує обсяги інших видів інформації.

Мультимедіа

Мультимедіа (англ. multimedia, від лат. multum — «багато» і medium — «середовище») — комбінація текстової, графічної, звукової, а останнім часом відео- та анімаційної інформації в одному *інформаційному* об'єкті. Багато сучасних засобів мультимедіа працюють в інтерактивному режимі, тобто споживач інформації може впливати на процес мультимедійного подання.

Мультимедійне подання інформації широко застосовують у наукових дослідженнях, сучасному виробництві, освіті, рекламі.

Під час навчання мультимедійне подання інформації, діючи на різні канали сприйняття, дає змогу глибше зрозуміти й засвоїти матеріал, максимально наблизитись до реальності.

ТЕСТОВІ ЗАВДАННЯ

ЩО МИ РОЗУМІЄМО ПІД ІНФОРМАЦІЄЮ?

1. Слух, зір, смак, запах, відчуття
2. Зберігання, передавання, оброблення, пошук даних
3. Відомості про предмети, події, явища та процеси навколишнього світу
4. Текст, звук, номер, графіка

ЦИФРОВІ СИГНАЛИ — ЦЕ:

1. сигнали, неперервні як за інформативним параметром, так і за часом
2. сигнали, неперервні за інформативним параметром і дискретні за часом
3. сигнали, що визначені в певні моменти часу і можуть мати тільки певні дозволені рівні
4. сигнали, неперервні в часі й квантовані (дискретні) за інформативним параметром, визначені в будь-який момент часу, а інформативний параметр може мати не всі значення, а тільки певну кількість дозволених значень (рівнів)

Розділ 2
КОМП'ЮТЕР — ОСНОВНИЙ ЗАСІБ ОБРОБЛЕННЯ ІНФОРМАЦІЇ

У процесі історичного розвитку людство створило багато технічних пристроїв, машин, механізмів для полегшення і підвищення продуктивності праці людини. Широко відомими є машини, які застосовують у промисловості для оброблення матеріалів — металів, дерева, пластмас, для виробництва й перетворення енергії. Важко уявити собі транспортну галузь або сучасне сільське господарство без застосування транспортних і технічних засобів.

Комп'ютер, на відміну від інших технічних засобів, що полегшують фізичну працю і пов'язані з матеріалами або енергією, має справу з інформацією і призначений для полегшення та підвищення продуктивності розумової праці людини. Комп'ютер звільняє людину від малопродуктивної, рутинної виснажливої роботи й дає змогу зайнятися творчою працею, підвищуючи тим самим її продуктивність.

У попередньому розділі було розглянуто поняття інформації, її основні види та форми. Для перетворення інформації з однієї форми в іншу, оброблення інформації та здійснення інших інформаційних процесів було винайдено і створено спеціальні технічні засоби, основний серед яких — комп'ютер. Інформацію, що надходить у комп'ютер і підлягає обробленню, у комп'ютерних науках називають *даними*.

2.1. ПРИНЦИПИ РОБОТИ КОМП'ЮТЕРА

Комп'ютер як технічний пристрій для оброблення інформації побудований на таких основних принципах.

Принцип двійкового кодування інформації полягає у тому, що з усіх форм і видів інформації для сприйняття, запам'ятовування та оброблення комп'ютером вибрано єдину форму подання інформації — форму двійкових чисел.

Усе різноманіття форм і видів інформації, яку потрібно ввести й обробити комп'ютером, перетворюється у форму двійкових чисел. Це універсальна форма подання інформації, до якої можна звести всі інші форми інформації. Наприклад, числова інформація у звичній для сприйняття людиною десятковій системі числення, а також букви, розділові знаки, спеціальні символи текстової інформації перетворюються на двійкову систему числення згідно зі стандартами ASCII та Unicode. Графічна інформація поділяється на елементи, кожен із яких також перетворюється у форму двійкових чисел. У формі двійкових чисел кодується також колір графічного зображення.

Широке застосування двійкових чисел як універсальної форми подання інформації зумовлено такими важливими причинами. По-перше, двійкова система числення є близькою до оптимальної. Як відомо з математики, оптимальною є система числення з основою $\pi = 3{,}14$. По-друге, у двійковій системі числення виражають логічні змінні величини й логічні функції. По-третє, арифметичні операції із багаторозрядними двійковими числами (додавання, віднімання, множення, ділення) можна подати через логічні операції з однорозрядними двійковими числами. І, головне, для виконання логічних операцій створено технічні пристрої, які називають логічними елементами.

Принцип програмного керування. Видатний англійський математик Ч. Беббідж у 1833 р. розробив проєкт універсальної цифрової обчислювальної машини з програмним керуванням (прообраз сучасної ЕОМ), а разом із соратницею Адою Авґустою Лавлейс, донькою англійського поета Дж. Байрона, обґрунтував і показав, що аналітичну машину можна розглядати як засіб створення формул, музики, картин. В одній із заміток Лавлейс описала принцип обчислень чисел Бернуллі на аналітичній машині, який у подальшому було визнано першою програмою. Принцип керування полягає в тому, що оброблення інформації комп'ютером здійснюється в *автоматичному* режимі під керуванням *програми*. Термін «автомат» походить із давньогрецької мови й означає «такий, що діє сам», тобто комп'ютером керує не людина, а програма. Під програмою розуміють послідовність команд — розпоряджень або вказівок для виконання певних операцій (дій). Крім терміна «команда», вживаються також терміни «інструкція», «директива». Поняття програми тісно пов'язане з поняттям алгоритму.

Алгоритм — одне з фундаментальних понять математики й інформатики. Міжнародна організація стандартів (ISO) тлумачить алгоритм як «скінченну сукупність розпоряджень, які визначають розв'язок проблеми (задачі) за допомогою скінченної кількості операцій (дій)». **Програма** — це запис алгоритму за певною системою правил, яку називають *алгоритмічною мовою*.

Отже, комп'ютер обробляє інформацію в автоматичному режимі під дією програми, складеної людиною (програмістом) на основі розробленого алгоритму розв'язку проблеми (задачі).

Принцип програми, що зберігається в пам'яті комп'ютера, запропонував американський учений Джон фон Нейман у 1945 р. Така програма має значні переваги. Зазвичай у пам'ять комп'ютера завантажують багато програм різного рівня і характеру. Замінивши в пам'яті одну програму іншою, можна застосувати комп'ютер для розв'язання різних задач. Тобто програма, що зберігається в пам'яті комп'ютера, робить його *універсальним* пристроєм для оброблення інформації. Крім того, це дає змогу коригувати і змінювати саму програму, наприклад для поліпшення її характеристик та налагодження.

Принцип однорідності пам'яті означає, що команди програми подаються у формі двійкових чисел, як і інформація, що підлягає обробленню, і можуть зберігатися в такій самій пам'яті, тобто пам'ять є однорідною як для да-

них, так і для команд. Це дає змогу виконувати з командами такі самі операції, як і з даними.

Здатність виконувати операції з командами програми є дуже важливою. По-перше, змінюючи деякі команди в процесі їх виконання залежно від результатів попередньої роботи, можна значно підвищити гнучкість і адаптивність програм. По-друге, можна за допомогою команд однієї програми обробляти команди іншої програми. На цьому принципі ґрунтується дія трансляторів, тобто спеціальних програм, які перетворюють програми, написані однією алгоритмічною мовою, на програми, написані іншою алгоритмічною мовою.

Якщо дані й програми зберігаються в єдиній пам'яті, то такі комп'ютери належать до *принстонського* типу (за назвою Принстонського університету (США), де запропонували цю ідею). У комп'ютерах *гарвардського* типу (назвали на честь Гарвардського університету (США), вчені якого розробили цю структуру) дані й команди зберігаються окремо одне від одного.

Принцип адресності пам'яті. Пам'ять комп'ютера, у якій зберігаються програми й дані, побудовано у вигляді послідовності пронумерованих *елементів* пам'яті, які називають *клітинками* пам'яті, або *комірками* пам'яті (англ. cells memory). Порядковий номер елемента пам'яті — це його *адреса*.

Адресу елемента пам'яті в комп'ютері подано двійковим числом. Оскільки двійкова система числення є універсальною, за її допомогою в комп'ютері подають і дані, що підлягають обробленню, і команди програми, і адреси команд та даних.

Двійкове подання адрес команд і даних дає змогу виконувати з адресами такі самі операції, як і з даними й командами за допомогою таких самих технічних засобів.

У кожному елементі пам'яті може розміщуватися (запам'ятовуватися) одне двійкове число. Якщо двійкове число має вісім розрядів, його називають *байтом* (англ. byte), а відповідний елемент пам'яті, у якому розміщується таке двійкове число, — *однобайтним* елементом пам'яті.

На сучасному етапі розвитку комп'ютерної техніки широко застосовують пам'ять із 4-байтними (32-розрядними) і 8-байтними (64-розрядними) елементами пам'яті. Двійкове число, що розміщується в одному елементі пам'яті, в інформатиці часто називають *словом* (англ. word).

Згідно з **принципом послідовного виконання команд** команди для оброблення інформації виконуються одна за одною. Щоб забезпечити такий процес, команди також розміщуються в пам'яті послідовно, одна за одною, тобто наступна команда в пам'яті — за наступною адресою. Після виконання чергової команди виконуватиметься наступна команда, тобто команда, адреса якої на одиницю більша за адресу поточної команди.

Змінити такий порядок виконання команд можна *спеціальними командами — командами розгалуження* або *командами передавання керування*. Якщо у програмі трапляється така команда, то аналізується результат попередньої операції або спеціальний сигнал і залежно від результатів такого аналізу ви-

конується або команда за наступною адресою, тобто продовжується звичний порядок виконання команд, або команда із заздалегідь обумовленою адресою, яка в явному чи неявному вигляді міститься в цій програмі.

Саме принцип послідовного виконання команд і наявність спеціальних команд передавання керування забезпечують *автоматичний режим роботи комп'ютера*, тобто оброблення інформації комп'ютером без втручання людини.

2.2. ІЄРАРХІЧНІ РІВНІ КОМП'ЮТЕРА

Комп'ютер є технічним пристроєм, що має складну ієрархічну структуру. Щоб зрозуміти роботу комп'ютера, потрібно проаналізувати процеси в ньому на всіх рівнях ієрархії.

Фізичний рівень. Найнижчий рівень ієрархії — фізичний, точніше, електротехнічний рівень. Основними об'єктами на цьому рівні є електронні елементи — *транзистори, конденсатори, резистори*, носіями інформації — *електричні сигнали*. Процеси перетворення електричних сигналів в електронних елементах аналізують методами електротехніки й електроніки. Саме на цьому рівні розв'язують проблеми підвищення швидкодії пристроїв комп'ютера і зменшення енергоспоживання.

Рівень логічних елементів. Основними об'єктами аналізу на цьому рівні є *логічні елементи*, які виконують операції з *логічними сигналами*. Найпоширенішими є такі логічні елементи:

- елемент «І» («AND»), який виконує операцію кон'юнкції;
- елемент «АБО» («OR»), який виконує операцію диз'юнкції;
- елемент «НЕ» (NOT), який виконує операцію заперечення;
- елемент «І-НЕ» (NAND), який виконує операцію заперечення кон'юнкції тощо.

Логічні сигнали пов'язані з сигналами на попередньому фізичному рівні такою залежністю: весь діапазон можливої зміни фізичного сигналу (у переважній більшості випадків — це електрична напруга) поділяється на три зони — зона логічного 0, зона логічної 1, зона невизначеності, яка міститься між двома попередніми зонами.

Рівень цифрових пристроїв. Цифрові пристрої складаються з логічних елементів, з'єднаних у певний спосіб. Цифрові пристрої поділяють на *комбінаційні цифрові*, у яких вихідні логічні сигнали визначаються лише вхідними сигналами і структурою зв'язків між логічними елементами, і *цифрові пристрої з пам'яттю*, у яких вихідні логічні сигнали пристрою залежать як від вхідних, так і від вихідних сигналів логічних елементів, що входять до складу цього пристрою.

У комп'ютерах застосовують такі цифрові логічні пристрої:

- регістри — пристрої для зберігання двійкових чисел (двійкових слів);
- суматори — пристрої для додавання і віднімання двійкових чисел;
- дешифратори і шифратори — пристрої для перетворення двійкових чисел з однієї форми в іншу;
- мультиплексори і демультиплексори — пристрої для підключення окремих розрядів двійкового числа до певних пристроїв;

- пристрої порівняння — пристрої для порівняння двійкових чисел і формування команд: «більше», «менше», «рівно» тощо.

На цьому ієрархічному рівні операції виконують із багаторозрядними двійковими числами. Зв'язок із попереднім нижчим рівнем такий: кожен розряд двійкового числа — це логічний сигнал попереднього рівня. Цифрові пристрої — багаторозрядні, причому кожен розряд побудований на логічних елементах і має аналогічну структуру.

Крім *даних*, поданих послідовністю двійкових чисел, що обробляються цифровими пристроями, на цифрові пристрої надходять також *сигнали керування*: сигнал запису в цифровий пристрій, сигнал читання, сигнал прийому даних, сигнал видавання даних тощо.

Ієрархічні рівні команд. Сукупність команд комп'ютера утворює досить складну ієрархічну систему, тому пам'ять комп'ютера також має складну ієрархічну структуру. У процесі оброблення інформації в комп'ютері використовуються команди різного ієрархічного рівня.

Найнижчий ієрархічний рівень становлять **мікрокоманди** — логічні сигнали, що мають лише два значення, які керують певним цифровим пристроєм процесора: регістром, суматором, драйвером шини тощо. Якщо логічний сигнал має значення «1», це означає «виконати певну мікрооперацію». Найпоширенішими є мікрокоманди на виконання таких мікрооперацій:

- запис даних у регістр;
- активація драйвера шини, тобто видання даних на шину;
- читання даних із регістру тощо.

Мікрокоманди завжди формуються апаратним способом, а саме пристроєм керування процесора.

Машинні команди — це двійкові числа, які пересилаються з пам'яті у процесор і у яких закодовано вказівку процесору виконати певні дії з даними. Найпоширенішими машинними командами є арифметичні (додавання, віднімання), логічні (диз'юнкція, кон'юнкція, заперечення, сума за модулем 2), команди передавання керування тощо.

Машинні команди, на відміну від мікрокоманд, можуть бути реалізовані як *апаратно*, так і *програмно*. У разі *апаратної реалізації* кожну машинну команду виконує окремий цифровий пристрій. Якщо ж машинні команди реалізовані *програмно*, то кожній машинній команді відповідає мікропрограма, яка зберігається у так званій *керівній пам'яті* процесора. Мікропрограма складається з мікрокоманд, які реалізують алгоритм виконання машинної команди. Наприклад, машинна команда додавання складається з мікрокоманд, що реалізують такий алгоритм:

- записати перший доданок у перший регістр суматора;
- записати другий доданок у другий регістр суматора;
- результат операції (суму двох чисел) записати в акумулятор, тобто регістр, що приєднаний до виходу суматора.

Команди асемблера. Як уже неодноразово зазначено, машинні команди — це двійкові числа. Дані, що підлягають обробленню, адреси команд і

даних у пам'яті — це також двійкові числа. У сучасних комп'ютерах кількість розрядів двійкових чисел сягає 64 і більше.

На початку розвитку комп'ютерної техніки програмістові, щоб написати програму, потрібно було пам'ятати двійкові коди всіх машинних команд процесора, двійкові коди адрес, де зберігаються команди й дані. Зрозуміло, що написання програми в такий спосіб — це важка і виснажлива праця, яка до того ж малопродуктивна.

Щоб полегшити написання програм, було запропоновано замість двійкових кодів машинних команд писати мнемонічні імена команд, які людина запам'ятовує набагато легше, ніж двійкові числа (слово «мнемонічний» у перекладі з давньогрецької мови означає «те, що легко запам'ятовується»). Наприклад, замість двійкового коду *10000111* машинної команди «додавання» писати ім'я команди *ADD* (від англ. adding — «додавати») або ім'я *SUB* (від англ. subtraction — «віднімання»), *MOV* (від англ. move — «переміщувати») — команду переміщення двійкового числа з одного пристрою на інший.

Спеціальні програми-*асемблери* (англ. assembler — «складальник») перетворюють мнемонічні імена команд *ADD, SUB, MOV* тощо на двійкові числа. Простіше кажучи, асемблер — це «перекладач» із мови мнемонічних імен мовою машинних команд. Мову мнемонічних імен називають *мовою асемблера*, як і саму програму.

Команди алгоритмічних мов програмування. Писати програми за допомогою команд асемблера може лише висококваліфікований фахівець, який досконало знає будову й особливості певного комп'ютера. Це обмежує доступ до комп'ютера широких верств користувачів, які працюють у різних галузях. Виникає так званий семантичний розрив між системою понять користувачів комп'ютера і мовою машинних команд, яку «розуміє» комп'ютер.

Щоб подолати цей семантичний розрив, було розроблено спеціальні алгоритмічні мови високого рівня (Pascal, C++, Java тощо), які максимально наближені до традиційних засобів спілкування. Програму алгоритмічними мовами може написати користувач, якому не обов'язково знати будову, принципи функціонування й особливості комп'ютера.

Написана алгоритмічною мовою програма перетворюється на послідовність машинних команд без участі користувача за допомогою спеціальних програм-перекладачів, які входять до складу операційної системи і тому мають назву системних програм. Такі системні програми поділяють на два види: інтерпретатори і компілятори.

Інтерпретатори послідовно, одну за одною перетворюють (перекладають) команди алгоритмічної мови на машинні команди і одразу ж виконують ці команди.

Компілятори перетворюють програми алгоритмічними мовами на програми, що складаються з машинних команд. Далі комп'ютер виконує програму, перекладену мовою машинних команд.

Команди операційної системи. Щоб подолати семантичний розрив і полегшити взаємодію між широкими верствами користувачів і комп'ютерами,

у складі сучасних пристроїв є «посередники», тобто комплекс спеціальних програм — *операційна система* комп'ютера.

Крім інтерпретаторів і компіляторів до комплексу операційної системи входять спеціальні системні програми, які розподіляють ресурси комп'ютера, керують роботою пристроїв, здійснюють взаємодію комп'ютера і користувача тощо.

Найпоширенішими операційними системами на сучасному етапі розвитку є Windows фірми Microsoft та UNIX. Останнім часом широке застосування має некомерційна операційна система Linux, яка належить до так званого вільного (англ. free) програмного забезпечення.

Команди прикладного рівня (рівня користувача). На цьому рівні користувач запускає і виконує програми різноманітного призначення і складності: текстові і графічні редактори, бази даних, комп'ютерні ігри тощо.

2.3. АРХІТЕКТУРА фон НЕЙМАНА

Комп'ютер як технічний пристрій, призначений для уведення, оброблення і подавання інформації у формі, яка сприймається людиною, має складну будову. Щоб її описати, вживають багато визначень, зокрема «архітектура комп'ютера», «організація комп'ютера», «структура комп'ютера» тощо.

Комп'ютер у своєму історичному розвитку пройшов складний шлях від найпростіших пристроїв до сучасних технічних комплексів зі складною ієрархічною структурою. Першу науково обґрунтовану будову комп'ютера розробила група американських учених, зокрема Дж. фон Нейман, чиїм ім'ям назвали цю архітектуру (див. рис. 2.1). До складу комп'ютера, що має архітектуру фон Неймана, входять такі пристрої:

- пристрій (або пристрої) введення інформації (даних і програм), призначений для перетворення текстової, графічної, звукової та інших форм інформації у форму багаторозрядних двійкових чисел;
- процесор — пристрій, що обробляє інформацію під керуванням програми. Останнім часом для оброблення значних потоків інформації широко застосовують складні процесори, які складаються з кількох процесорів (ядер), що працюють паралельно під єдиним керуванням;
- пам'ять — пристрій, у якому зберігаються дані, що підлягають обробленню, результати оброблення, а також програма (програми), що керує обробленням інформації;
- пристрій керування, який керує уведенням, обробленням і виведенням інформації в автоматичному режимі, тобто без втручання людини;
- пристрій виведення інформації у формі, що сприймається людиною (користувачем комп'ютера).

Розглянемо докладніше будову кожного із цих пристроїв.

Пристрої введення інформації призначені для перетворення різних форм інформації (текстової, звукової, анімаційної тощо) у форму двійкових чисел. Розмаїттю форм інформації в навколишньому світі відповідають різно-

манітні пристрої введення: клавіатура призначена для введення текстової і цифрової інформації; мікрофон і звукова карта — для введення звукової інформації; відеокамера — для введення зображень.

Процесор. Інформація, перетворена пристроями введення у форму двійкових чисел, надходить до процесора, який обробляє цю інформацію, тобто виконує арифметичні, логічні та деякі інші види операцій із двійковими числами. Ці операції процесор здійснює під керуванням команд програми. Команди — це також двійкові числа, які надходять до процесора з пам'яті, де вони зберігаються.

Пам'ять призначена для зберігання інформації у формі двійкових чисел, яка надходить із пристроїв уведення. Крім того, зберігаються дані, тобто результати операцій, які виконує процесор.

Крім даних, у пам'яті зберігаються коди команд програм, які керують роботою процесора, а також адреси даних і команд.

Пристрій керування. Оброблення інформації в комп'ютері здійснюється в автоматичному режимі під керуванням програм. Для програмного керування цим процесом, координації і узгодження роботи різних пристроїв комп'ютера призначений пристрій керування.

У сучасних комп'ютерах зі складною ієрархічною будовою функції керування розподілені між усіма пристроями комп'ютера. Кожен пристрій введення і виведення має свій локальний пристрій керування, який часто називають *контролером*. Пам'ять також має свій пристрій керування. Загальне керування, координацію і узгодження роботи всіх пристроїв бере на себе *центральний пристрій керування* (англ. central processor unit, CPU).

Пристрої виведення інформації. Процесор обробляє інформацію у двійковій формі, однак людині важко її сприймати. Вихідні пристрої призначені для перетворення двійкової інформації в інші форми: текстову, звукову, анімаційну. Принтери, плотери, монітори — це пристрої виведення, призначені для подання інформації у звичній для людини формі.

2.4. ОРГАНІЗАЦІЯ ВЗАЄМОДІЇ ПРИСТРОЇВ КОМП'ЮТЕРА ПІД ЧАС ВИКОНАННЯ ПОТОЧНОЇ КОМАНДИ

Для роботи комп'ютера з оброблення інформації під керуванням програми потрібна чітка взаємодія всіх його пристроїв. Така взаємодія узгоджується і координується пристроєм керування. Розглянемо, як взаємодіють різні пристрої комп'ютера під час виконання поточної команди.

Цикл виконання машинної команди

Виконання машинної команди здійснює процесор у два етапи:

- етап вибірки чергової машинної команди із пам'яті і пересилання її у процесор, у регістр команди (рис. 2.1);

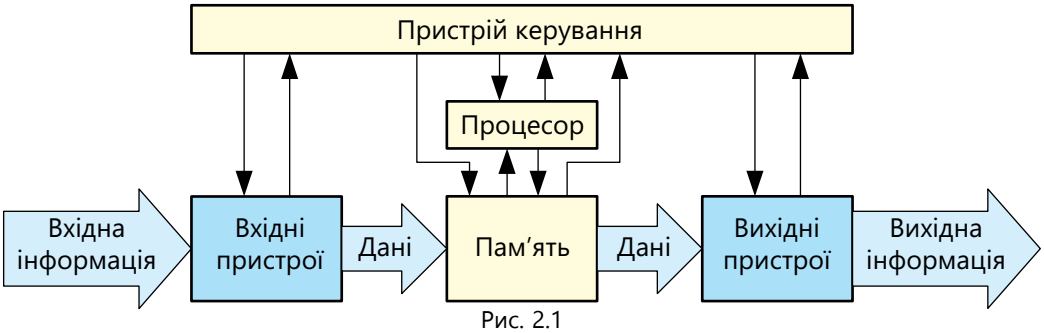


Рис. 2.1

- виконання команди, тобто певної операції з двійковими числами (операндами).

Своєю чергою кожен етап виконання машинної команди складається з послідовності *мікрооперацій*, які виконує операційний пристрій процесора.

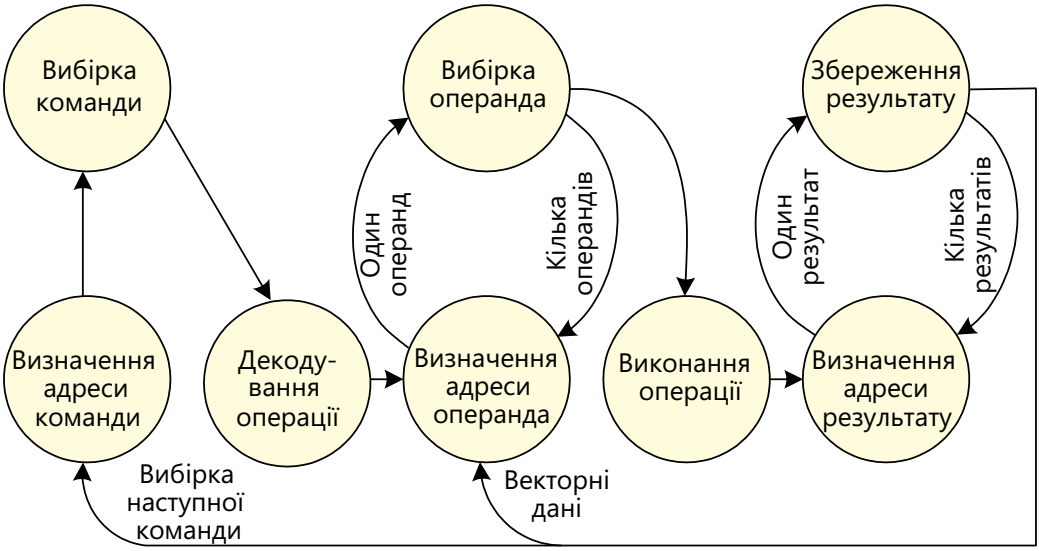


Рис. 2.2

Наприклад, виконання машинної операції додавання двох цілих чисел здійснюється в такій послідовності (рис. 2.2):

- вміст лічильника команд (PC — program counter), що містить адресу наступної команди, переміщується в регістр адреси (MAR — memory address register), який підключений до шини адреси (AB — address bus);
- за адресою, що міститься на шині адреси, вибирається елемент пам'яті, у якому міститься адреса наступної машинної команди;
- вибрана команда переміщується по шині даних (DB — data bus) в операційний пристрій процесора, у регістр команд (IR — instruction register).

На цьому закінчується етап вибірки команди і починається етап її виконання:

- розміщена в регістрі команд машинна команда декодується, тобто розкладається на послідовність елементарних операцій (мікрооперацій);
- за інформацією, що міститься в операндній (адресній) частині машинної команди, обчислюється адреса першого операнда і розміщується в регістрі адрес;
- з регістру адрес через системну шину адреса операнда надходить до пам'яті, вибирається потрібний елемент пам'яті;
- вибраний елемент пам'яті містить потрібний операнд, який через шину даних (DB — data bus) передається в процесор, у регістр даних (MDR — memory data register);
- якщо процесор двоадресний, то з пам'яті вибирається другий операнд за такою самою процедурою, як і перший;
- виконується операція, яку визначено операційною частиною машинної команди;
- якщо у процесорі реалізовано триадресну систему адресації, то формується адреса результату на основі інформації, що міститься в адресній частині машинної команди;
- сформована адреса результату завантажується в регістр адрес, через шину адрес вибирається потрібний елемент пам'яті;
- результат операції завантажується в регістр даних, через шину даних передається в пам'ять і завантажується за вибраною адресою;
- до вмісту програмного лічильника додається одиниця і здійснюється перехід до виконання наступної машинної команди.

ТЕСТОВІ ЗАВДАННЯ

КОМП'ЮТЕР – ЦЕ:

1. пристрій для автоматичного оброблення числової інформації
2. пристрій для зберігання інформації
3. пристрій для пошуку, збирання, зберігання, перетворення та використання інформації в цифровому форматі
4. сукупність програмних засобів, які керують інформаційними ресурсами

МАШИННІ КОМАНДИ — ЦЕ:

1. двійкові числа, які пересилаються з пам'яті у процесор і в яких закодовано вказівку процесору виконати певні дії з даними
2. десяткові числа, які вводять із клавіатури
3. мнемонічні імена команд, тобто буквені позначення команд
4. зарезервовані слова зі спеціального словника

Розділ 3
ЛОГІЧНІ Й АРИФМЕТИЧНІ ОСНОВИ КОМП'ЮТЕРА

Інформація в комп'ютері, як уже зазначено, подається й обробляється у формі двійкових чисел. Із двійковими числами виконують арифметичні операції, тому важливо знати властивості числової інформації, системи числення, зокрема двійкової, арифметичних операцій із двійковими числами.

Розряди двійкового числа можна трактувати як логічні величини, що мають лише два значення. Властивості логічних величин і операцій із ними є теоретичною основою для розроблення та аналізу комп'ютерних пристроїв.

3.1. ЛОГІЧНІ ОСНОВИ КОМП'ЮТЕРНОЇ ТЕХНІКИ

Логічна величина може мати одне з двох можливих значень. Такі величини називають *булевими* на честь англійського вченого Джорджа Буля, який уперше запропонував їх і дослідив.

Логічні величини відіграють велику роль у комп'ютерній техніці, оскільки операції з ними зручно реалізувати за допомогою спеціальних технічних пристроїв, що можуть мати два стани, наприклад: перемикач — «замкнено» і «розімкнено»; транзистор — «відкритий» і «закритий», магнітопровід — «намагнічено» і «розмагнічено» тощо.

Два альтернативні значення логічної величини можна позначати різними способами: двійковими цифрами 0, 1; словами «істинно», «хибно», «так», «ні» тощо. У подальшому викладі використовуватимемо для логічних змінних двійкові цифри **0** і **1**. Із логічними величинами можна виконувати операції, які називають *логічними*. Наука, яка вивчає логічні величини й операції з ними — *алгебра логіки*, або *булева алгебра*.

Логічні операції

Алгебра логіки визначає такі базові логічні операції з логічними величинами:

- операція *заперечення*, або *інверсії*, яку позначають рискою над змінною або штрихом, наприклад $\overline{x}$, або x';
- операція *диз'юнкції*, або *логічного додавання,* — символ «v» або «+»;
- операція *кон'юнкції*, або *логічного множення*, яку позначають символами «&», «$\wedge$» або «·», причому останній символ можна опускати.

Крім перелічених, у комп'ютерній техніці широко використовують логічну операцію *виключне АБО*. Цю логічну операцію часто називають також *сумою за модулем 2* і позначають символом $\oplus$.

Операція заперечення є унарною операцією, тобто операцією з однією змінною логічною величиною $y = f(x)$.

Операції диз'юнкції, кон'юнкції та суми за модулем 2 є бінарними, тобто операціями, які виконують щонайменше із двома змінними логічними величинами $y = f(x_1, x_2)$. Загальна кількість бінарних операцій із двома логічними величинами, що мають два можливі значення, дорівнює $2^{2^2} = 16$. Ці операції наведено в табл. 3.1.

Таблиця 3.1

	Набори аргументів, $x_1\ x_2$				$y = f(x_1, x_2)$
Функції	00	01	10	11	Назва функції
y_0	0	0	0	0	Константа 0
y_1	0	0	0	1	Кон'юнкція
y_2	0	0	1	0	Заперечення імплікації
y_3	0	0	1	1	Повторення x_1
y_4	0	1	0	0	Заперечення оберненої імплікації
y_5	0	1	0	1	Повторення x_2
y_6	0	1	1	0	Сума за модулем 2
y_7	0	1	1	1	Диз'юнкція
y_8	1	0	0	0	Стрілка Пірса
y_9	1	0	0	1	Еквіваленція
y_{10}	1	0	1	0	Заперечення x_2
y_{11}	1	0	1	1	Обернена імплікація
y_{12}	1	1	0	0	Заперечення x_1
y_{13}	1	1	0	1	Імплікація
y_{14}	1	1	1	0	Штрих Шефера
y_{15}	1	1	1	1	Константа 1

Таблиці істинності найпоширеніших операцій, у яких показано результати операцій за всіх можливих значень аргументів, наведено на рис. 3.1.

x_1	x_2	$y=x_1 \wedge x_2$
0	0	0
0	1	1
1	0	1
1	1	1

а

x_1	x_2	$y=x_1 \wedge x_2$
0	0	1
0	1	0
1	0	0
1	1	0

б

x_1	$y=\bar{x}$
0	1
1	0

в

x_1	x_2	$y=x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

г

Рис. 3.1

Закони алгебри логіки

Для операцій із логічними величинами справедливими є такі закони.

1. Комутативний
   $a \vee b = b \vee a$ $\qquad$ $a \wedge b = b \wedge a.$
2. Асоціативний
   $(a \wedge b) \wedge c = a \wedge (b \wedge c)$ $\qquad$ $(a \vee b) \vee c = a \vee (b \vee c).$
3. Дистрибутивний
   $a \wedge (b \vee c) = a \wedge b \vee a \wedge c$ $\qquad$ $a \vee b \wedge c = (a \vee b) \wedge (a \vee c).$
4. Ідемпотентності
   $a \vee a = a$ $\qquad$ $a \wedge a = a.$
5. Подвійного заперечення
   $\overline{\overline{a}} = a.$
6. де Моргана
   $\overline{a \wedge b} = \overline{a} \vee \overline{b}$ $\qquad$ $\overline{a \vee b} = \overline{a} \wedge \overline{b}.$
7. Поглинання
   $a \vee a \wedge b = a$ $\qquad$ $a \wedge (a \vee b) = a.$
8. Склеювання
   $x \wedge y \vee x \wedge \overline{y} = x$ $\qquad$ $(x \vee y) \wedge (x \vee \overline{y}) = x.$

Користуючись цими законами, можна перетворювати і спрощувати логічні вирази.

Застосування логічних операцій. У комп'ютерній техніці інформацію, як відомо, подано у вигляді двійкових чисел. Окремі розряди двійкового числа можна трактувати як логічні величини й виконувати з ними логічні операції. Такі операції називають побітовими (англ. bitwise) й найчастіше застосовують у комп'ютерній техніці для оброблення інформації:

- перевірка значень окремих розрядів двійкового числа;
- встановлення окремих розрядів в 1;
- встановлення окремих розрядів у 0;
- інвертування окремих розрядів.

3.2. ЛОГІЧНІ СИГНАЛИ

Логічні змінні величини в алгебрі логіки — це лише математична абстракція. Для фізичної реалізації логічних змінних використовують логічні сигнали. Сигналом, як відомо, називають фізичний процес, що несе інформацію. Відповідно логічний сигнал несе інформацію в логічній формі, тобто у формі логічних величин.

За фізичною природою логічні сигнали можуть бути електричні, пневматичні, гідравлічні. В *електричних* логічних сигналах інформативним параметром, тобто фізичною величиною, що несе інформацію, є здебільшого електрична напруга, хоча іноді застосовується електричний струм. Інформативним

параметром *пневматичного* логічного сигналу є тиск газу, здебільшого повітря, а *гідравлічного* — тиск рідини.

Оскільки логічні величини можуть мати лише два значення (0, 1), то логічний сигнал має два стани, або два рівні, — *високий* і *низький*. Сигнали, які можуть мати лише скінченну кількість значень, належать до класу *квантованих* сигналів. Отже, логічний сигнал є частковим випадком квантованого сигналу.

Логічні сигнали належать також до класу *дискретних* сигналів, тобто можуть змінюватися тільки через певний інтервал часу, який називають *періодом дискретизації*. Впродовж цього періоду дискретні сигнали, зокрема логічні, мають незмінне значення. Період дискретизації задається в цифрових пристроях спеціальним *сигналом синхронізації* (англ. clock). Отже, логічний сигнал є одночасно і квантованим, і дискретним.

На сучасному етапі розвитку цифрової техніки застосовують переважно електричні логічні сигнали, у яких двом значенням логічних величин (0, 1) відповідають два рівні електричної напруги — *високий* і *низький*.

Переважна більшість сигналів у природі, й електричні також, є *аналоговими*, тобто мають нескінченну кількість значень і можуть змінювати своє значення в будь-який момент часу. Щоб мати логічний сигнал, тобто дискретний квантований, потрібно перетворити фізичний аналоговий сигнал на логічний. Таке перетворення виконують за допомогою спеціальних пристроїв — аналого-цифрових перетворювачів (АЦП).

Щоб перетворити електричний аналоговий сигнал, а саме електричну напругу, на логічний сигнал, весь діапазон зміни значень напруги поділяють на три зони: зона (діапазон) логічного нуля; зона (діапазон) логічної одиниці; зона невизначеності. У цифрових пристроях електрична напруга, як інформативний параметр аналогового сигналу, змінюється в межах від нуля до напруги живлення, яка в багатьох випадках дорівнює 5 V. Зону (діапазон) логічного нуля в такому разі встановлюють від нуля до 0,8 V (рис. 3.2), зону логічної одиниці — від 2 V до напруги живлення, тобто до 5 V, а зону невизначеності — від 0,8 V до 2 V.

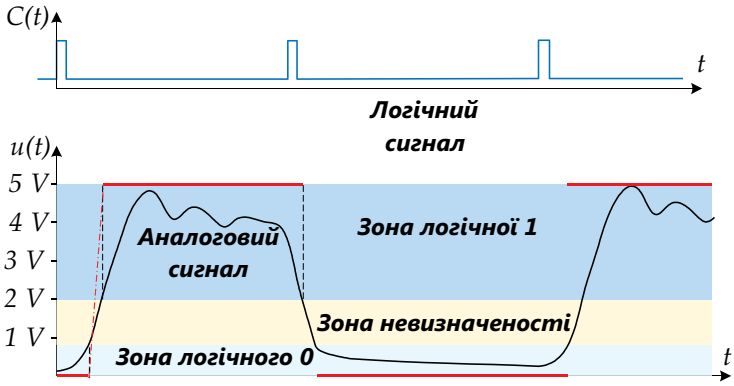


Рис. 3.2

3.3. ЛОГІЧНІ ЕЛЕМЕНТИ

Логічні елементи (англ. logic gate) — це елементарні цифрові пристрої, які виконують логічні операції зі вхідними логічними сигналами. За фізичною природою логічні елементи, як і логічні сигнали, можуть бути електричними, пневматичними, гідравлічними.

Сучасні цифрові пристрої побудовані, як правило, на електричних логічних елементах. Це зумовлено їхніми великою швидкодією, малим споживанням енергії, малими розмірами тощо. Електричні логічні елементи виготовляють із напівпровідникового матеріалу за сучасними інтегральними технологіями, які дають змогу на поверхні кристалів кремнію формувати польові й біполярні транзистори, резистори, конденсатори, з'єднувальні провідники у єдиному технологічному циклі. Створені за такими технологіями логічні пристрої називають *інтегральними мікросхемами*. На сучасному етапі розвитку всі логічні елементи і побудовані на їхній основі цифрові пристрої виготовляють у вигляді інтегральних мікросхем.

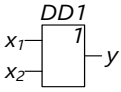
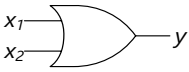
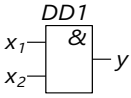
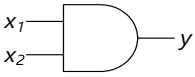
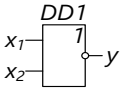
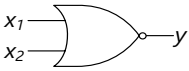
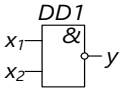
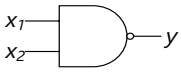
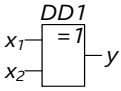
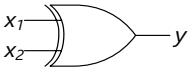
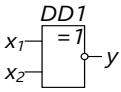
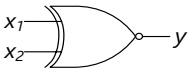
Головним елементом інтегральних мікросхем є транзистор, тому розрізняють логічні елементи, побудовані на *біполярних* і *польових* транзисторах.

За конструктивно-технологічними і схемо-технічними особливостями логічні елементи поділяють на такі види:

- *КМОН-логіка* — логічні елементи на комплементарних польових транзисторах з ізольованим затвором (англ. CMOS — complementary metal-oxide-semiconductor);
- *ЕЗЛ-логіка* — логічні елементи на біполярних транзисторах зі зв'язаними емітерами (англ. ECL — emitter coupled logic);
- *транзисторно-транзисторна (ТТЛ) логіка* — логічні елементи на біполярних транзисторах (англ. TTL — transistor-transistor logic);
- *ТТЛШ-логіка* — транзисторно-транзисторна логіка із транзисторами Шоткі (англ. Schottky TTL).

Для позначення логічних елементів на схемах електронних пристроїв згідно з державними стандартами застосовують умовні графічні зображення логічних елементів у вигляді прямокутника із символом логічної операції у верхньому правому куті. У англомовній технічній літературі (стандарт IEEE/ANSI) використовують дещо інші умовні графічні зображення логічних елементів. Назви найпоширеніших логічних елементів, їхні умовні графічні позначення і таблиці істинності для операцій, що реалізують ці елементи, подано в табл. 3.2.

Таблиця 3.2

Назва логічного елементу	Умовне позначення		Таблиця істинності
	ДСТУ	IEEE/ANSI	
НЕ (NOT)			x_1 | $y=\overline{x}$ 0 | 1 1 | 0
АБО (OR)			x_1 | x_2 | $y=x_1 \vee x_2$ 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1
І (AND)			x_1 | x_2 | $y=x_1 \cdot x_2$ 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1
АБО-НЕ (NOR)			x_1 | x_2 | $y=\overline{x_1 \vee x_2}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 0
І-НЕ (NAND)			x_1 | x_2 | $y=\overline{x_1 \cdot x_2}$ 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0
Виключне АБО (XOR)			x_1 | x_2 | $y=x_1 \oplus x_2$ 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 0
Виключне АБО-НЕ (XNOR)			x_1 | x_2 | $y=\overline{x_1 \oplus x_2}$ 0 | 0 | 1 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1

3.4. ПОНЯТТЯ ПРО ЧИСЛА Й СИСТЕМИ ЧИСЛЕННЯ

Числова інформація

Числова інформація є однією з найважливіших форм інформації.

Нагадаємо, що комп'ютер може обробляти інформацію лише у формі двійкових чисел, тобто в числовій формі. Спершу комп'ютер називали електронно-обчислювальною машиною (ЕОМ), тобто машиною для оброблення чисел. Інші форми інформації (текстову, графічну, відеоінформацію тощо) можна перетворити в цифрову форму.

Число — абстрактний об'єкт, який відображає кількісні характеристики матеріальних об'єктів. Потреба в числах виникла в людини дуже давно для визначення кількості предметів під час *лічби*. З розвитком суспільства поняття числа ускладнювалося. Сучасна математика використовує різні види чисел.

Числа один, два, три і так далі, які застосовують під час лічби для визначення кількості предметів, називають **натуральними числами**. Множину всіх натуральних чисел позначають символом **N** (або $\mathbb{N}$). Додатні й від'ємні числа, а також число нуль у сукупності — **цілі числа.** Множину всіх цілих чисел позначають символом Z (або $\mathbb{Z}$).

Числа, які можна подати як *відношення* двох цілих чисел, називають **раціональними** (англ. rational numbers). Множину всіх раціональних чисел позначають символом **Q** (або $\mathbb{Q}$).

Із раціональними тісно пов'язані **дробові числа** (англ. fractional numbers). Розрізняють *правильні дроби*, у яких чисельник менший за знаменник, і *змішані дроби*, у яких чисельник більший, ніж знаменник, і тоді це число складатиметься з *цілої* і *дробової частин*.

Числа, які не можна подати відношенням двох цілих чисел, називають **ірраціональними** (англ. irrational numbers). Прикладом ірраціональних чисел є число π=3,1415..., квадратний корінь із числа два $\sqrt{2}$=1,41... та ін. Раціональні та ірраціональні числа в сукупності становлять множину **дійсних чисел** (англ. real numbers). Множину всіх дійсних чисел позначають символом **R** (або $\mathbb{R}$).

Подальшим узагальненням чисел є **уявні числа** (англ. imaginary numbers), що ґрунтуються на понятті квадратного кореня з від'ємного числа, зокрема *уявної одиниці*. Уявна одиниця, яку позначають буквою i (або j), дорівнює $\sqrt{-1}$, тобто $i = j = \sqrt{-1}$.

Комплексні числа (англ. complex numbers) мають дві частини: *дійсну* і *уявну*. Множину всіх комплексних чисел позначають символом **C** (або $\mathbb{C}$).

Із розвитком писемності для подання числової інформації почали застосовувати спеціальні символи (знаки) — **цифри**. У повсякденному житті ми користуємося символами 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, які називають *арабськими цифрами*, оскільки до Європи вони прийшли від арабів, які, своєю чергою, їх запозичили в Давній Індії.

Системи числення

Система числення — це спосіб найменування і запису чисел за допомогою певного набору символів — *цифр*. Системи числення поділяють на *позиційні*, у яких число залежить від позицій, у яких стоять цифри, і *непозиційні*, де залежність від позицій не враховують. Непозиційні системи числення застосовували на початку розвитку людського суспільства, а на сучасному етапі їх майже повністю витіснили позиційні системи числення.

Прикладом *непозиційної* є римська система (римські цифри), яку сьогодні вживають, зокрема, для нумерації розділів книжок, для позначення годин на годинниковому циферблаті: I — для числа 1, V — 5, X — 10; L — 50; C — 100; D — 500; M — для числа 1000. Для позначення інших чисел застосовують комбінацію символів, наприклад для числа 8 — VIII, 40 — XL, 93 — XCIII тощо.

Найбільшого поширення в наш час набули *позиційні* системи числення, зокрема десяткова, яку використовують як у повсякденному житті, так і в науці. У такій системі число залежить не тільки від символів, якими його позначають, а й від місця, тобто *позиції*, на якій розташований той чи той символ.

Одним з основних понять позиційних систем є *основа*, або *база* (англ. radix, base), системи. Основою системи числення є сукупність символів, для зображення перших чисел натурального ряду (включно із числом нуль).

Залежно від кількості натуральних чисел, що входять до основи, позиційні системи числення можуть бути *десятковими* (англ. decimaly), *двійковими* (англ. binary), *вісімковими* (англ. octaly), *шістнадцятковими* (англ. hexadecimaly). Для позначення належності числа до певної системи числення використовують символ d для десяткової системи, b — для двійкової, o — для вісімкової, h — для шістнадцяткової, які записують наприкінці числа. Наприклад, 957.38d — десяткове число, 11001.101b — двійкове, 756.34o — вісімкове, 0c1b8.df9h — шістнадцяткове.

Десяткова система числення, якою ми повсякденно користуємося, для позначення перших *десяти* натуральних чисел використовує символи 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Двійкова — лише цифри 0 і 1. Вісімкова система числення використовує вісім цифр — 0, 1, 2, 3, 4, 5, 6, 7. Шістнадцяткова — десять арабських цифр і перші шість букв латиниці, тобто 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f.

Числа, які стоять у певних позиціях, називають *розрядами* позиційної системи. Кількість розрядів числа визначає його *розрядність*, тож числа у позиційних системах можуть бути однорозрядними, дворозрядними, n-розрядними.

У позиційних системах з основою b числа виражають формулою (3.1):

$$a_n \cdot b^n + a_{n-1} \cdot b^{n-1} + \ldots + a_1 \cdot b^1 + a_0 \cdot b^0 \cdot c_{-1} \cdot b^{-1} + c_{-1} \cdot b^{-1} + \ldots = \sum_{k=0}^{n} a_k \cdot b^k + \sum_{k=1}^{\infty} c_k \cdot b^{-k}, \quad 3.1$$

де b — основа системи числення;

a, c — натуральні числа, що входять до основи системи числення;
$a_k \cdot b^k$ — розряд цілої частини позиційної системи числення;
$c_k \cdot b^{-k}$ — розряд дробової частини позиційної системи числення;
b^{-k}, b^k — вага розряду;
k — показник степеня, ціле додатне число;
n — кількість розрядів цілої частини позиційної системи числення.

Записують числа в позиційній системі числення згідно з формулою (3.2):

$$a_n a_{n-1} a_{n-2} \cdots a_1 a_0 \,.\, c_1 c_2 c_3 \cdots \qquad 3.2$$

Для розділення цілої і дробової частин у США й Великій Британії використовують символ «точка» (.), а в Україні — символ «кома» (,).

Наприклад, десяткове число 3475.829 за формулою (3.2) можна подати так:

$$\begin{aligned} 3475.82 &= 3\cdot 10^3 + 4\cdot 10^2 + 7\cdot 10^1 + 5\cdot 10^0 + 8\cdot 10^{-1} + 2\cdot 10^{-2} = \\ &= 3\cdot 1000 + 4\cdot 100 + 7\cdot 10 + 5\cdot 1 + 8\cdot 0.1 + 2\cdot 0.01 = \\ &= 3000 + 400 + 70 + 5 + 0.8 + 0.02\,. \end{aligned} \qquad 3.3$$

Важливо зазначити, що основа в будь-якій позиційній системі числення має однаковий вираз — 10.

Переведення чисел з однієї системи числення в іншу

Оскільки система числення — це спосіб найменування і запису чисел, то саме число залишається тим самим у будь-якій системі числення. Отже, можна переводити числа із однієї системи числення в іншу, наприклад скориставшись формулою (3.1). Для цього потрібно виразити всі числа, що входять до формули (3.1), у новій системі числення і виконати всі дії, зазначені у формулі. Таким способом зручно переводити числа в десяткову систему числення, оскільки арифметичні дії звично робити в цій системі.

Переведення десяткових чисел у інші системи числення

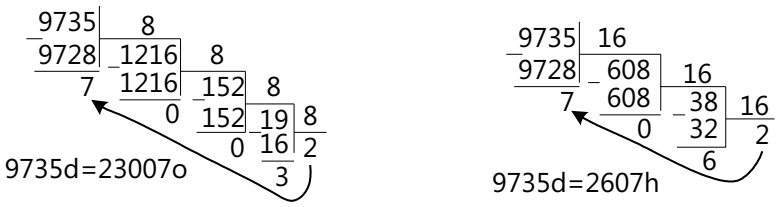
Десяткові числа в інші системи числення переводять окремо цілу частину і окремо дробову частину.

Переведення цілих десяткових чисел. Алгоритм переведення цілої частини десяткового числа в інші системи числення передбачає послідовне ділення на основу нової системи числення і фіксацію залишків ділення. Останній залишок буде найстаршим розрядом числа в новій системі, передостанній — наступним і так далі. Розглянемо переведення десяткових чисел на прикладах.

Приклад 1. Перевести ціле десяткове число 9735d у вісімкове.

Розв'язок

Послідовно ділимо число 9735d, а потім частки від ділення на основу нової системи числення, тобто на 8. Якщо записати залишки від послідовного ділення у зворотній послідовності (рис. 3.3, *а*), тобто найстаршим розрядом записати останній залишок, наступним розрядом — передостанній і нарешті наймолодшим — перший залишок, то отримаємо задане ціле число 23007o у новій, тобто вісімковій системі числення.



а *б*

Рис. 3.3

Переведення десяткових чисел у шістнадцяткову систему здійснюють за таким самим алгоритмом.

Приклад 2. Перевести ціле десяткове число 9735d у шістнадцяткове.

Розв'язок

Послідовним діленням на 16 (рис. 3.3, *б*), записуючи залишки у зворотній послідовності, отримаємо задане число в шістнадцятковій формі 2607h.

Приклад 3. Перевести ціле десяткове число 9735d у двійкове.

Розв'язок

Двійкове число отримуємо за таким самим алгоритмом, як і вісімкове й шістнадцяткове (рис. 3.4).

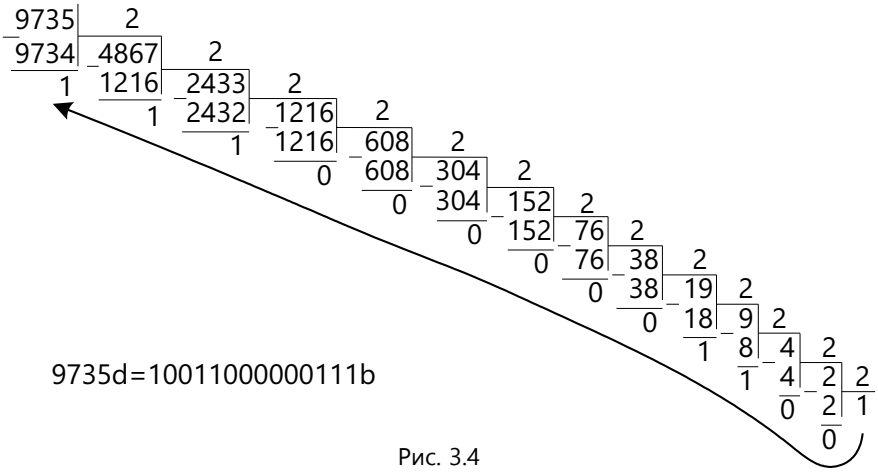
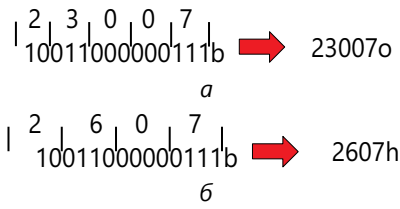


Рис. 3.4

Переведення двійкових чисел у вісімкові й шістнадцяткові, а також обернені переведення здійснювати дуже просто, оскільки 8 є третя степінь 2 (2^3 = 8), а 16 — четверта степінь (2^4 = 16). Щоб перевести двійкове число у вісімкове, потрібно його розряди поділити на трійки (тріади), починаючи з наймолодшого, а потім кожну трійку подати одним розрядом вісімкового числа (рис. 3.5). Переведення двійкового числа в шістнадцяткове здійснюють аналогічно, тільки розряди двійкового числа ділять не на трійки, а на четвірки (тетради).



а

б

Рис. 3.5

Вісімкове або шістнадцяткове число у двійкове переводять порозрядно, тобто кожен розряд заміщують його двійковим еквівалентом: вісімковий розряд — трійкою двійкових розрядів (рис. 3.6, *а*), а шістнадцятковий — четвіркою двійкових розрядів (рис. 3.6, *б*).

3 6 5 7 2
011 110 101 111 010

36572o=011110101111010b

а

6 A 9 D
0110 1010 1001 1101

6A9Dh=0110101010011101b

б

Рис. 3.6

Переведення дробових десяткових чисел. Дробові двійкові числа в інші системи числення переводять послідовним множенням заданого числа і дробової частини отриманих добутків на основу нової системи числення. Розрядами дробового числа в новій системі числення будуть цілі частини послідовних добутків. У новій системі числення задане дробове число може мати нескінченну кількість розрядів, тому при переведенні кількість розрядів обмежують для досягнення заданої точності.

Приклад 4. Перевести дробове десяткове число 0.33496d у вісімкове.

Розв'язок

Множимо задане дробове десяткове число на основу нової системи числення, тобто на 8. Дробову частину отриманого добутку знову множимо на 8 і далі повторюємо множення доти, доки не досягнемо заданої точності (рис. 3.7, *а*).

Приклад 5. Перевести дробове десяткове число 0.80664d у шістнадцяткове.

Розв'язок

Виконуємо множення заданого числа і дробових частин отриманих добутків на основу нової системи числення — 16. Цілі частини добутків перетворюємо з десяткової системи в шістнадцяткову і записуємо результат у шістнадцятковій системі числення (рис. 3.7, *б*).

0.33496
×8
2.67968
×8
5.43744
×8
3.49952
×8
3.99616
≈4.00000

0.80664
×16
12.90624
×16
14.49984
×16
7.99744
≈8.00000

0.59375
×2
1.18750
×2
0.37500
×2
0.75000
×2
1.50000
×2
1.00000
×2
0.00000

0.33496d → 0.2534o 0.80644d → 0.CD8h 0.59375d → 0.100110b

а *б* *в*

Рис. 3.7

Приклад 6. Перевести дробове десяткове число 0.59375d у двійкове.

Розв'язок

Спочатку множимо задане десяткове число на 2, а потім послідовно крок за кроком множимо дробову частину отримуваних добутків на 2. Дробове двійкове число маємо як послідовність цілих частин добутків (рис. 3.7, *в*).

3.5. ЧИСЛА З ФІКСОВАНОЮ ТОЧКОЮ І АРИФМЕТИЧНІ ДІЇ З НИМИ

Числову інформацію в комп'ютері подано у вигляді багаторозрядних двійкових чисел, які зберігаються та обробляються у двох формах: із *фіксованою точкою* (або *комою*) і з *рухомою точкою* (або *комою*). У двійкових числах із фіксованою точкою положення точки, яка поділяє число на *цілу* і дробову частини, фіксоване і не змінюється у процесі оброблення інформації, а в числах із рухомою точкою її положення змінюється («плаває») залежно від діапазону чисел, що обробляються.

Для чисел із фіксованою точкою застосовують здебільшого закріплення положення точки після наймолодшого розряду, тобто двійкові числа представляють *цілі числа*. На початкових етапах розвитку комп'ютерної техніки застосовували двійкові числа з фіксацією точки перед найстаршим розрядом, тобто подавалися дробовими числами, але зараз такий метод майже не використовують.

Двійкові цілі числа в комп'ютері можуть бути подані як *цілі числа без знака* (англ. unsigned integers), тобто *модулі* (англ. magnitude) чисел, і цілі *числа зі знаком* (англ. signed integers), тобто додатні й від'ємні цілі числа.

Процеси й об'єкти навколишнього світу характеризуються фізичними величинами в надзвичайно широкому діапазоні значень. У комп'ютері двійкові числа можуть мати скінченну кількість розрядів, тобто обмежений діапазон значень.

Найменшою одиницею числової інформації є *біт* (англ. BIT — BInary digiT) — розряд двійкового числа. Операції з окремими бітами використовують під час оброблення логічної інформації.

Наступна за розміром одиниця числової інформації — *напівбайт*, або *тетрада, нібл* (англ. nibble — «недогризок»), що містить чотири двійкові розряди.

Найпоширенішою одиницею числової інформації є *байт* (англ. byte), що складається з восьми двійкових розрядів. За допомогою байта можна закодувати 256 станів, тобто 256 елементарних одиниць інформації. Перші мікропроцесори, що масово випускала промисловість, були 8-розрядні, тобто оперували з байтами інформації.

У наступних поколіннях мікропроцесорів розрядність зростала. 8-розрядні мікропроцесори змінили 16-розрядні, або 2-байтні, їх, своєю чергою, — 32-розрядні, або 4-байтні. Нині випускають здебільшого 64-розрядні, або

8-байтні мікропроцесори. У технічній літературі одиницю інформації, кратну байту, називають *словом* (англ. word), або *подвійним словом* (англ. double word).

Двійкові числа зі знаками. Прямий, обернений і доповнювальний код двійкових чисел

Для алгебраїчного подання двійкових чисел у комп'ютері, тобто для подання як додатних, так і від'ємних чисел, застосовують прямий, обернений і доповнювальний коди двійкових чисел, у яких знак числа плюс (+) кодують нулем (0), а мінус (−) — одиницею (1). Для знака відводять спеціальний розряд двійкового числа — *знаковий*, що розміщений зазвичай на початку числа, тобто на позиції *найстаршого розряду* (англ. MSB — most significant bit). Решту розрядів відводять для подання модуля числа (рис. 3.8).

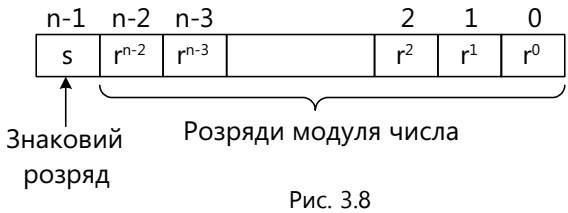


Рис. 3.8

Прямий код (англ. signed magnitude representation) числа визначають виразом:

$$A_d = \begin{cases} 0|A| & A \geq 0; \\ 1|A| & A \leq 0. \end{cases} \qquad 3.4$$

Прямі коди додатних і від'ємних чисел відрізняються тільки знаковими розрядами. На рис. 3.9 показано прямі коди 3-розрядних додатних і від'ємних двійкових чисел. Нуль у прямому коді має два зображення: додатний нуль «000000....» та від'ємний нуль «10000...».

Обернений код (англ. one's complement, або 1's complement). Обернений код n-розрядного числа D з основою r визначають за формулою

$$D_{r-1} = \left(r^n - 1\right) - D. \qquad 3.5$$

Наприклад, для 4-розрядного десяткового числа D=9264 обернений код дорівнює D_{r-1}=(10^4 – 1)−9264= =(10000−1)−9264=9999−9264=0735. Отже, кожен розряд оберненого коду є число, яке доповнює відповідний розряд прямого коду до 9.

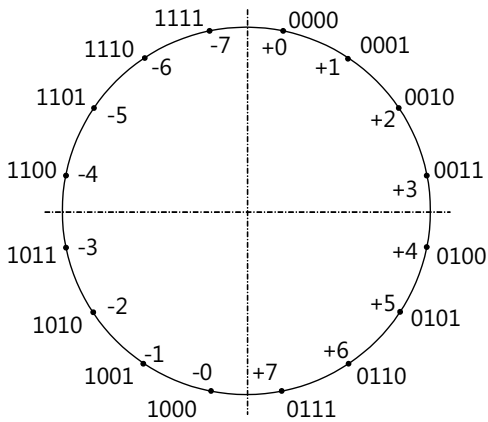


Рис. 3.9

Обернений код додатного числа такий самий, як і прямий код цього числа. Для від'ємного двійкового 8-розрядного числа D = 10111011 обернений код дорівнює D_{r-1} = (28 – 1) – 10111011 = (100000000 – 1) – 10111011 = = 11111111 – 10111011 = 01000100. Обернений код від'ємного двійкового числа можна отримати, якщо всі розряди прямого коду *інвертувати*, тобто нулі замінити одиницями, а одиниці — нулями, і знаковий розряд також. На рис. 3.10 наведено обернені коди 3-розрядних додатних і від'ємних двійкових чисел. Нуль в оберненому коді має два зображення: додатний нуль «000000...» і від'ємний нуль «11111111...».

Доповнювальний код (англ. two's complement, або 2's complement) n-розрядного числа D з основою r визначають за формулою:

$$D_{r-1} = \left(r^n\right) - D. \qquad 3.6$$

Наприклад, для 4-розрядного десяткового числа D = 9264 обернений код дорівнює D_{r-1} = (10^4) – 9264 = 10000 – 9264 = 0736. Отже, кожен розряд оберненого коду є числом, яке доповнює відповідний розряд прямого коду до 9.

Доповнювальний код додатного двійкового числа такий самий, як і прямий код цього числа. Для від'ємного двійкового 8-розрядного числа D = 10111011 доповнювальний код дорівнює

D_{r-1}=2^8 –10111011=100000000–10111011=01000101.

Доповнювальний код від'ємного двійкового числа можна отримати, додавши до його оберненого коду одиницю наймолодшого розряду.

На рис. 3.11 показано доповнювальні коди 3-розрядних додатних і від'ємних двійкових чисел. Нуль у доповнювальному коді має лише одне зображення «000...». Кількість від'ємних чисел на одиницю більша за кількість додатних чисел.

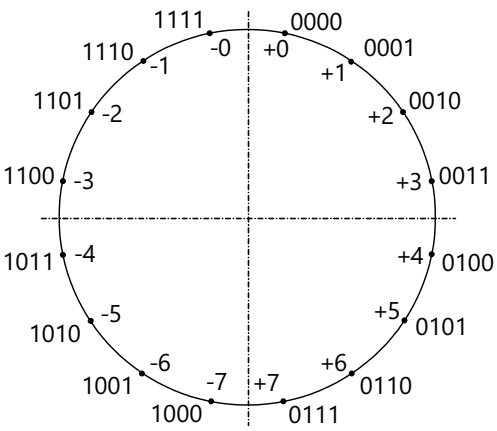


Рис. 3.10

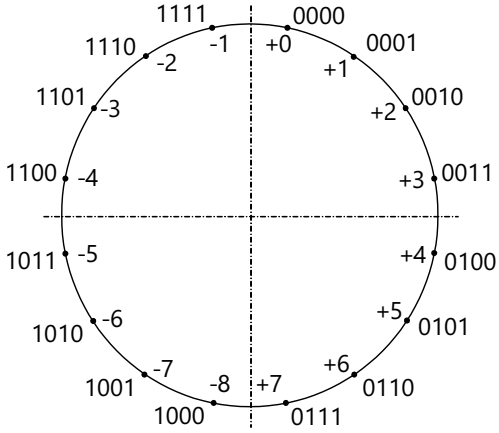


Рис. 3.11

3.6. ЧИСЛА З РУХОМОЮ ТОЧКОЮ І АРИФМЕТИЧНІ ДІЇ З НИМИ

Фізичні величини, що характеризують навколишній світ, охоплюють надзвичайно широкий діапазон: від астрономічних до величин мікросвіту. Для їх подання широко використовують *наукову форму запису* (англ. scientific notation) дійсних чисел:

$$A = M \cdot b^E, \qquad 3.7$$

де A — дійсне число;

M — мантиса (англ. significand or mantissa);

b — основа (база) системи числення;

E — показник степеня, або порядок (англ. exponent).

Користуючись цією формою запису, масу електрона записують як m_e = 9.1093822 · 10^{-31} кг, а масу Землі — як M = 5.9736 · 10^{24} кг. Така форма запису є компактною, і тому немає потреби записувати тридцять нулів після десяткової точки для маси електрона, або більш ніж двадцять розрядів для маси Землі.

Те саме число можна записати по-різному, наприклад число π:

3.1415 = 31.415 · 10^{-1} = 314.15 · 10^{-2} = 3141.5 · 10^{-3} = 0.31415 · 10^{+1} = 0.031415 · 10^{+2}. Здебільшого мантису вибирають у діапазоні 10 ≥ M > 1, таку форму запису називають *нормалізованою*.

Запис двійкового числа у формі (3.7) дістав назву *чисел із рухомою точкою* (англ. floating point numbers). Кількість двійкових розрядів, відведених під мантису, визначає *точність* подання числа, а кількість розрядів, відведених під показник степеня, — *діапазон* подання чисел. Отже, розподілом кількості розрядів розрядної сітки комп'ютера між мантисою і порядком узгоджуються суперечливі вимоги між точністю і діапазоном подання чисел. Щоб забезпечити сумісність комп'ютерів різних видів, потрібно стандартизувати подання двійкових чисел із рухомою точкою. Із цією метою було розроблено міжнародний стандарт IEEE 754, якого дотримуються провідні світові виробники комп'ютерів. Згідно із цим стандартом двійкові числа з рухомою точкою можуть бути подані у форматі зі звичайною (англ. single precision) і подвійною (англ. double precision) точністю. У форматі зі звичайною точністю двійкове число має 32 розряди, з них на знак відводиться один (найстарший) розряд, на показник степеня — 8 і на мантису — 23 розряди (рис. 3.12, *а*). Двійкове число подвійної точності має 64 розряди, з них один розряд відводиться на знак, 11 — на показник степеня і 52 — на мантису (рис. 3.12, *б*). Стандарт IEEE 754 передбачає використання нормалізованої мантиси, тобто манти-

Рис. 3.12

си в діапазоні $2 > M > 1$. Така нормалізована мантиса завжди має 1 у найстаршому розряді 2^0, що дає змогу не зберігати цей розряд у пам'яті. Такий «*схований*» (англ. hidden) розряд забезпечує точність подання мантиси у 24 розряди, тимчасом як зберігаються лише 23 розряди. У процесі виконання операцій схований розряд автоматично додається до мантиси. Це стосується також чисел із подвійною точністю.

Подання показників степені також має свої особливості. Стандарт передбачає подання показників степені у вигляді *зміщеного порядку*, або *порядку з надлишком* 127. 8 розрядів, відведених у форматі зі звичайною точністю під порядок числа, дають змогу виразити 254 показники степеня у діапазоні від -127_{10} до $+127_{10}$. Зміщений порядок отримують, додаючи число $127_{10} = 01111111_2$ до показника степені, який може бути як додатним, так і від'ємним. Зміщений порядок є додатним числом, що значно спрощує операції з двійковими числами з рухомою точкою.

32-розрядні двійкові числа звичайної точності згідно зі стандартом IEEE 754 розміщують на числовій осі у двох діапазонах (рис. 3.13). Від'ємні числа лежать у діапазоні від мінус $(2 - 2^{-23}) \cdot 2^{+127}$ до мінус $1 \cdot 2^{-127}$, а додатні — від плюс $1 \cdot 2^{-127}$ до плюс $(2 - 2^{-23}) \cdot 2^{+127}$.

Якщо при виконанні арифметичних операцій їх результат потрапить у діапазон чисел від мінус $1 \cdot 2^{-127}$ до нуля, то це означає *від'ємну втрату значущості*. *Додатна втрата значущості* має місце, якщо додатний результат операції буде у діапазоні від нуля до плюс $1 \cdot 2^{-127}$. Числа, що лежать у діапазоні додатної і від'ємної втрати значущості, близькі до нуля, і їх можна замінити нулем. Однак така заміна може у подальшому зумовити вимкнення і переривання, якщо, наприклад, потрібно буде виконати операцію ділення на це число. У разі втрати значущості стандарт IEEE 754 передбачає спеціальний формат чисел, які називають *ненормалізованими* (англ. denormalized numbers, або subnormal numbers). Найбільше ненормалізоване число містить усі нулі в розрядах порядку, нуль у схованому розряді й одиниці у 23 розрядах мантиси. Це число майже не відрізняється від найменшого нормалізованого числа $1 \cdot 2^{-127}$. Застосування ненормалізованих чисел призводить до *поступової* втрати значущості під час обчислень, на відміну від *стрибкоподібної* втрати значущості при застосуванні нуля в цьому діапазоні.

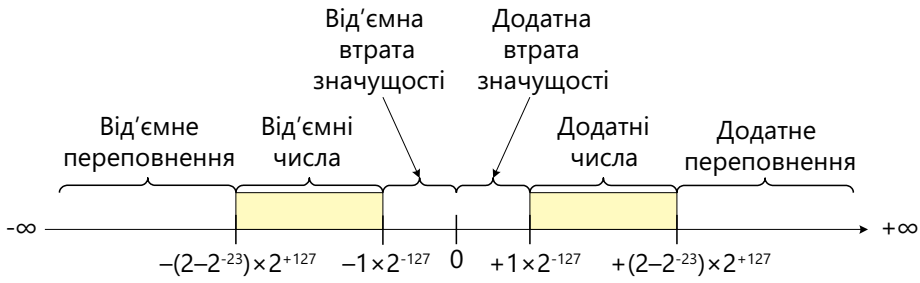


Рис. 3.13

У стандарті IEEE 754 передбачено використання додатного і від'ємного нулів, які відрізняються лише знаковими розрядами і містять нулі у всіх розрядах порядку і мантиси, включно зі схованим розрядом.

Від'ємне переповнення виникає, якщо результат арифметичних операцій буде менший за найменше від'ємне нормалізоване число мінус $(2 - 2^{-23}) \cdot 2^{+127}$, а *додатне переповнення* — якщо цей результат буде більший за найбільше нормалізоване додатне число плюс $(2 - 2^{-23}) \cdot 2^{+127}$. На відміну від втрати значущості, у переповнення не можна перейти поступово. Для розв'язання проблеми переповнення у стандарті IEEE 754 передбачено використання спеціальних форматів чисел: *додатної* і *від'ємної нескінченності*, які відрізняються лише знаковими розрядами і містять 1 у всіх розрядах порядку і всі 0 в розрядах мантиси. З такими спеціальними числами, як додатна і від'ємна нескінченності, можна виконувати операції, як і зі звичайними числами. Наприклад, якщо до нескінченності додати нормалізоване число, отримаємо нескінченність; якщо будь-яке нормалізоване число поділити на нескінченність, то отримаємо нуль.

Якщо результат операції невизначений, наприклад $\pm 0 \cdot \pm\infty$ або $\pm 0 / \pm 0$, то в стандарті IEEE 754 передбачено використання спеціального формату — **НЕ ЧИСЛО** (англ. NaN — not a number). Спеціальна величина НЕ ЧИСЛО виникає у результаті арифметичної операції, якщо всі розряди порядку встановлені в 1, а мантиса має ненульове значення.

ТЕСТОВІ ЗАВДАННЯ

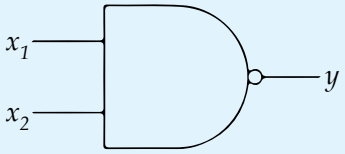
X1	*X2*	*Y=f(X1,X2)*
0	0	1
0	1	0
1	0	0
1	1	0

НАВЕДЕНА ВИЩЕ ТАБЛИЦЯ ІСТИННОСТІ — ЦЕ ТАБЛИЦЯ ІСТИННОСТІ ЛОГІЧНОЇ ФУНКЦІЇ:

1. диз'юнкції
2. кон'юнкції
3. заперечення диз'юнкції
4. заперечення кон'юнкції

ЛОГІЧНІ СИГНАЛИ – ЦЕ СИГНАЛИ, ЩО МАЮТЬ:

1. один рівень
2. два рівні
3. три рівні
4. чотири рівні



УМОВНЕ ГРАФІЧНЕ ЗОБРАЖЕННЯ, НАВЕДЕНЕ НА РИСУНКУ, — ЦЕ ЗОБРАЖЕННЯ ЛОГІЧНОГО ЕЛЕМЕНТА:

1. АБО (OR)
2. І (AND)
3. АБО-НЕ (NOR)
4. І-НЕ (NAND)

ДЕСЯТКОВЕ ЧИСЛО 8967d ДОРІВНЮЄ ТАКОМУ ВІСІМКОВОМУ ЧИСЛУ:

1. 21307o
2. 21407o
3. 21406o
4. 21309o

ШІСТНАДЦЯТКОВЕ ЧИСЛО F9C6h ДОРІВНЮЄ ТАКОМУ ДЕСЯТКОВОМУ ЧИСЛУ:

1. 63932d
2. 63923d
3. 63942d
4. 63924d

ПОНЯТТЯ ПРО АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРІВ

Апаратне забезпечення комп'ютерів (англ. computer hardware) — це сукупність фізичних пристроїв, що входять до складу комп'ютера.

Системний блок (англ. computer case) — це основна складова частина комп'ютера (рис. 4.1, *а*), містить блок живлення, материнську (системну, або основну) плату із центральним процесором і оперативною пам'яттю, різні накопичувачі (жорсткий диск, дисководи, приводи CD-ROM або DVD-ROM), плати розширення (графічну плату, звукову плату, мережну плату, модем), додаткові пристрої тощо.

Матери́нська пла́та (англ. motherboard) — плата друкованого монтажу (рис. 4.1, *б*), на якій змонтовано головні електронні пристрої комп'ютера, з'єднані між собою системою друкованих провідників. На материнській платі розміщуються: мікропроцесор; оперативна пам'ять; набір мікросхем керування, або чипсетів (англ. chipset); постійна пам'ять із системною BIOS (базовою системою уведення–виведення); слоти розширення; конектори для під'єднання інтерфейсних кабелів жорстких дисків, дисководів; конектори живлення; конектори послідовного (COM) й паралельного (LPT) портів, універсальної послідовної шини USB; конектор для підключення клавіатури й миші та низка інших компонентів. На материнській платі також можуть бути мікросхеми відеоадаптера, звукової плати й мережної карти.

Комп'ютерний блок живлення призначений для забезпечення пристроїв комп'ютера електричною енергією (рис. 4.1, *в*). Його завдання — перетворення змінної мережної напруги промислової частоти на постійну напругу, її стабілізація і захист від завад і наводок. У сучасних персональних комп'ютерах застосовують здебільшого імпульсні блоки живлення, що мають незначну масу і габарити й забезпечують захист від короткого замикання та перевантаження.

Периферійні пристрої (периферія) — це додаткові пристрої, які з'єднані з системним блоком кабелями.

Периферійні пристрої поділяють на такі види:

- вхідні пристрої;
- вихідні пристрої;
- пристрої пам'яті.

а

б

в

Рис. 4.1

Вхідні периферійні пристрої сприймають інформацію в різних формах із навколишнього світу, перетворюють її у двійкову форму і вводять у комп'ютер. Найпоширеніші вхідні периферійні пристрої: клавіатура, комп'ютерна миша, цифрова, або вебкамера, сенсорний екран, сканер, мікрофон тощо.

Вихідні периферійні пристрої перетворюють двійкову інформацію, оброблену комп'ютером, у форми інформації, що сприймаються людиною. До вихідних периферійних пристроїв належать: монітори, принтери, плотери, акустичні системи.

Основна пам'ять розміщена на материнській платі — на мікросхемах пам'яті, змонтованих на платах розширення. Крім основної, у комп'ютерах використовують додаткову, або вторинну, пам'ять великої ємності, розміщену на периферійних пристроях пам'яті. До таких пристроїв належать: магнітні диски (вінчестери), флешпам'ять, пам'ять на оптичних дисках і магнітних стрічках.

Розділ 4
ПРИСТРОЇ ВВЕДЕННЯ І ВИВЕДЕННЯ ІНФОРМАЦІЇ

Комп'ютери як технічні пристрої для оброблення інформації застосовують у різноманітних сферах діяльності — предметних галузях. Кожній предметній галузі відповідає специфічна форма інформації: в офісах переважає текстова і цифрова форми; у промисловості й технологічних лініях — інформація, отримана з датчиків фізичних величин; у медицині — інформація про стан людини, зібрана за результатами аналізу і з медичних діагностичних приладів. Усі ці види інформації пристрої введення сприймають і перетворюють у форму двійкових чисел.

Оброблену комп'ютером інформацію потрібно з двійкової форми перетворити у форми, звичні та зручні для сприйняття людиною. Таке перетворення виконують пристрої виведення інформації: дисплеї, принтери, плотери, звукові колонки тощо.

4.1. ПРИСТРОЇ ВВЕДЕННЯ ІНФОРМАЦІЇ

Оскільки комп'ютер обробляє інформацію лише у формі двійкових чисел, потрібні спеціальні пристрої, які переводять інформацію в числовій, текстовій, графічній, звуковій та інших формах у двійкову форму і вводять її в комп'ютер.

Розглянемо найпоширеніші на сучасному етапі розвитку комп'ютерної техніки пристрої введення інформації.

Клавіатура. За допомогою клавіатури в комп'ютер вводять символи, зокрема букви (літери) латиниці й кирилиці, розділові знаки, цифри, спеціальні символи. Стандартна клавіатура має від 101 до 104 клавіш, розміщених за стандартом QWERTY (перший рядок алфавітних клавіш починається з клавіш Q, W, E, R, T, Y). Клавіші клавіатури поділено на кілька функціональних груп: алфавітно-цифрові, функціональні, керування курсором, службові, клавіші додаткової панелі (рис. 4.2).

Алфавітно-цифрові клавіші (літери кирилиці й латиниці, цифри, спеціальні символи) використовують для введення текстової інформації і команд у текстовій формі. Клавіші працюють у двох режимах (регістрах): у нижньому регістрі — малі символи, у верхньому — великі. Перемикають регістри натисканням на клавішу *Shift* у момент введення поточного символу (нефіксоване перемикання) або за допомогою клавіші *Caps Lock* при введенні групи символів (фіксоване перемикання).

Рис. 4.2

Групу функціональних клавіш *F1* ... *F12* розміщено у верхній частині клавіатури. Функції цих клавіш залежать як від програми, яка в цей час виконується на комп'ютері, так і від операційної системи.

Клавіші керування курсором (← ↑ → ↓) подають команди на переміщення курсора на екрані. Курсор — це спеціальний значок на екрані монітора, який вказує на місце введення чергового символу. До цієї групи належать клавіші, які прокручують сторінку вгору — *Page Up*, або вниз — *Page Down*, а також клавіші, що переміщують курсор на початок рядка — *Home*, або в кінець рядка — *End*.

Група клавіш додаткової панелі дублює дію цифрових клавіш, клавіш керування курсором і деяких службових клавіш. Основне призначення додаткової панелі — введення значних обсягів цифрової інформації, тому клавіші на цій панелі розміщені в порядку, зручному для роботи.

Миша. Для керування комп'ютером, тобто введення в комп'ютер керівної інформації, широко застосовують маніпулятор — мишу, яка дістала цю назву за зовнішню схожість із однойменною твариною (рис. 4.3).

Рис. 4.3

Сканер. Сканер — пристрій для автоматизованого зчитування графічної інформації (рис. 4.4). Принцип дії сканера полягає в тому, що встановлене на рухомій каретці джерело світла опромінює графічний об'єкт — креслення, фотографію, картину. Відбитий від зображення світловий потік сприймається фотодатчиком і перетворюється у форму двійкових чисел. Джерело й приймач переміщуються над зображенням послідовно, рядок за рядком, тобто сканують зображення. Характеристикою якості сканера є кількість елементів зображення — точок на одиницю довжини дюйм (англ. dots per inch — dpi).

Рис. 4.4

Сенсорний екран. Сенсорний екран (англ. touchscreen) — пристрій введення інформації, який реагує на дотик (рис. 4.5). За принципом дії розрізняють резистивні, поверхнево-акустичні, ємнісні, інфрачервоні сенсорні екрани.

Рис. 4.5

Під час дотику змінюються локальні характеристики екрана. Координати точки дотику фіксуються і передаються на оброблення спеціальній програмі.

Сенсорні екрани широко застосовують у мобільних телефонах, кишенькових комп'ютерах, терміналах з оплати різних послуг, інформаційних системах вокзалів тощо.

Вебкамера. Сучасна вебкамера — цифровий пристрій, який знімає зображення, перетворює їх у цифрову форму, стискає інформацію і передає її комп'ютерною мережею (рис. 4.6). Вебкамера складається з оптичної системи, матриці світлочутливих елементів і спеціального програмного забезпечення.

Рис. 4.6

Вебкамери застосовують для проведення відеоконференцій, в охоронних системах, у відеотелефонії, для моніторингу ситуації в офісах і торговельних залах, для спілкування через Інтернет та в багатьох інших галузях.

4.2. ПРИСТРОЇ ВИВЕДЕННЯ ІНФОРМАЦІЇ

Людина сприймає текстову, цифрову, графічну й відеоінформацію тощо через свої органи чуттів у різних формах. Комп'ютер, як уже зазначено, сприймає і обробляє інформацію у формі двійкових чисел, тому відповідні пристрої перетворюють внутрішнє подання інформації у форми, що сприймаються людиною.

Розглянемо найпоширеніші пристрої виведення інформації.

Відеоінформація (від лат. video — «дивлюся, бачу») — один із найважливіших видів інформації, який людина сприймає за допомогою органів зору — очима. Порівняно зі звуковою і текстовою інформацією, відтворення відеоінформації потребує значно більших апаратних і програмних ресурсів.

Відеокарта (графічна карта, графічний адаптер, графічний прискорювач, англ. video card) — електронний цифровий пристрій, призначений для перетворення двійкової інформації на аналоговий відеосигнал. Відеокарту виготовляють у вигляді окремої плати розширення, яку вставляють у конектор материнської плати (рис. 4.7) або вбудовують (інтегрують) у материнську плату.

Рис. 4.7

У деяких комп'ютерах, наприклад призначених для комп'ютерних ігор, для оброблення відеоінформації потрібна велика продуктивність цен-

трального процесора. Для його розвантаження оброблення відеоінформації покладають на спеціальний процесор — *процесорний пристрій прискорення* (англ. accelerated processing units, APU).

Зі зростанням процесорної потужності сучасних відеокарт збільшується споживання ними електричної енергії. Відеокарти сучасних комп'ютерів є одними з основних споживачів електричної енергії, тож відведення тепла є одним з основних завдань під час проєктування відеокарт.

Сучасні відеокарти складаються з *плати друкованого монтажу* (англ. printed circuit board), на якій монтують такі електронні компоненти:

- *графічний процесорний пристрій* (англ. graphics processing unit, GPU) з великою продуктивністю, призначений для складного оброблення відеоінформації;
- *пристрій для відведення тепла* (англ. heat sink), призначений для запобігання перегріванню відеокарти, що може призвести до її псування. Застосовують як пасивні розсіювачі тепла — радіатори, так і активні — мініатюрні вентилятори — кулери (англ. cooler). Для потужних комп'ютерів іноді використовують водяне охолодження;
- *відеопам'ять* (англ. video memory) достатньо великого обсягу (від 128 мегабайтів до 8 гігабайтів) із великою швидкодією (частота синхронізації від 1 гігагерца до 6,3 гігагерца), призначена для роботи з графічним процесором;
- *цифро-аналоговий перетворювач*, призначений для перетворення цифрового сигналу на аналоговий і подавання його на вхід пристрою відображення. Треба зазначити, що сучасні пристрої відображення мають цифровий вхід і не потребують цифро-аналогових перетворювачів.

Пристрої відображення інформації. Для відображення відеоінформації (тексту, креслень, фотографій, картин, фільмів, анімацій тощо) використовують дисплей або монітор. *Дисплей* (англ. display — «відображувати») має ширше застосування (наприклад, дисплей мобільного телефона), *монітор* (англ. monitor — «слідкувати») пов'язаний здебільшого з комп'ютером або телеекраном дистанційного спостереження.

Сучасні комп'ютерні монітори бувають кількох типів:

- на основі електронно-променевої трубки (CRT);
- рідкокристалічні (LCD, TFT як підвид LCD);
- плазмові;
- проєкційні;
- OLED-монітори;
- віртуальні ретинальні.

У перших комп'ютерах використовували *монітор на основі електронно-променевої трубки* (англ. cathode ray tubes, CRT) (рис. 4.8, *а*). Спочатку ці пристрої були монохромними, у 1970-ті з'явилися кольорові зразки. Монітори на основі електронно-променевої трубки мають велику швидкодію, високу роздільну здатність. Недоліками є великі геометричні розміри (особливо углиб) і маса, тому такі монітори поступово витісняються.

а *б*

Рис. 4.8

Принцип дії *рідкокристалічного дисплея* (англ. liquid crystal display, LCD) (рис. 4.8, *б*) ґрунтується на властивості рідких кристалів змінювати свою просторову орієнтацію під дією електричного поля. Оскільки рідкі кристали оптично анізотропні, то під дією електричного поля змінюються також їхні оптичні властивості. Для поліпшення якості зображення в рідкокристалічних дисплеях почали застосовувати *тонкоплівкові транзистори* (англ. thin-film-transistor, TFT). Такий різновид має назву TFT LCD дисплей.

Перевагою рідкокристалічних дисплеїв є незначне енергоспоживання, тому їх з успіхом застосовують у пристроях на батарейках і акумуляторах. Крім того, такі пристрої можна виготовити плоскими й будь-яких розмірів.

Плазмовий (газорозрядний) дисплей, або *плазмова панель* (англ. plasma display panel, PDP), — пристрій відображення інформації, дія якого базується на явищі світіння люмінофору під впливом ультрафіолетових променів, що виникають за електричного розряду в іонізованому газі, тобто в плазмі. Плазмові панелі забезпечують високу якість зображення (контрастність, глибина кольорів) на великій площі (до кількох метрів у діагоналі). Недоліками плазмових панелей є значне енергоспоживання, поступове вигорання екрана, залежність якості зображення від висоти над рівнем моря.

Проєкційний монітор — світловий прилад, що перерозподіляє світло лампи з концентрацією світлового потоку на поверхні малого розміру або в малому обсязі. Проєктори є в основному оптико-механічними або оптично-цифровими приладами, що дають змогу за допомогою джерела світла проєктувати зображення об'єктів на поверхню, розташовану поза приладом, — екран.

OLED-монітор ґрунтується на технології OLED (англ. organic light-emitting diode). Органічний світлодіод — напівпровідниковий прилад, виготовлений з органічних сполук, які ефективно випромінюють світло під час пропускання крізь них електричного струму. Випромінювальний електролюмінісцентний шар зазвичай містить полімерні речовини, які дають змогу органічним складовим бути як слід депонованими. Вони розташовуються в так званих рядках та стовпчиках за площею підкладки простим процесом «друку». У результаті отримуємо матрицю з пікселів, які випромінюють світіння різних кольорів. Передбачають, що виробництво таких дисплеїв буде набагато дешевшим, ніж виробництво рідкокристалічних дисплеїв.

Віртуальний ретинальний монітор (англ. virtual retinal display, VRD; retinal scan display, RSD) — технологія пристроїв виведення, що формує зображення безпосередньо на сітківці ока. В результаті користувач бачить зображення, які «висять» у повітрі перед ним. Пристрої такого типу наближені до систем до-

повненої реальності, оскільки зображення віртуальних об'єктів, які бачить користувач, накладаються на зображення об'єктів реального світу.

У пристроях відображення інформації використовують такі типи *відеоадаптерів*:

- MDA (monochrome display adapter — монохромний адаптер дисплея) — найпростіший відеоадаптер, був у перших IBM PC. Працює в текстовому режимі з роздільною здатністю 80×25 (720×350, матриця символу — 9×14), підтримує п'ять атрибутів тексту: звичайний, яскравий, інверсний, підкреслений і миготливий. Частота рядкової розгортки — 15 Кгц. Інтерфейс із монітором — цифровий: сигнали синхронізації, основний відеосигнал, додатковий сигнал яскравості;
- HGC (Hercules graphics card — графічна карта Hercules) — розширення MDA із графічним режимом 720×348, розроблене фірмою Hercules;
- CGA (color graphics adapter — кольоровий графічний адаптер) — перший адаптер із графічними можливостями, який працює в текстовому режимі з роздільною здатністю 40×25 і 80×25 (матриця символу — 8×8), у графічному — з роздільною здатністю 320×200 або 640×200;
- EGA (enhanced graphics adapter — покращений графічний адаптер) — подальший розвиток CGA, застосований у перших PC AT. Додано роздільну здатність 640×350, що в текстових режимах дає формат 80×25 за матриці символу 8×14 і 80×43 — за матриці 8×8. Кількість одночасно відображуваних кольорів — 16, палітру розширено до 64 кольорів (по два розряди яскравості на кожен колір);
- MCGA (multicolor graphics adapter — багатобарвний графічний адаптер) — уведений фірмою IBM у ранніх моделях PS/2;
- VGA (video graphics array — масив візуальної графіки) — розширення MCGA сумісний з EGA, введений фірмою IBM у середніх моделях PS/2;
- IBM 8514/a — спеціалізований адаптер для роботи з високою роздільною здатністю (640×480×256 і 1024×768×256), з елементами графічного прискорювача;
- IBM XGA (IBM extended graphics array — розширений масив візуальної графіки) — наступний спеціалізований адаптер IBM, розширено колірний простір (режим 640×480×64k), додано текстовий режим 132×25 (1056×400);
- SVGA (Super VGA — «понад» VGA) — розширення VGA з додаванням вищої роздільної здатності й додаткового сервісу.

Використовують такі різновиди *інтерфейсного кабелю*:

- D-Sub (D-subminiature) — попри назву («надмініатюрний»), найбільший за розмірами електричний роз'єм, що нині застосовують, зокрема в комп'ютерній техніці;
- DVI (digital visual interface — цифровий відеоінтерфейс) — стандарт на інтерфейс і відповідний роз'єм, призначений для передавання відеозображення на рідкокристалічні монітори й проєктори;

- USB (universal serial bus — універсальна послідовна шина) — послідовний інтерфейс передавання даних для середньошвидкісних і низькошвидкісних периферійних пристроїв;
- HDMI (high-definition multimedia interface — мультимедійний інтерфейс високої чіткості) — дає змогу передавати цифрові відеосигнали високої роздільної здатності й багатоканальні цифрові аудіосигнали із захистом від копіювання (HDCP). Роз'єм HDMI забезпечує цифрове DVI-з'єднання декількох пристроїв за допомогою відповідних кабелів. Основна різниця між HDMI і DVI полягає в тому, що роз'єм HDMI менший за розміром, інтерфейс оснащений технологією захисту від копіювання HDCP (High Bandwidth Digital Copy Protection), а також підтримує передавання багатоканальних цифрових аудіосигналів;
- DisplayPort — новий стандарт сигнального інтерфейсу для цифрових дисплеїв. Підтримує HDCP версії 1.3 і має пропускну здатність вдвічі більшу, ніж Dual-Link DVI, низький вольтаж живлення та низькі сторонні наводки, розміри роз'єму Mini DisplayPort приблизно дорівнюють USB.

Якість комп'ютерних моніторів визначається *такими параметрами:*

- *розмір екрана* — довжина діагоналі (традиційно вимірюють у дюймах);
- *співвідношення між горизонтальним і вертикальним розмірами екрана* — випускають монітори зі співвідношеннями 4:3, 5:4, 16:10, 16:9;
- *роздільна здатність дисплея* — кількість пікселів за вертикаллю і горизонталлю;
- *глибина кольору* — кількість бітів на кодування одного пікселя (від монохромного (1 біт) до 32-бітного);
- *розмір зерна* (для CRT) або *пікселя* (для LCD);
- *частота оновлення зображення* — кількість зображень, яка виводиться на екран за одну секунду (у герцах, для LCD майже однакова);
- *швидкість відклику пікселів* (не для всіх типів моніторів, у LCD, як правило, суттєво нижча, ніж у CRT);
- *максимальний кут огляду* — максимальний кут, під яким не виникає суттєвого погіршення якості зображення (актуально для LCD);
- *яскравість* — вимірюється в канделах на квадратний метр;
- *відстань між точками* — дистанція між сусідніми точками одного кольору;
- *тривалість відгуку* — тривалість переходу пікселя від активного стану (білий колір) до пасивного (чорний колір) і назад;
- *дельта-Е* — параметр розрізнення кольорів;
- *потужність споживання* — потужність, яку споживає монітор у робочому стані.

Комп'ютерний принтер (англ. printer — «друкар») — пристрій для друкування текстової інформації і зображень на твердий носій, зазвичай на папір (рис. 4.9).

За кількістю кольорів принтери поділяють на монохромні (однокольорові), які використовують лише один колір (чорний), і багатокольорові. Багатокольорові принтери для передавання кольорів зображення використовують модель CMYK, тобто колір розкладається на базові кольори: cyan — блакитний, magenta — пурпурний, yellow — жовтий, kobalt — кобальт (чорний, названо за основним компонентом, який надає фарбі чорного кольору).

Рис. 4.9

За принципом дії розрізняють матричні, струменеві, лазерні, сублімаційні, 3D-принтери. Розглянемо найпоширеніші види принтерів.

Матричні принтери почали застосовувати одними з перших. Друкування зображень здійснюється за допомогою друкувальної головки, яка складається з матриці голок. Найбільшого поширення набули матричні принтери з 9 та 24 голками. Під дією електромагніту голка вдаряє по паперу через фарбувальну стрічку, залишаючи на папері відбиток у вигляді точки. Друкувальна головка переміщується вздовж рядка, і друк здійснюється послідовно рядок за рядком. Фарбувальна стрічка просякнута фарбувальною пастою, здебільшого чорного кольору, і поступово переміщується під друкувальною головкою, забезпечуючи рівномірне витрачання фарбувального матеріалу.

Основними недоліками матричних принтерів є монохромність, невисока якість друку, низька швидкодія, високий рівень шуму, тож їх поступово витісняють досконаліші моделі. Водночас, зважаючи на їхню невисоку вартість, простоту обслуговування тощо, матричні принтери дотепер широко застосовують у касових апаратах, бухгалтерії, для друку квитків на транспорті, документів нестандартних форматів, друку на рулонах паперу. Фінансові документи й документи суворої звітності, надруковані на матричних принтерах, важко підробити, оскільки під час друку голки деформують поверхневий шар паперу.

У **струменевих принтерах** друкування здійснюється за рахунок мікроскопічних крапель барвника — чорнил, що формуються в мікроскопічних камерах, виштовхуються через дюзи з великою швидкістю і потрапляють на носій (папір або інший матеріал). За принципом формування мікрокрапель струменеві принтери поділяють на термальні й п’єзоелектричні.

Через мікрокамеру з барвником термального принтера проходить короткочасний імпульс електричного струму. У результаті цього різко зростає температура, барвник миттєво закипає, тиск пари стрибкоподібно зростає, під його дією через дюзу вилітають краплини барвника і спрямовуються електричним полем на носій.

У п'єзоелектричних принтерах імпульс тиску в мікрокамері з барвником створюється за допомогою п'єзокристала, який різко змінює свої геометричні розміри під дією імпульсу напруги.

За допомогою струменевих принтерів досить просто створювати багатокольорові зображення, для цього потрібно лише забезпечити принтер чорнилом різних кольорів. Зазвичай потрібний колір отримують за рахунок накладання в різних пропорціях базових кольорів системи CMYK.

Лазерні принтери. Процес друкування в лазерному принтері здійснюється в кілька етапів. Спочатку заряджають фотовал — циліндр, покритий тонким шаром світлочутливого матеріалу. На поверхню фотовала під дією високої напруги рівномірно наносять електричний негативний заряд. Далі фотовал сканують променем лазера, який відбивається від носія графічної або текстової інформації, відхиляється дзеркалом, фокусується оптичною системою і проходить рядок за рядком поверхнею фотовала, розряджаючи електричний заряд у точці його фокусування. Ступінь розряджання електричного заряду залежить від інтенсивності променю, яка модулюється відповідно до яскравості зображення. У такий спосіб на поверхні фотовала створюється рельєф електричного поля, який точно відповідає зображенню.

На оброблений лазером фотовал подають тонер — спеціальний порошок, який слугує барвником. Тонер під дією електричного поля притягується до заряджених ділянок фотовала й утримується на них. Далі тонер із фотовала під дією електричного поля позитивного заряду переноситься на папір і утримується на ньому силами електричного поля.

Зображення на папері з нанесеним тонером закріплюють за рахунок температури й тиску за допомогою печі. Піч складається з двох циліндрів: верхнього, всередині якого розташований нагрівальний елемент, і нижнього, який міцно притискає папір до нагрітої поверхні верхнього циліндра. Притиснутий до нагрітого паперу тонер плавиться і проникає вглиб паперу. Вийшовши з печі, тонер охолоджується і швидко твердне, утворюючи стійке до зовнішніх впливів зображення.

Крім монохромних, останнім часом дедалі більшої популярності набувають кольорові лазерні принтери. Для відтворення всього спектра кольорів застосовують чотири тонери з базовими кольорами системи CMYK, які наносять на папір послідовно один за одним у відповідній пропорції.

Лазерні принтери мають низку переваг порівняно, зокрема, зі струменевими принтерами:

- більша швидкодія, оскільки промінь лазера пересувається швидше, ніж інертна друкувальна головка струменевого принтера;
- висока роздільна здатність за рахунок високої точності фокусування лазерного променю;
- вартість однієї роздруківки лазерного принтера менша, ніж струменевого, оскільки одного картриджа з тонером вистачає на кілька тисяч сторінок, а чорнильні картриджі потрібно значно частіше заправляти або міняти;

- лазерні роздруківки стійкіші до зовнішніх впливів, зокрема до підвищеної вологості;
- злежування тонера після тривалої перерви легко виправити легким струшуванням картриджа, а засихання чорнил у дюзах можна ліквідувати промиванням в умовах сервісного центру.

Графопобудовник, або плотер (від англ. plot — «креслити»), — пристрій виведення інформації з комп'ютера, який із великою точністю робить креслення, схеми, карти на папері великих форматів (рис. 4.10).

а *б*

Рис. 4.10

За способом переміщення носія зображення плотери поділяють на планшетні й рулонні. У планшетних плотерах (рис. 4.10, *б*) носій нерухомо закріплений на плоскому столі, а головка переміщується двома перпендикулярними напрямками. У рулонних плотерах носій закріплено на барабані, який, обертаючись, переміщує носій (рис. 4.10, *а*).

За принципом формування зображення розрізняють векторні й растрові плотери.

За типом пристрою, що наносить зображення, є плотери перові, струменеві й електростатичні. Робочим інструментом перових плотерів є перо, яке переміщується і позиціонується з великою точністю, формуючи векторне зображення. Принцип дії струменевих і електростатичних плотерів аналогічний принципу дії струменевого й лазерного принтерів.

Звукова карта (англ. sound card) — електронний пристрій комп'ютера, призначений для введення і виведення звукової інформації. Звукові карти можуть бути вбудовані в материнську плату (інтегровані) або виконані у вигляді плат розширення (рис. 4.11, *а*).

Цифровий сигнал, записаний у пам'яті або згенерований за допомогою цифро-аналогового перетворювача, що входить до складу звукової карти, перетворюється на аналоговий сигнал, здебільшого на електричну напругу. Напруга підсилюється підсилювачем і подається на *вихідний контактний пристрій*, або *конектор* (англ. connector), звукової карти. До вихідного контактного пристрою можуть бути під'єднані навушники або гучномовець. Щоб відрізнити конектор від інших контактних пристроїв звукової карти, його маркують світло-зеленим кольором.

Цифро-аналоговий перетворювач, підсилювач, фільтри та деякі інші допоміжні пристрої входять до складу мікросхеми звуку, яка розміщується на звуковій карті. Деякі сучасні звукові карти для відтворення звуку високої якості мають кілька мікросхем звуку.

Для введення звукових сигналів у комп'ютер використовують два вхідні контактні пристрої: для слабких сигналів, наприклад із мікрофона, — контактний пристрій рожевого кольору з позначкою «microphone», для звукових сигналів високого рівня — контактний пристрій блакитного кольору з позначкою «line in». Вхідні звукові сигнали підсилюються, перетворюються аналого-цифровим перетворювачем на послідовність двійкових чисел, які заносяться в пам'ять комп'ютера в режимі прямого доступу до пам'яті DMA (англ. direct memory access).

Звукова карта перетворює вхідний звуковий сигнал на електричний або записаний у пам'яті цифровий сигнал на вихідний електричний сигнал. Для відтворення звуку, тобто для перетворення електричного сигналу на звуковий, призначені *гучномовці* (англ. loudspeaker), *навушники* (англ. headphones). Для високоякісного відтворення звуку використовують *акустичні системи* (рис. 4.11, *б*).

Рис. 4.11

4.3. ІНФОРМАЦІЙНІ МАГІСТРАЛІ (ШИНИ)

Комп'ютер являє собою систему порівняно автономних пристроїв, що мають різне функціональне призначення. Потоки інформації пересилаються між цими пристроями за допомогою інформаційних магістралей — *комп'ютерних шин* (англ. computer bus). Характерною ознакою шини є підключення до неї кількох (в окремих випадках усіх) пристроїв комп'ютера.

Основними параметрами шини є розрядність і тактова частота. *Розрядність*, або *ширина шини* (англ. bus width), — кількість провідників у шині, по яких одночасно можна передати всі розряди двійкової інформації. Інтервал часу, впродовж якого здійснюється передавання однієї порції інформації від передавача до приймача, називають *тактовим періодом*, або просто *тактом*. *Тактова частота* — це величина, обернена до тактового періоду.

Шина, з фізичного погляду, — це набір електричних провідників, по яких передають електричні сигнали. З механічного погляду — це плата друкованого монтажу з мідними доріжками на ізоляційній основі (рис. 4.12), на якій закріплено з'єднані з доріжками *контактні пристрої — конектори*, призначені

для механічного й електричного з'єднання пристроїв комп'ютера, які також виготовляють у вигляді плат друкованого монтажу. Такі плати друкованого монтажу з припаяними на них мікросхемами називають *платами розширення*.

Рис. 4.12

Пристрої комп'ютера під'єднують до шини за допомогою *драйвера* — спеціального пристрою, що забезпечує *необхідну потужність* вихідних сигналів пристрою і *допустимий рівень спотворень* під час передавання сигналів через шину.

Протокол шини — це система правил, яких слід дотримуватися, щоб забезпечити правильне функціонування шини.

Система шин сучасного комп'ютера є основним апаратно-програмним ресурсом, що забезпечує взаємодію компонентів комп'ютера на системному рівні. Частини сучасного комп'ютера мають складну внутрішню будову, різні ієрархічні рівні й працюють за складними алгоритмами. Для забезпечення взаємодії складників комп'ютера через шину використовують спеціальний пристрій керування шиною, з відповідним програмним забезпеченням — контролер шини (англ. controller — «пристрій керування»).

Види шин. За способом передавання двійкової інформації шини поділяють на *послідовні* і *паралельні*. У *послідовних* розряди двійкового числа передаються по шині послідовно, один за одним по тому самому провіднику. У *паралельних* шинах всі розряди двійкового числа передаються одночасно, кожен розряд окремим провідником. У послідовних шинах значні затрати часу, але малі апаратні затрати, а в паралельних — навпаки.

За видом інформації, яка передається по шинах, розрізняють:

- шини даних (англ. data bus);
- шини адрес (англ. address bus);
- шини керування (англ. control bus).

За цільовим призначенням:

- шини «процесор–пам'ять»;
- шини введення і виведення;
- системні шини.

Між процесором і основною пам'яттю здійснюється найінтенсивніший обмін інформацією, тому доцільно для цього використовувати окрему високопродуктивну шину. У сучасних мікропроцесорах таку шину називають шиною переднього плану (англ. front-side bus, FSB).

Шина введення і виведення слугує для пересилання інформації між процесором або пам'яттю і пристроями введення і виведення. Більшість периферійних пристроїв набагато повільніші за процесор (за винятком відеосисте-

ми), тому така шина не має великої пропускної здатності. Типові приклади: паралельні шини PCI (peripheral component interconnect) та SCSI (small computer system interface) і послідовна шина USB (universal serial bus). Для зв'язку з відеосистемою використовують локальну шину з великою пропускною здатністю AGP (accelerated graphics port).

Шину, спільну для пам'яті і пристроїв введення і виведення, називають системною. Серед стандартизованих системних шин універсальних комп'ютерів найпоширенішими є шини Unibus, Fastbus, Futurebus, VME, Multibus-II. У персональних комп'ютерах використовують шини ISA (Industry Standard Architecture), EISA (Extended ISA), MCA (Micro Channel architecture).

ТЕСТОВІ ЗАВДАННЯ

ЯКИЙ ПРИСТРІЙ ВИКОРИСТОВУЮТЬ ДЛЯ ПЕРЕТВОРЕННЯ ГРАФІЧНОЇ ІНФОРМАЦІЇ НА ЕЛЕКТРОННУ?

1. Сканер
2. Вебкамера
3. Лазерний принтер
4. Плотер

В ЯКОМУ ПРИНТЕРІ ВИКОРИСТОВУЮТЬ МАТРИЦЮ ГОЛОК?

1. Матричному
2. Лазерному
3. Струменевому
4. Сублімаційному, з внутрішнім нагрівальним елементом

Розділ 5
ПРОЦЕСОРИ (МІКРОПРОЦЕСОРИ)

Процесор (англ. processor) — головний пристрій («серце») комп'ютера, призначений для оброблення інформації у формі двійкових чисел. В англомовній технічній літературі для позначення процесора широко використовують абревіатуру *CPU (central processor unit)*, тобто центральний процесорний пристрій. Сучасні процесори виготовляють на поверхні пластини (чіпа) напівпровідникового матеріалу. Вони мають малі розміри, й тому їх називають *мікропроцесорами*.

Мікропроцесор у складі комп'ютера виконує такі основні *функції*:

- визначення адрес команд і операндів;
- вибір і пересилання команд з основної пам'яті;
- дешифрування цих команд;
- вибір і пересилання даних із пам'яті, а також із регістрів процесорів і адаптерів;
- приймання і оброблення запитів та команд із контролерів пристроїв введення і виведення;
- оброблення отриманих даних і запис результатів оброблення в оперативну пам'ять і регістри мікроконтролерів пристроїв введення і виведення;
- формування сигналів керування пристроями комп'ютера.

Мікропроцесор має такі основні *параметри*:

- розрядність;
- робоча тактова частота;
- система команд;
- розмір кеш-пам'яті;
- технологія виготовлення;
- електричні характеристики;
- механічні й конструктивні характеристики.

Розрядність шини даних визначає кількість розрядів двійкових чисел, із якими мікропроцесор здійснює операції. Кількість розрядів впливає на точність і діапазон подання двійкових чисел. Від розрядності шини адрес залежить розмір адресного простору основної (оперативної) пам'яті, яка адресується мікропроцесором.

Робоча тактова частота мікропроцесора визначає його швидкодію, оскільки кожна команда виконується за певну кількість тактів.

Система команд — це перелік, вид і тип команд, які може виконувати мікропроцесор. Використовують різні типи мікропроцесорів: CISC — із повним набором команд; RISC — зі скороченим набором команд; VLIW — із командами великої довжини тощо.

Кеш-пам'ять зменшує кількість звернень до основної пам'яті, а отже збільшує загальну швидкодію всієї системи. Кеш-пам'ять сучасних мікропроцесо-

рів має зазвичай два рівні: L1 — кеш-пам'ять першого рівня, розміщена безпосередньо на кристалі мікропроцесора; L2 — кеш-пам'ять другого рівня, розміщена на материнській платі.

Як уже зазначено, мікропроцесори виготовляють за інтегральними технологіями на поверхневому шарі напівпровідникової кристалічної пластини (кристалі). Технології виготовлення мікросхем постійно вдосконалюють, зокрема зменшують розмір основного елемента — транзистора. Перший мікропроцесор 4004 фірми Intel мав розмір транзистора 10 мкм, мікропроцесор 8080 — 6 мкм, мікропроцесор Pentium 4 виготовляють за 0,09 мкм технологією.

З електричних характеристик найважливішою є напруга живлення мікропроцесора, яка має тенденцію до зниження. 8-розрядний мікропроцесор 8080 живився від напруг +5 В, –5 В, 12 В; 16-розрядний мікропроцесор 8086 — лише від напруги 5 В, а мікропроцесор Pentium 4 має напругу живлення 1,6 В. Зниження напруги живлення зменшує тривалість перехідних процесів, тобто сприяє збільшенню швидкодії. Водночас таке зменшення знижує завадостійкість і збільшує енерговиділення. Відведення тепла від кристала є гострою проблемою, яка стоїть перед розробниками нових видів мікропроцесорів.

До механічних характеристик належать також форма і розмір корпусу, у якому розміщено кристал мікропроцесора, тип і кількість контактів контактного пристрою, за допомогою якого мікропроцесор кріпиться на платі.

5.1. ІСТОРІЯ РОЗВИТКУ ПРОЦЕСОРІВ

Історія розвитку процесорів повністю відповідає історії розвитку комп'ютерної техніки загалом.

На *першому етапі* (від 1940-х до кінця 1950-х рр.) процесори створювали на основі електромеханічних реле й електронних вакуумних ламп. Такі процесори мали низьку надійність, малу швидкодію і значне енергоспоживання.

Під час *другого етапу* (із середини 1950-х до середини 1960-х) для виготовлення процесорів застосовували транзистори, які монтували на платах друкованого монтажу й встановлювали у стійки. Процесор складався з кількох стійок.

Із застосуванням мікросхем почався *третій етап* (від середини 1960-х) розвитку процесорів. У технології виготовлення мікросхем застосовували кристали напівпровідникового матеріалу (кремнію), на поверхневому шарі якого виготовляли електронні компоненти (транзистори, резистори, конденсатори тощо), і з'єднували провідниками в єдиному технологічному циклі. Така технологія дала змогу в багато разів зменшити розміри електронних елементів і їхнє енергоспоживання. На початкових етапах створювали мікросхеми малого ступеня інтеграції, які містили кілька електронних елементів. Із розвитком інтегральних технологій мікросхеми містили вже цифрові пристрої — тригери, регістри, суматори. Пізніше

з'явилися мікросхеми окремих пристроїв процесора — арифметико-логічні пристрої, мікропрограмні пристрої керування тощо.

Четвертим етапом розвитку (з початку 1970-х) стало створення великих і надвеликих мікросхем, на поверхні кристалу яких розміщувалися спершу мільйони транзисторів, а згодом і весь процесор. Такі пристрої, повністю розташовані на одній мікросхемі, назвали *мікропроцесорами*. Поступово майже всі процесори почали виготовляти за інтегральними технологіями як мікропроцесори. Нині слова «процесор» і «мікропроцесор» є синонімами.

Перший мікропроцесор випустила фірма Intel у 1971 р. — 4-розрядний мікропроцесор 4004, який складався з понад 2 тисяч транзисторів і мав тактову частоту 92,4 кГц. Наступним етапом був випуск фірмою Intel у 1974 р. 8-розрядного мікропроцесора 8080. Цей пристрій завдяки своїм характеристикам дістав широке застосовування в різних галузях, його використовують і в наш час.

Завдяки масовому виробництву порівняно дешевих мікропроцесорів став можливим випуск *персональних комп'ютерів*, які зробили революцію в інформаційній сфері.

Інтегральна технологія виготовлення мікропроцесорів стрімко розвивається; їхні характеристики — розрядність, тактова частота, продуктивність — постійно поліпшуються. Наприклад, для продуктивності комп'ютера діє *закон Мура (Moor)*: продуктивність мікропроцесорів подвоюють через кожні 1,5 року. Зараз випускають 64-розрядні мікропроцесори з тактовою частотою кілька Гігагерц. Сучасні мікропроцесори стали *багатоядерними*, тобто складаються з кількох (двох, чотирьох) порівняно автономних пристроїв оброблення інформації (ядер), які працюють паралельно.

5.2. КЛАСИФІКАЦІЯ ПРОЦЕСОРІВ

У процесі розвитку комп'ютерної техніки було розроблено багато типів мікропроцесорів відповідно до потреб різних галузей виробництва й побуту. Все розмаїття мікропроцесорів можна класифікувати за такими характеристиками.

1. За призначенням:

- загального призначення, або універсальні;
- спеціалізовані.

Серед спеціалізованих мікропроцесорів треба виділити окрему групу *мікроконтролерів*, призначених для роботи в системах керування в режимі реального часу.

2. За місцем у структурі комп'ютера:

- центральний процесор (CPU — central processor unit);
- сопроцесор — спеціальний процесор, призначений для виконання складних математичних операцій, наприклад операцій із рухомою точкою;
- процесор у складі мультипроцесорної системи;

- контролер пристрою введення–виведення;
- контролер особливих режимів роботи, наприклад контролер переривань, контролер прямого доступу до пам'яті тощо.

3. За системою команд:
- RISC (reduced instruction set) — зі скороченим набором команд;
- CISC (complete instruction set) — із повним набором команд.

4. За типом операндів:
- скалярний мікропроцесор, який обробляє окремі двійкові числа;
- суперскалярний, який паралельно обробляє кілька двійкових чисел, організувавши, здебільшого, конвеєрне оброблення;
- векторний процесор, який обробляє цілі набори чисел (вектори, матриці).

5. За типом керування:
- апаратна реалізація;
- мікропрограмна реалізація.

6. Розрядність. За свою історію мікропроцесори пройшли шлях від перших 4-розрядних до сучасних 64-розрядних. На практиці застосовують також 16-розрядні і 32-розрядні мікропроцесори.

5.3. БУДОВА ПРОЦЕСОРІВ

Процесор, як правило, поділяють на дві частини: *операційну*, призначену для виконання операцій із двійковими числами, і *пристрій керування*, який керує роботою комп'ютера в автоматичному режимі. Вперше у світі таку структуру процесора запропонував видатний український учений, академік Віктор Михайлович Глушков.

Операційна частина сучасних мікропроцесорів складається з одного або кількох *виконавчих модулів* (англ. execution unit), які обробляють інформацію у формі цілих двійкових чисел (чисел із фіксованою точкою) за допомогою відповідного цілочислового *арифметико-логічного пристрою* (arithmetic logic unit, ALU). Виконавчі модулі й *головний модуль керування* становлять *ядро* (англ. core) мікропроцесора.

Засоби керування мікропроцесора поділяють на такі частини: керування пристроями комп'ютерної системи, керування виконанням програм, керування процесом виконання команд. Керування комп'ютерною системою охоплює такі режими, як переривання і прямий доступ до пам'яті, запуск і зупинка процесора. Керування програмами полягає в забезпеченні виконання переходів і циклів, виклику і повернення підпрограм тощо. Керування виконанням команд забезпечує виконання всього циклу команди: вибірки з пам'яті, її декодування, безпосереднє виконання і збереження результатів.

Крім ядра до складу мікропроцесорів входять багато інших модулів, які дають змогу здійснювати складне оброблення інформації з високою продуктивністю. *Модуль оброблення чисел із рухомою точкою* (англ. floating point

unit, FPU) підвищує продуктивність процесора під час оброблення складної математичної інформації.

У деяких мікропроцесорах обчислення адрес здійснюється окремо від інших обчислень спеціальним модулем.

Для короткотермінового збереження вхідних даних, проміжних і кінцевих результатів операцій призначені регістри мікропроцесора, які поділяють на *регістри загального призначення* і *спеціалізовані регістри*. Регістри загального призначення часто об'єднують в єдиний модуль — *регістровий файл*.

Крім регістрів, на кристалі мікропроцесора розміщується кеш-пам'ять першого рівня (англ. primary cache), зокрема кеш даних (англ. data cache) і кеш команд (англ. code cache). Застосування кеш-пам'яті значно зменшує середню тривалість доступу до пам'яті.

Для збільшення продуктивності мікропроцесорів широко застосовують конвеєрне оброблення інформації. Для його реалізації на кристалі мікропроцесора розміщені модуль декодування команд (англ. instruction decoder), модуль виконання команд із випередженням (англ. prefetch unit), модуль передбачення розгалужень (англ. branch predictor).

Важливою частиною мікропроцесора є *інтерфейсний модуль*, який забезпечує взаємодію мікропроцесора через шини даних і адрес із пам'яттю та пристроями введення і виведення.

До складу мікропроцесорів входять також додаткові пристрої, наприклад таймери, схеми синхронізації, скидання тощо.

Будова мікропроцесора 8080A

Мікропроцесор 8080A, структуру якого наведено на рис. 5.1, складається з таких основних частин:

- регістри загального призначення;
- регістри спеціального призначення;
- регістри тимчасового зберігання;
- арифметико-логічний пристрій (ALU);
- пристрій керування;
- внутрішня шина даних.

Розглянемо докладніше ці пристрої.

Регістри загального призначення. 8-розрядні регістри загального призначення B, C, D, E, H, L об'єднані в регістровий файл і через мультиплексор пов'язані з внутрішньою шиною даних. Ці регістри доступні програмісту й можуть бути об'єднані в 16-розрядні пари B, D, H. Регістри H і L, крім загального, мають спеціальне призначення: у них тимчасово зберігається адреса, у регістрі H її старший байт, а в L — молодший. Для тимчасового зберігання другого і третього байтів виконуваної команди призначені регістри W, Z, які недоступні програмісту.

До регістрів спеціального призначення належать: акумулятор, лічильник команд, вказівник стека, регістр коду команди, регістр даних, регістр адреси, регістр ознак і регістри тимчасового зберігання AT і T.

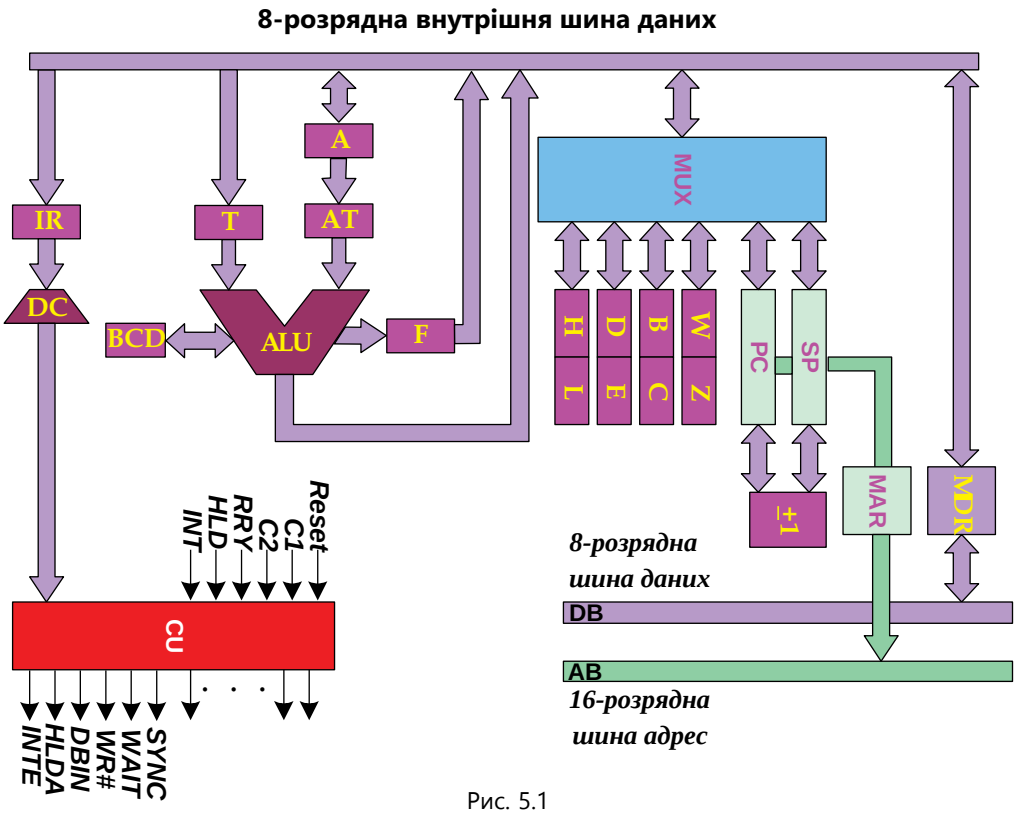


Рис. 5.1

Акумулятор (A) — 8-розрядний регістр, призначений для зберігання результату операції, яку виконує арифметико-логічний пристрій. Крім цього спеціального призначення, регістр A може використовуватися як регістр загального призначення, доступний програмісту.

Лічильник команд (англ. program counter, PC) — 16-розрядний регістр, призначений для зберігання адреси поточної команди. Після виконання чергової команди в лічильник команд автоматично додається одна або кілька одиниць (залежно від кількості байтів у команді) для формування адреси наступної команди.

Вказівник стека (англ. stack pointer, SP) — 16-розрядний регістр, який містить адресу звернення до особливої ділянки пам'яті — стека. У вказівник стека автоматично додається одиниця, коли в стек записується байт інформації, і віднімається одиниця, коли байт зчитується зі стека.

Регістр коду команди (англ. instruction register, IR) — 8-розрядний регістр, у якому зберігається код команди, тобто інформація, яку саме поточну команду з множини команд має виконувати процесор. Код команди подається на дешифратор DC, який розпізнає команду і передає до пристрою керування.

Регістр даних (англ. memory data register, MDR) — 8-розрядний регістр, підключений до шини даних і призначений передавати байт даних у пам'ять і приймати з пам'яті.

Регістр адреси (англ. memory address register, MAR) — 16-розрядний регістр, підключений до шини адрес і призначений для подавання адреси на пам'ять.

Регістр ознак (англ. flag, F) — 8-розрядний регістр, містить такі ознаки результату операції, виконаної арифметико-логічним пристроєм:

- Z (англ. zero) — ознака нульового результату операції (Z = 1);
- CY (англ. carry) — ознака перенесення з найстаршого розряду;
- S (англ. sign) — ознака знаку результату;
- P (англ. parity) — ознака парності кількості одиничних розрядів у результаті;
- AC (англ. auxiliary carry) — ознака додаткового перенесення з молодшого напівбайта в старший;
- T, AT (англ. temp, auxiliary temp) — регістри тимчасового зберігання операндів на час виконання поточної команди і запису результатів в акумулятор.

Арифметико-логічний пристрій (англ. arithmetic logic unit, ALU) — головний пристрій, «серце» мікропроцесора, призначений для виконання таких видів операцій із 8-розрядними двійковими числами:

- пересилання;
- арифметичні;
- логічні;
- зсуву й обертання;
- умовного й безумовного переходів і переходів на підпрограми;
- керування і спеціальні.

Пристрій керування (англ. control unit, CU) — формує сигнали для керування як внутрішніх пристроїв мікропроцесора, так і зовнішніх пристроїв. На мікропроцесор надходять такі сигнали від зовнішніх пристроїв:

- F1, F2 — дві послідовності імпульсів для синхронізації від спеціального генератора;
- READY — сигнал готовності пам'яті або зовнішнього пристрою для пересилання інформації;
- INT (англ. interrupt) — сигнал запиту на переривання;
- RESET — сигнал скидання мікропроцесора в початковий стан;
- HOLD — сигнал запиту на прямий доступ до пам'яті від зовнішніх пристроїв.

З урахуванням цих сигналів, а також стану внутрішніх пристроїв мікропроцесор формує такі вихідні сигнали:

- INTE (англ. interrupt enable) — сигнал підтвердження переривання;
- HLDA (англ. hold acknowledge) — сигнал підтвердження прямого доступу до пам'яті;
- DBIN (англ. data bus input) — сигнал, який свідчить, що шина даних перебуває в режимі приймання інформації;
- SYNC (англ. synchronization) — сигнал, який свідчить, що на шину даних надійшло слово стану процесора;

$\overline{WR}$ (англ. write) — сигнал запису інформації в пам'ять;

WAIT — сигнал очікування готовності даних.

Інформаційні магістралі. Мікропроцесор із пам'яттю і пристроями введення та виведення обмінюється інформацією за допомогою таких магістралей:

1) 8-розрядна шина даних (англ. data bus), якою до мікропроцесора з пам'яті й пристроїв введення надходять дані для оброблення. Цією самою шиною оброблена мікропроцесором інформація передається в пам'ять і пристрої виведення. Отже, шина даних є двоспрямованою, оскільки інформація по ній передається в обох напрямках;

2) 16-розрядна шина адрес (adress bus), якою сформовані мікропроцесором адреси передаються до пристроїв пам'яті для читання і запису інформації. Шина адрес є односпрямованою;

3) шина керування, по лініях якої керівні сигнали, сформовані мікропроцесором, передаються в пам'ять, пристрої введення та виведення інформації. Окремі лінії шини керування призначені для зворотного зв'язку: інформація про стан пам'яті, пристроїв введення і виведення передається в мікропроцесор.

Алгоритм роботи мікропроцесора 8080A

Мікропроцесор 8080A має такі режими роботи:

- основний режим — виконання команд поточної програми;
- переривання;
- прямий доступ до пам'яті.

Розглянемо роботу мікропроцесора в основному режимі роботи. Робота всіх пристроїв мікропроцесора узгоджується в часі, тобто синхронізується за допомогою двох періодичних імпульсних сигналів F1 і F2, зсунутих один щодо одного на половину періоду. Ці імпульсні сигнали формує спеціальний генератор, частота якого має високий ступінь стабільності завдяки використанню кварцового резонатора. Інтервал часу, що дорівнює періоду сигналів F1, F2, називають *тактом*. Це тривалість однієї елементарної операції, яку виконують пристрої мікропроцесора, наприклад запис інформації в регістр, зчитування з регістра тощо. Залежно від пристроїв мікропроцесора, які виконують елементарні операції, такти поділяють на п'ять видів — T1, T2, ..., T5.

Послідовність тактів, упродовж яких виконується сукупність елементарних операцій для реалізації одного звернення мікропроцесора до пам'яті, — це *машинний цикл*. Він може мати від 3 до 5 тактів: M1, M2, ..., M5.

Під час виконання поточної команди мікропроцесор узгоджує свою роботу з роботою пам'яті й пристроїв введення–виведення інформації, генеруючи *сигнали керування*, які поділяють на два види:

- сигнали INTE, HLDA, DBIN, SYNC, $\overline{WR}$, WAIT, які виводять на окремі контакти (виводи) мікропроцесора і які наявні в кожному такті;
- сигнали, які залежать від стану внутрішніх пристроїв мікропроцесора, згруповані у 8-розрядне двійкове слово (байт) — *слово стану процесо-*

ра (англ. processor state word, PSW). PSW видається на шину даних у кожному першому машинному циклі M1, у другому такті T2 і супроводжується сигналом синхронізації SYNC. Сигналом SYNC слово стану процесора записується в спеціальний регістр, виходи якого під'єднано до шини керування.

Слово стану процесора PSW — це байт даних, у розряди якого заносяться такі сигнали:

- D0 (INTA) — підтвердження переривання;
- D1 ($\overline{WO}$) — операції запису в пам'ять або в пристрій виведення;
- D2 (STACK) — запис у стек;
- D3 (HLTA) — підтвердження прямого доступу до пам'яті;
- D4 (OUT) — звернення до пристроїв введення–виведення;
- D5 (M1) — пересилання першого байта команди (коду операції) з пам'яті в процесор;
- D6 (INP) — пересилання інформації з пристрою введення–виведення в процесор;
- D7 (MEMR) — зчитування інформації з пам'яті.

Мікропроцесор 8080A використовує десять видів машинних циклів, які визначають комбінацією розрядів слова стану процесора:

- FETCH (M1) — доставлення першого байта команди, що містить код операції, з пам'яті в процесор;
- MEMORY READ — читання байта інформації з пам'яті;
- MEMORY WRITE — запис байта інформації з пам'яті;
- STACK READ — читання байта інформації зі стека;
- STACK WRITE — запис байта інформації в стек;
- INPUT — читання байта інформації з пристрою введення;
- OUTPUT— запис байта інформації в пристрій виведення;
- INTERRUPT — переривання, тобто тимчасове призупинення виконання поточної програми і перехід до виконання спеціальної програми — програми оброблення переривань;
- HOLD — тимчасове призупинення виконання поточної команди, звільнення шин адрес, даних і керування та передавання їх у тимчасове користування пристроям введення–виведення для прямого доступу (запис або читання) до пам'яті;
- HOLD&INTERRUPT — прямий доступ до пам'яті під час переривання.

Цикл команди — це інтервал часу, потрібний для доставлення і виконання однієї команди. На етапі доставлення поточна команда (1-байтна, 2-байтна або 3-байтна) вибирається в пам'яті і пересилається через шину даних у регістр команд процесора. На етапі виконання команда дешифрується і перетворюється на послідовність керівних логічних сигналів.

Кожна команда може мати від одного до п'яти машинних циклів. Кожне звернення процесора до пам'яті або пристроїв введення–виведення виконується впродовж одного машинного циклу. На доставлення кожного байта команди потрібно виділити один машинний цикл, тому цикл 3-байтної команди

складається мінімум із трьох машинних циклів. Важливо зазначити, що машинний цикл FETCH (M1) міститься в циклі будь-якої команди.

Тривалість етапу виконання команд залежить від виду команди. У деяких команд етап доставлення і етап виконання здійснюються впродовж одного машинного циклу. Інші команди потребують додаткових машинних циклів для запису або читання інформації з пам'яті чи пристроїв введення–виведення. Враховуючи те, що цикл команди може мати від одного до п'яти машинних циклів, його тривалість може дорівнювати від чотирьох до вісімнадцяти тактів.

Розглянемо для прикладу виконання циклу двох команд.

Команда ADD r — це однобайтна команда додавання байта даних одного з регістрів загального призначення (B, C, D, E, H, L) до байта даних акумулятора. Оскільки всі дані, необхідні для виконання операції додавання, вже містяться в регістрах процесора, то потрібно лише одне звернення до пам'яті до доставлення в процесор єдиного байта команди, що містить код операції. Отже, команда ADD r виконується впродовж лише одного машинного циклу FETCH (M1).

У першому такті T1 у регістр адреси з лічильника команд PC завантажується адреса першого байта виконуваної команди. Із затримкою на половину такту на шину даних видається слово стану програми і вихідним сигналом SYNC заноситься в спеціальний регістр для формування сигналів керування на шині керування.

Наявність сигналу готовності пам'яті READY перевіряється в другому такті T2. Якщо сигналу READY немає, то процесор переходить у стан очікування WAIT, який може тривати кілька тактів. Якщо ж сигнал READY з'явився, то процесор переходить у третій такт.

Перший байт команди, вибраний у пам'яті за адресою на шині адрес, пересилається через шину даних у процесор і завантажується в регістр команд IR.

Четвертий такт T4 призначений для дешифрування коду команди з наступним її виконанням. У конкретному випадку команди ADD r після її дешифрування пристроєм керування формується послідовність керівних сигналів, під дією яких до першого входу арифметико-логічного пристрою під'єднується акумулятор A, а до другого — один із регістрів загального призначення, визначений кодом команди. Одночасно на арифметико-логічний пристрій надходить сигнал, який із набору операцій вибирає операцію додавання. Здійснюється операція додавання, і її результат завантажується в акумулятор A. На цьому операція ADD r закінчується. Як ми бачимо, цю операцію було виконано за чотири такти T1, T2, T3, T4 одного машинного циклу M1.

Розглянемо тепер цикл 3-байтної *команди SHLD*, яка виконується за п'ять машинних циклів. Командою SHLD вміст регістрів H і L пересилається в пам'ять і завантажується у два сусідні елементи пам'яті, причому адреса, за якою завантажується вміст регістра H, на одиницю більша за адресу, за якою завантажується вміст регістра L. Адресу, за якою завантажується вміст регі-

стра L, занесено в другий і третій байти команди. Команда SHLD виконується впродовж послідовності таких п'яти машинних циклів:

- M1 (FETCH). Упродовж чотирьох тактів цього машинного циклу з пам'яті в регістр команд процесора IR пересилається перший байт команди — її код. В останньому такті цього машинного циклу інкрементується лічильник команд PC, тобто його вміст збільшується на одиницю;
- M2 (MEMORY READ). Цей машинний цикл складається з трьох тактів T1, T2, T3. За цей час із пам'яті в регістр тимчасового зберігання Z пересилається другий байт команди. Лічильник команд знову інкрементується;
- M3 (MEMORY READ). За три такти цього машинного циклу в регістр тимчасового зберігання W процесора пересилається третій байт команди. Після інкрементації лічильник команд вказуватиме адресу наступної команди;
- M4 (MEMORY WRITE). У цьому машинному циклі вміст регістра L пересилається в пам'ять за адресою, яка міститься в парі регістрів W, Z. Наприкінці машинного циклу вміст пари регістрів W, Z інкрементується;
- M5 (MEMORY WRITE). В останньому машинному циклі команди SHLD за адресою, яка міститься в парі регістрів W, Z, вміст регістра H пересилається в пам'ять.

ТЕСТОВІ ЗАВДАННЯ

ЯКІ ВІДМІННОСТІ ІСНУЮТЬ МІЖ МІКРОПРОЦЕСОРОМ І ПРОЦЕСОРОМ?

1. Процесор — це основна частина комп'ютера, а мікропроцесор — проста мікросхема на материнській платі
2. Пристрої мають різні функції
3. Система команд і швидкість їх виконання різні
4. Немає жодних відмінностей

ЗА ЯКИМИ ХАРАКТЕРИСТИКАМИ КЛАСИФІКУЮТЬ МІКРОПРОЦЕСОРИ, ДО ЯКИХ НАЛЕЖИТЬ ВЕКТОРНИЙ ПРОЦЕСОР, ЯКИЙ ОБРОБЛЯЄ ЦІЛІ НАБОРИ ЧИСЕЛ (ВЕКТОРИ, МАТРИЦІ)?

1. За системою команд
2. За типом операндів
3. За місцем у структурі комп'ютера
4. За призначенням

Розділ 6
ПАМ'ЯТЬ КОМП'ЮТЕРА

Одним з основних інформаційних процесів є збереження інформації. Призначені для цього *пристрої збереження інформації* (англ. data storage device) часто називають *пристроями пам'яті* (англ. memory device), або просто *пам'яттю*.

Інформація, що підлягає обробленню в комп'ютерах і мікропроцесорах, перетворюється пристроями введення у форму двійкових чисел і заноситься в пам'ять. Оброблена процесором і призначена для виведення інформація також зберігається в пам'яті. Таку інформацію називають *даними*. Крім даних у пам'яті комп'ютера зберігаються також *програми* й *адресна інформація*.

Пам'ять разом із процесором є основою комп'ютера. Основними операціями процесора з пам'яттю є *запис* інформації (даних) у пам'ять і *читання* з пам'яті, які об'єднано спільною назвою — *звернення до пам'яті*.

Інформацію, що зберігається в пам'яті комп'ютера, можна поділити на такі види:

- дані, що надходять від пристроїв введення;
- результати операцій, що надходять від процесора;
- програми оброблення інформації, які, своєю чергою, можна поділити на системні й прикладні програми;
- інформація про адреси команд і операндів;
- керівна інформація для периферійних пристроїв комп'ютера.

За фізичною природою пам'ять поділяють на такі види:

- електронна, виготовлена у вигляді інтегральних мікросхем надвеликого ступеня інтеграції;
- магнітна, виготовлена у формі диска, на поверхні якого нанесено тонкий шар магнітної речовини;
- оптична, також у формі диска, на поверхні якого за допомогою лазера змінюються оптичні властивості речовини.

6.1. ЗАГАЛЬНІ ВІДОМОСТІ

Види, характеристики й параметри пам'яті. Пам'ять можна класифікувати за різними ознаками:

- за фізичним принципом — електронна; магнітна; оптична;
- за місцем розташування — внутрішня і зовнішня;
- за способом доступу — з послідовним, довільним, прямим і асоціативним;
- за залежністю від джерела енергії — енергонезалежна і енергозалежна.

Основними параметрами пам'яті є обсяг, швидкодія і вартість. Обсяг пам'яті вимірюють у байтах, кілобайтах (1K = 10^3 байтів), мегабайтах

(1M = 10^3 K), гігабайтах (1G = 10^3 M) і терабайтах (1T = 10^3 G). Показником швидкодії є тривалість доступу до пам'яті, тобто інтервал часу, впродовж якого інформацію записують до пам'яті або читають із пам'яті. Вартість пам'яті визначають кількістю грошових одиниць (доларів або центів) на один біт інформації.

Принцип локальності пам'яті (від лат. locus — «місце») полягає в тому, що звернення до пам'яті, тобто запис або читання інформації з пам'яті, здійснюється не рівномірно по всьому адресному простору, а зосереджене переважно в одному місці. Часто принцип локальності пам'яті виражають формулою «90/10»: 90 % звернень до пам'яті пов'язано з доступом до 10 % усього обсягу пам'яті.

Принцип локальності пам'яті за зверненнями має два аспекти: просторовий і часовий. *Просторова локальність* (англ. spatial locality) полягає в тому, що коди програм і дані зосереджені в невеликій ділянці адрес пам'яті, і ймовірність того, що адреса наступного звернення буде близька до адреси попереднього, дуже велика. Це зумовлено тим, що коди програм мають лінійні ділянки, у яких адреси йдуть підряд, одна за одною. Крім лінійних ділянок програми містять багато циклів, у яких обмежена кількість команд виконується багаторазово. Дані, які обробляють програми, здебільшого структуровані, й такі структури даних розміщені в сусідніх елементах пам'яті.

Часова локальність (англ. temporal locality) полягає в тому, що якщо в поточний момент часу програма звертається до пам'яті за певними адресами, то є велика ймовірність, що в найближчому майбутньому будуть звернення за сусідніми адресами.

Ієрархія пам'яті. У комп'ютерній і мікропроцесорній техніці використовують пристрої пам'яті з різними характеристиками та параметрами, побудовані на різних фізичних принципах. Ці пристрої розміщуються на різних ієрархічних рівнях відповідно до обсягу, швидкодії, вартості тощо.

Є три головні ієрархічні рівні пам'яті комп'ютера: надоперативна, або регістрова; оперативна, або основна; зовнішня, або вторинна.

Надоперативна пам'ять побудована на регістрах, розміщена на тій самій мікросхемі, що й процесор, і фактично є частиною процесора. Швидкодія надоперативної пам'яті є найвищою (тривалість доступу становить частки наносекунди) і сумірною зі швидкодією процесора. Обсяг надоперативної (регістрової) пам'яті сучасних комп'ютерів не перевищує кількох кілобайтів.

Оперативна пам'ять (інші назви — *основна, первинна,* англ. primary memory) є електронною і складається з мікросхем пам'яті. Швидкодія оперативної пам'яті, яка визначається тривалістю доступу, є приблизно на порядок нижчою за швидкодію процесора і не перевищує одиниць мікросекунд. Обсяг оперативної пам'яті лежить у межах від сотень мегабайтів до одиниць гігабайтів і має стійку тенденцію до зростання.

Надоперативна (регістрова) і оперативна пам'ять значно відрізняються одна від одної за швидкодією, обсягом і вартістю. Для подолання такого розриву між ними розміщують проміжну пам'ять — *кеш-пам'ять*.

Зовнішню пам’ять, яку часто називають *вторинною* (англ. secondary memory), реалізовано на магнітних і оптичних носіях інформації. Тривалість доступу до зовнішньої пам’яті становить від одиниць до десятків мілісекунд, а обсяг сягає одиниць терабайтів.

Розрив у характеристиках основної і вторинної пам’яті також долають, вводячи додаткову проміжну — дискову кеш-пам’ять.

Інформацію між ієрархічними рівнями передають так. Спершу процесор звертається до регістрової пам’яті, де зосереджено відповідно до принципу локальності найактуальнішу інформацію, ймовірність потреби в якій є найвищою.

Якщо розподілення інформації між ієрархічними рівнями побудовано за принципом локальності, то в більшості випадків таке звернення є успішним і потрібна інформація з великою швидкістю пересилається між процесором і регістровою пам’яттю.

У разі неуспішного звернення, коли потрібної інформації в регістровій пам’яті немає (що трапляється рідко), відбувається процес пересилання потрібної інформації через багатоступеневу кеш-пам’ять з основної пам’яті в регістрову. Тривалість такого процесу є набагато більшою за тривалість пересилання між процесором і регістровою пам’яттю, і це значно сповільнює роботу програм. Для підвищення загальної ефективності роботи застосовують апаратні й програмні засоби, щоб частка неуспішних звернень до регістрової пам’яті була мінімальною. Крім того, у тих випадках, коли не вдається уникнути такого звернення, пересилати інформацію між ієрархічними рівнями згідно з принципом локальності доцільно не одиничними словами, а цілими блоками.

Пересилання інформації між первинною і вторинною пам’яттю є аналогічним пересиланню між регістровою і первинною, тільки в набагато більшому масштабі, зокрема значно різняться обсяги блоків інформації, що пересилаються між цими ієрархічними рівнями.

Ієрархічну будову пам’яті комп’ютера можна наочно пояснити на прикладі бібліотеки. Весь книжковий фонд бібліотеки розподілений між галузями знань і зафіксований у систематичному каталозі. Книжки, які висвітлюють якесь конкретне питання, зосереджено в одному місці, і в такий спосіб реалізується принцип локальності. Наприклад, книжки з питань виробництва сірчаної кислоти зібрано на одній або кількох полицях одного стенда, а книжки з історії XVIII століття — у зовсім іншому місці, можливо в іншій кімнаті й на іншому поверсі.

Кожна книжка має свій ідентифікаційний номер — аналог адреси в пам’яті комп’ютера.

Коли читач відвідує читальний зал, він замовляє кілька книжок і працює з ними за окремим столом. Кілька книжок на його робочому столі — аналог регістрової пам’яті. Доступ до будь-якої сторінки займає кілька секунд. Щоб продовжити роботу наступного разу, читач просить бібліотекаря не здавати

його замовлення до книгосховища, а залишити на окремій полиці, яка є аналогом кеш-пам'яті.

Зрідка читачеві потрібно повернути взяту книжку, роботу над якою він закінчив, і замовити нову. Нова книжка зберігається в книгосховищі, і для виконання замовлення потрібно почекати приблизно пів години. Книгосховище бібліотеки — аналог основної пам'яті комп'ютера. Тривалість доступу до потрібної інформації в книгосховищі набагато перевищує тривалість доступу до потрібної сторінки книжки, що лежить на робочому столі.

В окремих випадках потрібної книжки немає в одній бібліотеці, однак є можливість замовити її в іншій бібліотеці, скориставшись міжбібліотечним абонементом. Пересилання книжки поштою між бібліотеками триває кілька днів, а іноді місяць. Інші бібліотеки, пов'язані міжбібліотечним абонементом, — аналог зовнішньої пам'яті.

6.2. ОСНОВНА (ОПЕРАТИВНА) ПАМ'ЯТЬ

Основна, або *оперативна,* пам'ять — це пам'ять, до якої процесор звертається безпосередньо під час виконання програм. Оперативна пам'ять пов'язана з процесором шинами адрес, даних і керування. Щоб виконати програму, її потрібно спочатку завантажити в оперативну пам'ять. Виконуючи програму, процесор звертається до оперативної пам'яті і зчитує команди та дані. Після виконання команд процесор пересилає в пам'ять результати.

Пам'ять із довільним доступом (RAM)

Пам'ять із довільним доступом (англ. random access-memory, RAM) — вид пам'яті, у якій тривалість звернення до пам'яті (запис або читання) майже однакова для всіх елементів пам'яті незалежно від розташування елемента й обсягу пам'яті. У пам'ять із довільним доступом можна записувати інформацію, а також читати її, причому тривалість операції запису й читання приблизно однакова. Ця пам'ять належить до виду *енергозалежної* (англ. volatile), тобто інформація в ній втрачається, якщо немає електроживлення.

На початкових етапах розвитку обчислювальної техніки для зберігання інформації застосовували електромеханічні реле, лінії затримки, запам'ятовувальні електронно-променеві трубки. Пізніше широко використовували пам'ять на магнітних сердечниках. На сучасному етапі розвитку майже скрізь застосовують напівпровідникову пам'ять із довільним доступом у вигляді мікросхем.

Статичну пам'ять із довільним доступом (англ. static random-access memory, SRAM) виготовляють за напівпровідниковою інтегральною технологією. Для створення мікросхем пам'яті використовують здебільшого польові комплементарні транзистори, які характеризуються малим споживанням енергії і достатньо високою швидкодією, а також зрідка — біполярні транзистори.

На рис. 6.1, *а* зображено типовий елемент статичної пам’яті. Два інвертори, зібрані на транзисторах *VT1*, *VT2* та *VT3*, *VT4*, з’єднані перехресними зв’язками з виходу на вхід, утворюють тригер, який має два стійкі стани і може зберігати один біт інформації. Транзистори *VT5*, *VT6* призначені для запису й читання інформації. Якщо на затвори транзисторів *VT5*, *VT6* подати сигнал дозволу *WL* (word line), то інформацію про стан тригера буде передано на лінію *BL* (bit line) (читання), або з лінії *BL* — на тригер (запис). Інформацію можна читати і записувати в прямій (*BL*) або інверсній ($\overline{BL}$) формах.

Важливою позитивною рисою статичної пам’яті є велика швидкодія (одиниці наносекунд), яка сумірна зі швидкодією процесора. З іншого боку, наявність в елемента пам’яті шести транзисторів зумовлює значну площу, яку займає цей елемент на кристалі, тому мікросхеми статичної пам’яті значно дорожчі за мікросхеми динамічної пам’яті. Крім того, елемент статичної пам’яті споживає значно більше енергії, ніж елемент динамічної.

Динамічна пам’ять із довільним доступом (англ. dynamic random-access memory, DRAM) побудована на принципі тимчасового (близько кількох мілісекунд) збереження інформації у формі електричного заряду на конденсаторі. Типовий елемент динамічної пам’яті складається з конденсатора і транзистора для заряджання і розряджання цього конденсатора (рис. 6.1, *б*). Наявність заряду на конденсаторі і, відповідно, високий рівень напруги на ньому розглядають як логічну одиницю, а відсутність заряду і низький рівень напруги — як логічний нуль. Заряджений конденсатор через неідеальність ізоляції поступово розряджається, і втрачається записана інформація. Щоб не допустити цього, конденсатор потрібно періодично підзаряджати («освіжувати»).

Сучасні інтегральні технології виготовлення мікросхем динамічної пам’яті бурхливо розвиваються, і завдяки зменшенню розмірів елементів зростає їхня кількість на одиниці площі кристалу. Це дає змогу збільшувати обсяг інформації мікросхем пам’яті й зменшувати вартість, яка припадає на один біт. Оперативну (основну) пам’ять сучасних комп’ютерів побудовано на мікросхемах динамічної пам’яті. Необхідність періодично підзаряджати конденсатори динамічної пам’яті значно збільшує тривалість доступу, порівняно зі статич-

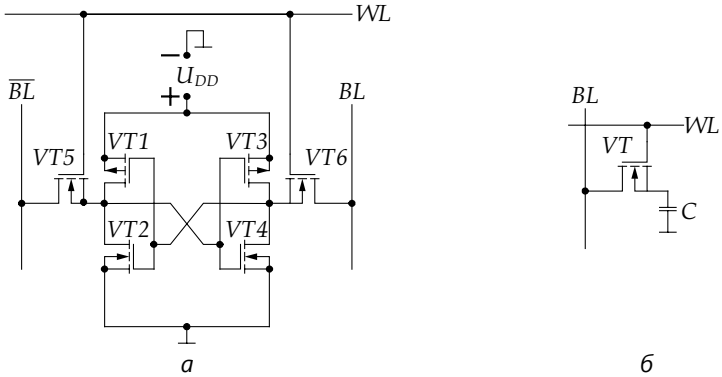


Рис. 6.1

ною пам'яттю, тому динамічна пам'ять значно програє статичній у швидкодії. Динамічна пам'ять більш чутлива до дії сторонніх чинників (світло, тепло) і до завад та шумів.

Постійна пам'ять

Постійна пам'ять, пам'ять тільки для читання (англ. read-only memory, ROM) — тип пам'яті, призначений для зберігання і відтворення даних. Запис інформації у цю пам'ять здійснюється або одноразово під час виготовлення, або особливим способом за допомогою спеціального обладнання. Процедуру запису інформації у постійну пам'ять часто називають *програмуванням пам'яті*, а пристрої, які її виконують, — *програматорами*. Виготовляють таку пам'ять зазвичай у вигляді мікросхем. Постійна пам'ять належить до типу *енергонезалежної* (англ. non-volatile) пам'яті, здатної зберігати занесену інформацію при вимкненні живлення.

Постійну пам'ять застосовують для зберігання програм, які не змінюються або змінюються дуже рідко. Прикладом такої програми є *BIOS (basic input / output system* — базова система введення / виведення), яку використовують для керування і перевірки працездатності комп'ютера одразу ж після увімкнення живлення.

У процесі розвитку комп'ютерної техніки було розроблено кілька різновидів постійної пам'яті.

Mask ROM. Найпростіші *маскові пристрої пам'яті* почали виготовляти у вигляді інтегральних мікросхем із впровадженням у виробництво інтегральних напівпровідникових технологій. Інформацію в ці пристрої заносять у процесі виробництва за технологією фотолітографії, її неможливо змінити впродовж усього періоду експлуатації. Такі мікросхеми економічно вигідно випускати тільки великими партіями. Це доцільно робити, якщо записана в них інформація буде загального вжитку.

PROM. Для задоволення індивідуальних потреб користувачів було створено напівпровідникові мікросхеми з *одноразовим програмуванням* (англ. programmable read-only memory, PROM), у які інформацію записує користувач після виготовлення мікросхеми за допомогою спеціального пристрою — програматора (англ. PROM programmer). Програмування здійснюється шляхом пропускання значного струму через спеціальну перемичку в мікросхемі, у результаті чого вона руйнується і втрачається електричне з'єднання (контакт) між двома електричними провідниками. Втрата електричного контакту сприймається як одне значення логічної величини (наприклад, як логічний нуль), а наявність контакту — як протилежне значення (приміром, логічна одиниця). Будь-яка помилка, допущена під час програмування, робить мікросхему непридатною для користування.

EPROM. Щоб розширити функціональні можливості постійної пам'яті, було розроблено технології виготовлення *мікросхем багаторазового програмування* (англ. erasable programmable read-only memory, EPROM). Інформацію, записану в таку мікросхему, можна було видалити (англ. erase — «витирати,

зішкрябувати») за допомогою сильного ультрафіолетового світла, яким через спеціальне прозоре кварцове віконце опромінювали поверхню мікросхеми. Нову інформацію записували подаванням електричних імпульсів підвищеної напруги. Такі мікросхеми витримували до тисячі циклів видалення і перепрограмування.

EEPROM — це мікросхеми, у яких інформацію видаляють електричним способом, подаючи підвищену напругу (англ. electrically erasable programmable read-only memory, EEPROM). Повторний запис інформації здійснюється також електричним способом і триває значно довше (кілька мілісекунд на біт), ніж процес читання. Видалення і повторний запис інформації в мікросхемі можна здійснювати безпосередньо за місцем розташування в апаратурі без переміщення її в програматор.

Роль елементу пам’яті в мікросхемах з електричним стиранням виконує *польовий транзистор із рухомим затвором* (англ. floating-gate MOSFET, FGMOS). Цей транзистор відрізняється від звичайного польового транзистора наявністю двох затворів: керівного і рухомого, розміщеного в середовищі високоякісного діелектрика (SiO_2) між каналом і керівним затвором (рис. 6.2, *а*). Транзистор із рухомим затвором працює в одному з трьох режимів: запису, стирання і читання інформації.

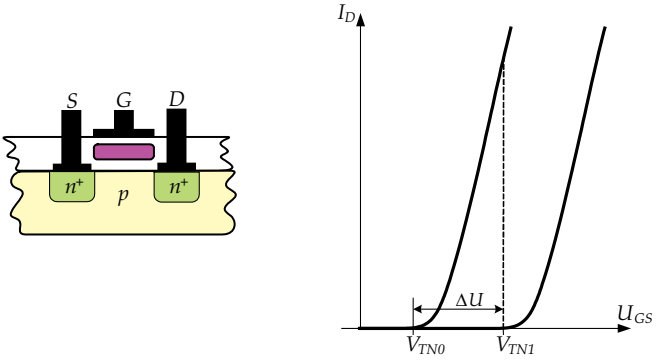
Якщо до стоку транзистора прикласти напругу близько +7 В щодо витоку, а до затвору — близько 12 В, то утвориться провідний канал, яким носії заряду (електрони) рухатимуться від витоку до стоку. Під дією поперечного електричного поля між затвором і каналом електрони зі значною енергією («гарячі» електрони) переходитимуть крізь тонкий шар діелектрика і накопичуватимуться на рухомому електроді, утворюючи на ньому негативний заряд. Завдяки високоякісному діелектрику, що оточує рухомий електрод, заряд на ньому може зберігатися до десяти років. Наявність заряду на рухомому затворі сприймають як логічний нуль, а відсутність — як логічну одиницю. Отже, на транзисторі може зберігатися один біт інформації.

Якщо записану інформацію потрібно видалити (стерти), то між витоком і керівним затвором прикладають обернену напругу: близько +6 В до витоку і близько −9 В до керівного затвору. Під дією цієї напруги електрони з рухомого затвора переходять до витоку завдяки тунельному ефекту.

Для читання записаної інформації до стоку щодо витоку прикладають напругу близько +1 В і створюють поздовжнє електричне поле. Для створення поперечного поля до керівного затвору прикладають напругу близько +5 В. Значення цієї напруги має бути більшим за порогову напругу V_{TN0}.

Якщо рухомий затвор не заряджений, то під дією поперечного поля утворюється провідний канал, яким носії зарядів під дією поздовжнього поля рухаються від витоку до стоку, утворюючи струм (рис. 6.2, *б*).

Якщо ж рухомий затвор заряджений, то він екранує поперечне електричне поле, і порогова напруга транзистора зміщується на ΔU. Прикладена до затвора напруга буде меншою, ніж порогова напруга V_{TN1} транзистора із зарядженим рухомим затвором. Провідний канал не утворюється, і струму між



а *б*

Рис. 6.2

витоком і стоком майже немає. Отже, записаний біт інформації сприймається за наявності (логічна одиниця) або відсутності (логічний нуль) струму в транзисторі.

Флешпам’ять (англ. flash memory) — це різновид пам’яті з електричним видаленням інформації, у якій процес програмування і стирання інформації здійснюється за значно коротший період часу, ніж у традиційній EEPROM пам’яті. Інформація записується і видаляється не окремими розрядами, а цілими блоками. Звідси і походить назва пам’яті: англійське слово flash означає «спалах».

Запис і видалення інформації блоками дає змогу збільшити густину упакування елементів на кристалі й у такий спосіб значно збільшити обсяг інформації в мікросхемах флешпам’яті.

Ці позитивні якості флешпам’яті сприяли широкому впровадженню її в різних галузях і поступовому витісненню традиційних видів пам’яті.

6.3. КЕШ-ПАМ’ЯТЬ

Основна проблема для розробників сучасних комп’ютерів — невідповідність між високою продуктивністю процесорів, точніше мікропроцесорів, і порівняно низькою швидкодією мікросхем динамічної пам’яті (DRAM), на яких побудована основна пам’ять. Із часом проблема загострюється, оскільки продуктивність процесорів подвоюється приблизно через півтора року, тимчасом як швидкодія пам’яті подвоюється приблизно через десять років (закон Мура).

Щоб зменшити розрив між швидкодією процесора і основної пам’яті, застосовують додаткову пам’ять порівняно невеликої ємності зі швидкодією, сумірною зі швидкодією процесора, побудовану на мікросхемах статичної пам’яті (SRAM). *Кеш-пам’ять* (від англ. cache — «таємний склад, схованка») повністю керується операційною системою і недоступна для програміста, ніби «схована» від нього. Треба зазначити, що кеш-пам’ять — не додаткова

пам’ять, що розширює адресний простір основної пам’яті. У кеш-пам’ять на короткий термін під час виконання програми заносяться і зберігаються лише копії команд і даних, якими в той час користується програма. Цей процес називають *відображенням* ділянок основної пам’яті на кеш-пам’ять. Оскільки обсяг кеш-пам’яті набагато менший за об’сяг основної, то в кеш-пам’яті міститься лише обмежена кількість команд і даних виконуваної програми.

Переваги використання кеш-пам’яті ґрунтуються на принципі локальності пам’яті, що полягає, як уже зазначено, у великій імовірності використання лише невеликої «локальної» ділянки пам’яті. Значну частину більшості програм становлять лінійні ділянки, на яких команди розміщені одна за одною за сусідніми адресами. Крім того, у програмах часто трапляються цикли, у яких обмежена сукупність команд виконується багато разів.

Відповідно до принципу локальності пам’яті, під час кожного звернення до основної пам’яті в кеш-пам’ять копіюється блоками обмежена кількість команд і даних із локальної ділянки основної пам’яті, ймовірність звернення до яких є високою. Такі блоки інформації називають *лінійками кеша* (англ. line cache). Кеш-пам’ять і основна пам’ять поділені саме на такі блоки.

Крім копіювання блоків інформації за кожного звернення до основної пам’яті, другим важливим принципом кешування пам’яті є збереження в кеш-пам’яті блоку інформації не тільки за останнього звернення, а й кількох попередніх звернень.

Виконання програм із використанням кеш-пам’яті відбувається так. За кожного звернення до пам’яті процесор спочатку звертається до кеш-пам’яті. Якщо потрібну інформацію вже занесено до кеш-пам’яті під час попередніх звернень, вона пересилається в процесор, витрачуючи на це набагато менше часу порівняно з таким пересиланням з основної пам’яті. Факт знаходження потрібної інформації в кеш-пам’яті називають *влученням* (англ. hit), а за невдалого звернення — *промахом* (англ. miss). Ефективність кеш-пам’яті визначається *часткою влучень* (англ. hit rate) у загальній кількості звернень.

У разі невдалого звернення до кеш-пам’яті процесор звертається до основної пам’яті, але при цьому пересилає в кеш-пам’ять не тільки потрібну інформацію, а й цілий блок — лінійку кеша.

Якщо пошук інформації в основній пам’яті здійснюється за адресою, то в кеш-пам’яті — *за ознакою*, тому кеш-пам’ять характеризують як *асоціативну пам’ять*. Такою ознакою слугує належність інформації до певного блоку основної пам’яті. Її зазвичай називають ярликом, або *тегом* (англ. tag). Теги зберігаються в кеш-пам’яті разом з інформацією, яку вони характеризують. Для цього їм відводиться частина кеш-пам’яті — *пам’ять тегів*.

До кожного тегу в пам’яті додаються додаткові розряди, які відображають властивості інформації, що міститься в лінійці кеша. *Розряд V дійсності* (наявності) (англ. valid) показує, що інформація, яка зберігається в цій лінійці кеша, є дійсною (V = 1), тобто точною копією інформації з певного блоку оперативної пам’яті, або недійсною (V =0), тобто не відповідає жодному блоку інформації оперативної пам’яті. Інформація може бути недійсною (V = 0), напри-

клад, якщо до оперативної пам’яті завантажили іншу програму, а копія одного з блоків пам’яті ще залишається в лінійці кеша. *Розряд M модифікації* показує, що у результаті роботи програми інформація в цій лінійці кеша змінилася (M = 1) порівняно з інформацією у відповідному блоці операційної пам’яті, або не змінилася (M = 0).

У сучасних комп’ютерах застосовують складну багаторівневу кеш-пам’ять, здебільшого дворівневу, хоча є багато досліджень, де обґрунтовують доцільність трирівневої і навіть чотирирівневої кеш-пам’яті. Перший рівень сучасних систем кеш-пам’яті (L1) має об’єм, що не перевищує 64 Кбайта і розміщується на одному кристалі з процесором. Другий рівень (L2) має значно більший об’єм (256 або 512 Кбайтів, зрідка 1 Мбайт), а швидкодію і вартість дещо нижчу і реалізується на одній мікросхемі статичної пам’яті (SRAM). Розміщується кеш-пам’ять другого рівня на системній платі, хоча останнім часом спостерігається тенденція розміщення цієї пам’яті на одному кристалі з процесором, завдяки чому значно зменшується довжина з’єднувальних провідників і тому збільшується швидкодія.

Пам’ять на магнітних дисках

Магнітні диски (англ. hard disk drive, HDD) донедавна були основним пристроєм («робочою конячкою») для побудови систем зовнішньої пам’яті. Інформація на магнітний диск записується шляхом намагнічування тонкого шару магнітного матеріалу, нанесеного на поверхню пластини.

Конструкція магнітного диска. Магнітний диск складається з кількох пластин, що закріплені на рухомій осі — шпинделі (рис. 6.3). Кутова частота обертання шпинделя лежить у межах від 4200 до 15 000 обертів за хвилину.

Рис. 6.3

Пластини виготовляють із немагнітного матеріалу — сплаву алюмінію, скла, кераміки. На обидві поверхні пластини наносять тонкий шар магнітного матеріалу, останнім часом для цього здебільшого використовують кобальт. Читання і запис інформації на диск здійснюють за допомогою *головок запису і читання* (англ. read-and-write heads). Головки закріплені на спеціальному рухомому важелі (англ. actuator arm) і розміщуються над поверхнею пластини на висоті в кілька нанометрів. Переміщуючи за допомогою спеціального приводу важіль, можна позиціонувати головки в потрібному місці над поверхнею пластини.

Елементи конструкції магнітного диска є виробами точної механіки, дуже чутливими до механічних пошкоджень, і тому вся конструкція міститься в закритому корпусі (гермоблоці), який захищає від проникнення пилу і слугує екраном для електричних і магнітних полів.

Принцип дії. Щоб записати інформацію на магнітний диск, у головці запису і читання створюють магнітне поле, яке діє на магнітний матеріал поверхні диска, змінюючи його намагніченість. Після припинення дії магнітного поля завдяки залишковій намагніченості матеріалу ця інформація залишається зафіксованою впродовж тривалого часу.

Читають інформацію за рахунок дії залишкового магнітного поля ділянки магнітного матеріалу на чутливий елемент (сенсор), розміщений у головці запису й читання. У ранніх моделях магнітних дисків читання інформації здійснювалося шляхом наведення електрорушійної сили в чутливому елементі головки під дією залишкового магнітного поля. Недоліком такого чутливого елемента є залежність значення наведеної електрорушійної сили від швидкості руху ділянки магнітного матеріалу щодо головки. У сучасних магнітних дисках чутливим елементом є магніторезистор — спеціальний мініатюрний резистор, опір якого залежить від інтенсивності магнітного поля.

Структура даних магнітного диска. Кожна пластина магнітного диска має дві поверхні, які поділені на концентрично розміщені *доріжки* (англ. tracks), ширина яких сумірна з розміром головки. Сусідні доріжки відокремлені одна від одної проміжком із немагнітного матеріалу для уникнення впливу магнітного поля однієї доріжки на іншу. Сукупність доріжок однакового радіуса, тобто розміщених одна над одною, утворює *циліндр*.

Доріжки, своєю чергою, поділені на *сектори* (англ. sectors), які також відокремлені один від одного проміжками. Сектор містить найменшу одиницю інформації вторинної пам'яті. Крім інформації, яку обробляє комп'ютер, у кожному секторі міститься службова інформація: фізична адреса сектора, яка складається з номера циліндра, номера поверхні, або номера головки і номера сектора; засоби для ідентифікації і синхронізації; засоби для перевірки правильності зчитаної інформації і коригування помилок зчитування.

Параметри і характеристики магнітних дисків.

Ємність (англ. capacity) — один із найважливіших параметрів. Ємність сучасних магнітних дисків досягла значення одиниць терабайтів.

Тривалість доступу (англ. access time) характеризує швидкодію пам'яті на магнітних дисках і для сучасних дисків становить одиниці мілісекунд. Тривалість доступу має три складники: тривалість пошуку потрібної доріжки, затримка, зумовлена обертанням, і тривалість перетворення. Розглянемо докладніше ці складники.

Тривалість пошуку потрібної доріжки (англ. seek time) — це час, потрібний, щоб перемістити важіль із головками на потрібну доріжку. Цей параметр залежить від розміщення попередньої і наступної позиції і швидкості переміщення головок. Середня тривалість пошуку потрібної доріжки лежить у межах від 3 до 9 мілісекунд, а максимальна не перевищує 20 мілісекунд.

Затримка, зумовлена обертанням (англ. rotational latency), — це час від моменту фіксування головки на потрібній доріжці до моменту проходження початку потрібного сектора під головкою. Ця затримка залежить від швидко-

сті обертання шпинделя і місця фіксування головки щодо потрібного сектора. Максимальна затримка дорівнює тривалості повного обороту шпинделя.

Тривалість перетворення (англ. transfer time) — це час, упродовж якого повністю, від початку до кінця записується або зчитується інформація в одному секторі.

Апаратне й програмне забезпечення магнітних дисків. Крім механічної частини, до складу магнітних дисків входить досить складне апаратне й програмне забезпечення для перетворення сигналів головок, узгодження і обміну інформацією з комп'ютером, яке зазвичай називають *інтерфейсом*. Найпоширенішим інтерфейсом твердих магнітних дисків є ATA (англ. advanced technology attachment).

Сучасні магнітні диски мають у своєму складі додаткову електронну кеш-пам'ять, яка відіграє таку саму роль, як кеш-пам'ять комп'ютера, і значно поліпшує якісні характеристики магнітних дисків.

Твердотілі накопичувачі

Розрив між швидкодією процесора і зовнішньою пам'яттю на твердих магнітних дисках неухильно збільшується, тож розробники комп'ютерної техніки шукають шляхи підвищення швидкодії зовнішньої пам'яті. Одним із перспективних шляхів подолання такого розриву є розроблення і виробництво *твердотілих накопичувачів* (англ. solid-state drive, SSD). Твердотілі накопичувачі — це пристрої зовнішньої, вторинної пам'яті на основі мікросхем пам'яті, контролера для керування процесами запису і читання інформації та додаткового автономного джерела живлення (акумулятора) (рис. 6.4). Найбільше використовують два типи мікросхем пам'яті: флеш-пам'ять і мікросхеми динамічної пам'яті, аналогічні мікросхемам, що застосовують в оперативній пам'яті.

Рис. 6.4

Твердотілі накопичувачі порівняно з пам'яттю на магнітних дисках мають такі переваги:

- набагато більша швидкодія, яка сумірна зі швидкодією оперативної пам'яті;
- немає рухомих механічних частин, завдяки чому є стійкість до механічних факторів (ударів, падіння тощо) і безшумна робота;
- менші габарити й маса і низьке енергоспоживання.

Завдяки таким характеристикам твердотілі накопичувачі поступово витісняють магнітні диски.

Пам’ять на оптичних дисках

Рис. 6.5

Оптичний диск — це пластина круглої форми, на якій інформацію записують і зчитують оптичним способом за рахунок зміни коефіцієнта відбивання світлових променів (рис. 6.5). На початкових етапах розвитку оптичної пам’яті основу диска виготовляли з полікарбонату, на який наносили тонкий шар матеріалу зі значним коефіцієнтом відбивання (алюміній), а зверху для захисту — шар прозорого лаку. Двійкову інформацію записували так: утворене за допомогою променю потужного лазера заглиблення (англ. pit) на поверхні диска змінювало коефіцієнт відбивання поверхні, і це сприймалося як логічний нуль, а ділянка (англ. land) між заглибленнями зі значним коефіцієнтом — як логічна одиниця. Заглиблення і ділянки між ними розміщували на спіральній доріжці, яка починалася на певній відстані від центра диска і закінчувалася на його краю.

Для зчитування інформації за допомогою спеціального двигуна диск обертався навколо осі, промінь малопотужного лазера спрямовувався на доріжку, відбивався від поверхні і сприймався чутливим елементом. Заглиблення і ділянки на доріжці змінювали коефіцієнт відбивання поверхні, і в такий спосіб модулювалася інтенсивність прийнятого чутливим елементом відбитого променю. На виході чутливого елемента утворювався логічний сигнал, який підсилювався і передавався далі на оброблення.

Перші оптичні диски — *компакт-диски* (англ. compact disc, CD) — призначалися для запису звукової інформації. Аналогічні диски для комп’ютерної техніки — *CD-ROM* (англ. compact disc read-only memory), тобто диск, із якого можна лише зчитувати інформацію. Масове виробництво CD і CD-ROM дисків здійснювали за такою технологією: спочатку на дорогому устаткуванні виготовляли матрицю, з якої потім за допомогою штампувальної машини робили потрібну кількість копій.

Із розвитком технологій з’явилися дешеві пристрої, які дали можливість самим користувачам одноразово записувати інформацію на заготовку («болванку») диска, — *WORM* (англ. write-once read-many), тобто диск з одноразовим записом і багаторазовим читанням, або просто *CD-R*.

Оптичні диски з можливістю одноразового запису інформації явно програвали магнітним дискам, на яких інформацію можна було багаторазово як зчитувати, так і записувати, і тому зусилля провідних фірм із виробництва оптичних дисків спрямували на усунення цього недоліку. Було розроблено технологію видалення і багаторазового запису інформації на оптичний диск, що отримав назву *CD-RW* (англ. rewritable compact disk). За цією технологією запис інформації на оптичний диск здійснювали шляхом зміни за допомогою лазера фазового стану спеціальної речовини (сплаву срібла, індію, антимонію

і телуру) з кристалічного на аморфний. У кристалічному стані ця речовина прозора для світлових променів, а в аморфному — непрозора.

Щоб істотно збільшити обсяг інформації на оптичному диску і мати можливість записувати відеоінформацію, передусім фільми, було розроблено технологію запису інформації на обох сторонах диска на двох шарах речовини, розміщених один над одним. Крім того, за рахунок застосування більш короткохвильового лазера було істотно збільшено ступінь упакування інформації. Диски, виготовлені за такою технологією, — *DVD* (англ. digital videodisk — «цифровий відеодиск», або digital versatile disc — «цифровий багатогранний (універсальний) диск»).

Blu-ray disc — це оптичний диск із високою ємністю, розроблений для запису, перезапису та відтворення відео високої чіткості (з роздільною здатністю 1920×1080 точок) і даних із підвищеною щільністю. Диски Blu-ray підтримують вищу роздільну здатність і досконаліші формати відео та аудіо порівняно з DVD-дисками. Назва цієї технології пов’язана із синьо-фіолетовим лазером, який використовують для зчитування дисків (англ. blue ray — «блакитний промінь»). Диск Blu-ray може зберігати 25 Гб на шар, а двошарові диски можуть вміщувати до 50 Гб.

Технологія виробництва оптичних дисків нині інтенсивно розвивається і вдосконалюється, зокрема триває розроблення оптичних дисків із використанням голографії (англ. holographic versatile disc, HVD), органічних речовин — протеїнів (англ. protein-coated disc, PCD) тощо.

ТЕСТОВІ ЗАВДАННЯ

ЧИМ ЗУМОВЛЕНО НЕОБХІДНІСТЬ ПОБУДОВИ СИСТЕМИ ПАМ'ЯТІ ЗА ІЄРАРХІЧНИМ ПРИНЦИПОМ?

1. Забезпечення необхідної ємності й високої швидкодії за прийнятну ціну
2. Для ефективної взаємодії всіх типів пристроїв пам'яті
3. Забезпечення захисту пам’яті
4. Зменшення розриву в характеристиках основної і вторинної пам’яті

ЯКУ ПАМ'ЯТЬ НАЗИВАЮТЬ НАДОПЕРАТИВНОЮ?

1. Електронну регістрову пам’ять
2. Основну пам'ять
3. Пам'ять на магнітних дисках
4. Зовнішню пам'ять

Розділ 7
ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ. ОСНОВНІ ПОНЯТТЯ

Одним із фундаментальних принципів роботи комп'ютерів є принцип програмного керування. Відповідно до цього принципу комп'ютер функціонує під керуванням програми, що зберігається в його пам'яті.

У сучасних комп'ютерах діє комплекс програм, що має складну ієрархічну структуру, — *програмне забезпечення* (англ. software) комп'ютерів. Крім програм, до програмного забезпечення належать також засоби для написання, налагодження, тестування програм, підтримка їх роботи впродовж життєвого циклу, засоби захисту і багато інших допоміжних засобів.

Програмне забезпечення — це нематеріальна частина комп'ютера, його «інтелект». Воно призначено для керування матеріальною частиною комп'ютера — апаратним забезпеченням. Програмне й апаратне забезпечення утворюють нерозривну єдність і не можуть існувати одне без одного.

Процеси оброблення інформації можна реалізувати як апаратно, так і програмно. Апаратний спосіб реалізації забезпечує більшу швидкодію порівняно з програмним, однак є менш гнучким, оскільки, щоб змінити або модернізувати процедуру оброблення інформації, потрібно замінити один апаратний пристрій, що виконує це оброблення, на інший. Якщо ж цю процедуру реалізувати програмно, то потрібно лише записати в пам'ять замість однієї програми іншу. Якщо за програмної реалізації буде виявлено помилку, її можна виправити навіть на увімкненому комп'ютері, замінивши неправильні команди на правильні. Така сама помилка, допущена за апаратної реалізації, призведе до необхідності замінити один пристрій на інший.

Співвідношення між програмним і апаратним забезпеченням визначається призначенням комп'ютера: якщо йдеться про оброблення інформації у вузькоспеціалізованій галузі за одним або кількома алгоритмами з максимальною швидкодією, апаратне забезпечення значно переважає програмне. Прикладами можуть бути мікропроцесори, вбудовані в побутові пристрої: пральні машини, кухонні комбайни, кавоварки тощо. Комп'ютери широкого призначення мають програмне забезпечення зі складною ієрархічною структурою, оскільки потрібно обробляти інформацію різного ступеня складності в різноманітних галузях. У таких комп'ютерах апаратно реалізовано лише операції, що мають масовий характер, тобто трапляються набагато частіше, ніж решта операцій.

7.1. ВИДИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Програми, що входять до програмного забезпечення, поділяють за призначенням на системні, прикладні та інструментальні.

Системне програмне забезпечення — це комплекс програм, призначених для керування пристроями комп'ютера: процесором, пам'яттю, пристроями введення і виведення. Головним складником системного програмного забезпечення є *операційні системи* — сукупність взаємопов'язаних програм для керування ресурсами комп'ютера, завантаження в пам'ять і запуску програм, взаємодії (інтерфейсу) з користувачами комп'ютера. Частину програм операційної системи може бути записано («зашито») в постійну пам'ять (англ. firmware). Крім програм операційної системи, до системного програмного забезпечення належать *утиліти* (англ. utility — «користь») — програми, що виконують вузькоспеціалізовані службові функції. Для взаємодії з користувачами призначені програми, що становлять інтерфейс користувача. У сучасних комп'ютерах застосовують здебільшого *графічний інтерфейс користувача* (англ. graphical user interface, GUI).

Прикладне програмне забезпечення — це програми, призначені для виконання конкретних задач користувача. Ці програми можуть написати як самі користувачі, так і професійні програмісти. У наш час інформаційні технології широко застосовують майже у всіх сферах життєдіяльності, тому складно перелічити всі види прикладних програм. Прикладне програмне забезпечення можна умовно поділити на такі категорії:

- прикладні програми повсякденного користування — об'єднані зазвичай в єдиний комплекс, мають схожий інтерфейс і можуть звертатися одна до одної безпосередньо. Прикладами таких комплексів програм для роботи в офісах є *Microsoft Office*, *LibreOffice* та *iWork*;
- прикладне програмне забезпечення для інформаційної підтримки роботи підприємств, зокрема роботи планового і фінансового відділів, складського господарства, взаємодії з клієнтами та постачальниками;
- інформаційні системи, призначені для пошуку й оброблення інформації, — є незамінним інструментом працівників сучасних інтелектуальних професій: учених, інженерів, медиків, архітекторів тощо;
- програмне забезпечення проєктних робіт і виготовлення проєктної документації, наприклад системи P-CAD, OrCAD — для розроблення електричних і електронних пристроїв, AutoCAD — для проєктування електричних, механічних та деяких інших пристроїв і систем, ArchiCAD — для архітектурного проєктування тощо;
- навчальне програмне забезпечення для інформаційної підтримки діяльності навчальних закладів;
- програмне забезпечення для моделювання об'єктів і процесів із метою дослідження їхніх властивостей і характеристик;
- програмне забезпечення для проведення дозвілля, зокрема комп'ютерні ігри.

Інструментальне програмне забезпечення (англ. programming tool) — це засоби для проєктування, створення, налагодження і супроводу програм. Треба зазначити, що інструментальне програмне забезпечення є «робочим ін-

струментом» фахівців однієї з найпрестижніших і найінтелектуальніших професій — *програмістів*.

Інструментальне програмне забезпечення можна поділити на такі категорії:

- програмні інструменти (англ. tools) для виконання окремих етапів або видів програмування;
- комплекс засобів для розв'язання вузького класу задач (англ. workbenches);
- інтегровані середовища розроблення (англ. integrated development environments, IDE) — повний комплект засобів для виконання задач упродовж усього життєвого циклу програми.

7.2. АЛГОРИТМИ

Процеси оброблення інформації за допомогою комп'ютера тісно пов'язані з поняттям алгоритму. Під *алгоритмом* в інформатиці розуміють сукупність дій (операцій) з оброблення інформації, які потрібно виконати в певному порядку, щоб отримати бажаний результат.

Поняття алгоритму застосовують не тільки в інформатиці або математиці. Наприклад, рецепти приготування різноманітних страв, які зібрано в кулінарних книжках, можна трактувати як алгоритми. Слово «алгоритм» походить від імені видатного вченого аль-Хорезмі, який жив у місті Хорезм у IX столітті. У своїх наукових працях він уперше у світі виклав правила виконання арифметичних операцій із десятковими числами.

На сучасному етапі розвитку поняття алгоритму є одним із фундаментальних в інформатиці й математиці. Алгоритм має такі основні властивості:

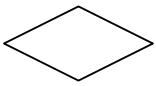
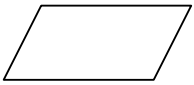
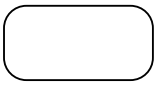
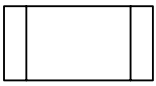
1) універсальність (масовість) — застосовують до різноманітних видів вхідних даних;
2) дискретність — процедура оброблення інформації складається зі скінченної кількості окремих кроків;
3) однозначність — правила і порядок виконання дій не допускають неоднозначного тлумачення;
4) результативність — після виконання алгоритму обов'язково отримують кінцевий результат;
5) скінченність — кінцевий результат отримують за скінченну кількість кроків.

Способи вираження алгоритмів. Алгоритм можна виразити різними способами.

1. *Словесний* — алгоритм описують природними мовами: англійською, українською тощо. Перевагою такого способу є зрозумілість алгоритму для широкого кола користувачів, а не тільки для професіоналів. Недоліком є неоднозначність трактування окремих кроків або всього алгоритму.

2. *Графічний* — алгоритм задають за допомогою умовних графічних символів (див. табл. 7.1). Графічне зображення алгоритму часто називають *блок-схемою* (англ. flowchart). Перевагами графічного способу є наочність, компактність, можливість легко виявляти й виправляти помилки.

Таблиця 7.1

Символи	Значення
	Виконання дій (оператор)
	Розгалуження
	Введення даних
	Початкова і кінцева дії
	Сукупність дій (підпрограма)
	Послідовність виконання

3. *Алгоритмічні мови програмування.* Оскільки комп'ютер сприймає інформацію лише у формі двійкових чисел, алгоритм, призначений для виконання комп'ютером, має бути записаний за допомогою двійкових чисел. Таку форму запису алгоритму називають *мовою машинних команд*. Алгоритм, записаний такою мовою, — це *програма*. Комп'ютер є виконавцем алгоритму, а створює алгоритм і відповідну програму програміст. Створення програм у двійкових кодах є важкою, виснажливою, малопродуктивною роботою, тому використовують спеціальні штучні мови — алгоритмічні мови програмування. Як і природні мови, алгоритмічні мають свою *граматику*, тобто систему положень і правил, згідно з якими записують алгоритми. Алгоритмічні мови наближені до мов і систем позначень, якими користуються спеціалісти в певній галузі, тому створювати алгоритми і відповідні програми такими мовами значно простіше й ефективніше, ніж мовою машинних команд. Створену й відлагоджену програму алгоритмічною мовою потім «перекладають» мовою машинних команд для виконання комп'ютером.

Типові структури алгоритмів. В алгоритмах використовують поєднання елементів, які часто повторюються як усередині конкретного алгоритму, так і в різних алгоритмах. Фактично всі алгоритми складаються з таких типових структур. На рис. 7.1 наведено блок-схеми основних типових структур алгоритмів. Розглянемо докладніше кожну типову структуру.

Лінійна структура (рис. 7.1, *а*) характеризується тим, що дії (операції) виконуються послідовно, одна за одною. Лінійні структури легко надаються для конвеєрного оброблення, яке широко застосовують у сучасних комп'ютерах для підвищення їхньої продуктивності.

Розгалуження (рис. 7.1, *б*) використовують в алгоритмах, коли необхідно перевірити певну умову, наприклад, чи є результат останньої дії нулем (додатним, від'ємним тощо). Якщо умова виконується, то реалізується одна операція або послідовність операцій, а якщо не виконується, то потрібно перейти до виконання інших операцій. Отже, чи виконувати ту чи ту (альтернативну) час-

тину програми — залежить від результату операції, який, своєю чергою, залежить від вхідних даних. Оскільки під час написання програми неможливо передбачити, який буде результат останньої операції перед розгалуженням, передбачити, яку частину програми виконуватиме комп'ютер після розгалуження, можна лише з певною ймовірністю на основі статистичних даних. У сучасних комп'ютерах застосовують механізми передбачень переходів із високим ступенем імовірності.

Напіврозгалуження (рис. 7.1, *в*) відрізняється від розгалуження тим, що за дотримання умови виконується операція або низка операцій, а за недотримання — жодна операція не виконується.

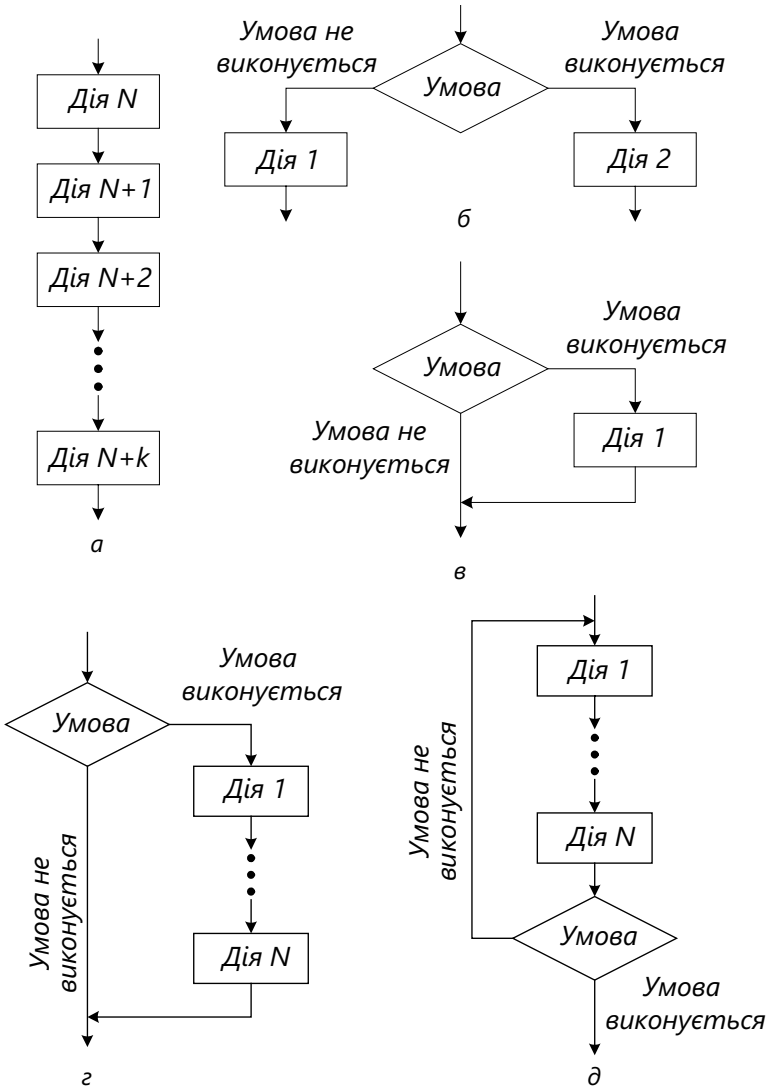


Рис. 7.1

Цикл із передумовою. В алгоритмах оброблення інформації часто трапляється ситуація, коли певну кількість операторів потрібно виконувати багаторазово з іншими вхідними даними. Щоб зменшити обсяг програми й багаторазово не повторювати ті самі оператори, оформлюють структуру — цикл. Потрібні оператори записують лише раз, а кількість повторень виносять в умовний оператор. Якщо умовний оператор стоїть на початку циклу — це цикл із *передумовою* (рис. 7.1, *г*), якщо наприкінці — цикл із *постумовою* (рис. 7.1, *д*).

7.3. МОВИ ПРОГРАМУВАННЯ

Мови програмування — це штучні мови, призначені для запису алгоритмів оброблення інформації за допомогою комп'ютера з метою полегшення роботи програміста і підвищення її ефективності та продуктивності. Якщо природні мови слугують для обміну інформацією між людьми, то мови програмування — для передавання команд (інструкцій) комп'ютеру для їх виконання. Тому в мовах програмування не допускають неоднозначного тлумачення команд: кожна має точно й чітко вказувати, яку дію має виконати комп'ютер.

Ієрархічні рівні мов програмування. За весь час існування комп'ютерної техніки було розроблено близько восьми тисяч різноманітних мов програмування, однак єдиної їх класифікації немає. У технічній літературі мови програмування часто класифікують за ієрархічними рівнями. Чим вищий рівень, тим «ближче» він до людського сприйняття; чим нижче рівень, тим «ближче» до сприйняття комп'ютером, тобто до послідовності двійкових чисел (рис. 7.2).

Програми для перших комп'ютерів писали *машинними кодами,* тобто за допомогою двійкових чисел, кожне з яких означало певну команду. Такі програми являли собою стовпчики двійкових чисел. Щоб створювати такі прог-

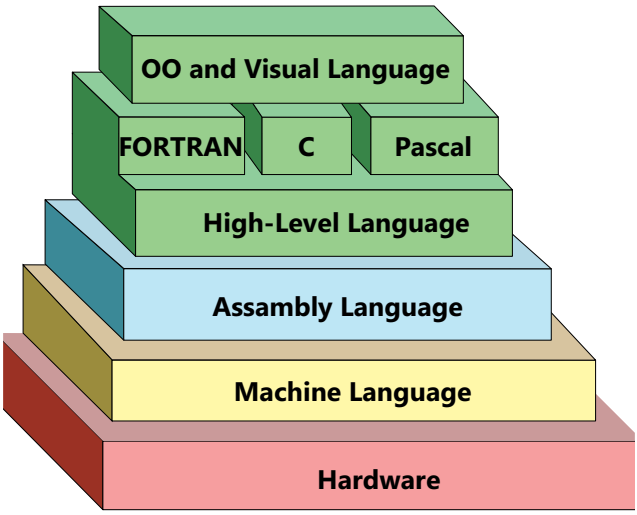


Рис. 7.2

рами, програміст мав досконало знати будову комп'ютера, пам'ятати всі параметри його пристроїв, як-от пам'ять, процесор, пристрої введення–виведення. Крім того, він мав пам'ятати, яке двійкове число означає певну команду, вручну розміщувати програми в пам'яті за певними адресами, які також виражаються двійковими числами, слідкувати, щоб програми в пам'яті не перекривали одна одну під час роботи. Процес створення таких програм був важким і малопродуктивним. Мова програмування машинними кодами є мовою найнижчого рівня.

Щоб полегшити роботу й підвищити продуктивність праці програміста, було розроблено мови *асемблера*. У цих мовах машинні команди, записані двійковими числами, замінювали мнемонічними словами (мнемонікою), тобто такими словами, які легко запам'ятати. Двійкове число, що означало команду додавання, замінювали мнемонікою ADD, що є скороченням англійського слова addition — «додавання». Програмістові значно легше запам'ятати слова, ніж багаторозрядні двійкові числа. Двійкові адреси в програмі також замінювали мітками, вираженими словами. Мови асемблера значно полегшили роботу програмістів, підвищили її продуктивність, дали змогу легко виявляти й виправляти помилки. Оскільки команди для різних комп'ютерів відрізняються, то розробляли мову асемблера для кожного типу комп'ютера. Тому мови асемблера і мови машинних команд називають *машинно-залежними* (вживання слова «машина» пояснюється тим, що на початкових етапах розвитку комп'ютери називали *електронно-обчислювальними машинами (ЕОМ)*).

Перехід із мови машинних команд на мову асемблера значно полегшив створення програм, підвищив продуктивність праці програмістів. Бурхливий розвиток електроніки, зокрема розроблення і широке застосування в комп'ютерах інтегральних мікросхем, зумовив різке збільшення продуктивності комп'ютерів, зменшення їхньої маси, габаритів, енергоспоживання і, головне, їхньої вартості. Такі комп'ютери почали масово застосовувати в різних предметних галузях.

Для програмування мовою рівня асемблера необхідно досконало знати будову конкретного комп'ютера, характеристики і можливості його пристроїв. Щоб використати таку програму на комп'ютерах з іншою будовою, доводиться суттєво переробляти програму, а в багатьох випадках взагалі писати нову.

Крім того, програмування мовами рівня асемблера ускладнює масове використання комп'ютерів фахівцями різних галузей, які не є професійними програмістами.

Для подолання цих труднощів розроблено *мови високого рівня* абстракції, з використанням понять, близьких до прикладних галузей. Наприклад, математичні операції записують у традиційній формі, тому такі програми може створювати й непрофесійний програміст.

На сучасному рівні розвитку інформаційних технологій використовують велику кількість мов високого рівня, які ґрунтуються на різних підходах і концепціях і призначені для оброблення інформації в різноманітних сферах.

7.4. ОПЕРАЦІЙНІ СИСТЕМИ

Операційні системи (англ. operating system, OS) — це комплекс спеціальних програм, призначених для керування апаратними і програмними ресурсами комп'ютера і для взаємодії з користувачем. Операційні системи забезпечують ефективність і надійність роботи комп'ютерів, звільняючи користувача від рутинної роботи з керування апаратними пристроями комп'ютера і програмним забезпеченням і надаючи йому зручний інтерфейс, тобто засоби взаємодії. До найпопулярніших належать такі операційні системи: Android, BSD, iOS, Linux, OS X, QNX, Microsoft Windows, Windows Phone, IBM z/OS.

Операційна система виконує комплекс складних задач. Коротко розглянемо найважливіші з них.

Керування процесами (англ. process management). У сучасних комп'ютерах одночасно на різних стадіях виконуються кілька програм. Виконання кожної програми і підтримку цього виконання з боку операційної системи називають *процесом*. Операційна система розподіляє апаратні й програмні ресурси комп'ютера між процесами, надає можливість процесам розподіляти інформацію і обмінюватися нею, забезпечує захист процесів від взаємного несанкціонованого доступу.

Переривання (англ. interrupt) — це сигнал, який надходить до процесора від апаратних пристроїв або програм і свідчить про певну подію в комп'ютерній системі, яка потребує негайного реагування з боку процесора. Такою подією може бути натискання на клавішу клавіатури або кнопку миші, сигнал готовності пристроїв введення і виведення тощо.

Сигнал переривання формується спеціальним апаратним забезпеченням — *контролером переривань*. Отримавши сигнал переривання, процесор призупиняє виконання поточних програм і передає керування спеціальним програмам — *обробникам переривань* (англ. interrupt handler), які входять до складу операційних систем. Обробники переривань, отримавши у своє розпорядження необхідні ресурси, здійснюють всі необхідні дії і після завершення передають керування перерваній програмі.

Керування пам'яттю (англ. memory management). Пам'ять сучасних комп'ютерів — це складна ієрархічна система, до складу якої входять пристрої пам'яті, різноманітні за обсягом і швидкодією і побудовані за різноманітними фізичними принципами. Керування такою системою є непростим завданням.

Для керування пам'яттю у складі операційної системи є спеціальне програмне й апаратне забезпечення, зокрема *пристрій керування пам'яттю* (англ. memory management unit, MMU).

Керування пам'яттю забезпечує динамічне виділення пам'яті (англ. dynamic memory allocation) програмам, що виконуються, використовуючи для цього механізми сторінкової і сегментної організації пам'яті, віртуальної пам'яті, кеш-пам'яті різних рівнів.

Керування файловою системою. Файлова система — спосіб організації даних, який використовує операційна система для збереження інформації

у вигляді упорядкованої сукупності даних (файлів) на носіях інформації. Файли мають ім'я, розмір, інші атрибути й об'єднуються в каталоги (директорії). Файли й каталоги нижнього рівня можуть входити до каталогу верхнього рівня, утворюючи деревовидну структуру каталогів. Каталог найвищого рівня називають кореневим каталогом.

Керування введенням і виведенням інформації. Для введення інформації в комп'ютер і її виведення застосовують пристрої, побудовані за різними фізичними принципами, із різними характеристиками і призначенням. Керувати такими різноманітними пристроями вручну на фізичному рівні майже неможливо. Операційна система керує пристроями введення і виведення на двох рівнях. На нижньому рівні керування конкретними моделями пристроїв, з урахуванням будови, принципу дії, характеристик і параметрів, здійснюють за допомогою спеціальних програм — *драйверів пристрою*. Виробники зазвичай постачають драйвери разом із пристроями, як їхні невід'ємні частини.

На вищому рівні застосовують *універсальний інтерфейс введення–виведення*, який забезпечує взаємодію, з одного боку з драйверами, а з іншого — з користувачем.

Мережна підтримка. У сучасних операційних системах передбачено засоби для роботи в комп'ютерних мережах із підтримкою протоколів обміну даними. Засоби мережної підтримки дають операційним системам можливості:

- надавати локальні ресурси (дисковий простір, принтери тощо) в спільне користування через мережу, тобто функціонувати як сервер;
- звертатися до ресурсів інших комп'ютерів через мережу, тобто функціонувати як клієнт.

ТЕСТОВІ ЗАВДАННЯ

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ КОМП'ЮТЕРА — ЦЕ:

1. комплекс програм для керування зовнішніми пристроями
2. комплекс програм для керування пам'яттю
3. комплекс програм для керування апаратним забезпеченням
4. комплекс програм для керування введенням–виведенням даних

ПРОГРАМНА ФОРМА ПОДАННЯ АЛГОРИТМІВ — ЦЕ:

1. подання мовами програмування
2. подання словами природної мови
3. подання зображенням із графічних символів
4. подання математичним виразом

Розділ 8
МОВИ ПРОГРАМУВАННЯ НИЗЬКОГО РІВНЯ

8.1. МІКРОКОМАНДИ Й МОВА МІЖРЕГІСТРОВИХ ПЕРЕСИЛАНЬ RTL

Дані у формі двійкових чисел, із якими виконуються мікрооперації (операнди), а також результати виконання мікрооперацій зберігаються в регістрах операційного пристрою, тому послідовність виконання мікрооперацій формально можна записати як послідовність пересилань інформації між регістрами. Засіб, за допомогою якого цей формальний запис здійснюють, називають мовою пересилань даних між регістрами (register transfer language, RTL). Символічна мова RTL — зручний інструмент для опису внутрішньої організації цифрових пристроїв, зокрема мікропроцесорів, а також для їх проєктування. Найпоширеніші символи мови RTL наведено у табл. 8.1.

Таблиця 8.1

Символи	Позначення	Приклади
Великі букви латиниці й цифри	Регістри загального і спеціального призначення	MAR (memory address register) — регістр адреси пам'яті; PC (program counter) — програмний лічильник
( ) круглі дужки	Частина розрядів регістра	R1(0-7), R2(L)
[ ] квадратні дужки	Адреса пам'яті	M[MAR]
—> стрілка	Напрям передавання даних	MAR <— PC
: двокрапка	Умова, за якої виконується мікрооперація	Z: A <— A+B
, кома	Розділяє мікрооперації, які виконуються паралельно в одному такті	R1 <— R2, SP <— SP+1

Мікропрограма виконання машинної команди, записана мовою RTL, складається з послідовності рядків, у кожному з яких записано одну або кілька мікрокоманд, що виконуються впродовж одного такту. Рядки розміщені в тій самій послідовності, що й мікрооперації з даними.

8.2. МАШИННІ КОМАНДИ Й АРХІТЕКТУРА СИСТЕМИ КОМАНД

Машинні команди — це двійкові числа, що зберігаються в пам'яті комп'ютера й пересилаються через шину даних у центральний процесор, який безпосередньо виконує ці команди. Організована сукупність усіх машин-

них команд, які виконує центральний процесор, утворює *систему команд*. Цей ієрархічний рівень комп'ютера в англомовній технічній літературі має назву ISA (instruction set architectures) — архітектура системи команд.

Створення програм за допомогою машинних команд мало місце тільки на початкових етапах розвитку комп'ютерної техніки. Нині комп'ютерні програми створюють на вищому ієрархічному рівні за допомогою алгоритмічних мов високого рівня, а потім перетворюють (транслюють або інтерпретують) на рівень машинних команд.

У більшості сучасних комп'ютерів машинні команди поділено на послідовність елементарних дій — мікрокоманди. Послідовність мікрокоманд, призначених для виконання однієї машинної команди, називають мікропрограмою.

RISC і CISC комп'ютери

На сучасному етапі розвитку комп'ютерної техніки активно розвиваються дві концепції побудови системи команд комп'ютерів: RISC і CISC комп'ютери.

RISC (англ. reduced instruction set computing) — комп'ютер зі скороченим набором команд (інструкцій). Його основні риси:

- невелика кількість команд, більшість з яких виконуються за один такт;
- однакова довжина всіх команд;
- невелика кількість різних форматів команд;
- немає команд, які працюють з операндами в пам'яті;
- для звернення до пам'яті призначені лише дві команди — запис у пам'ять і читання з пам'яті;
- невелика кількість типів даних;
- значна кількість регістрів загального призначення.

RISC комп'ютери застосовують у галузях, де потрібна велика продуктивність: сервери, робочі станції тощо.

CISC (англ. complex instruction set computer) — комп'ютер зі складним набором команд. Основні принципи побудови таких комп'ютерів було розроблено на ранніх етапах розвитку комп'ютерної техніки, коли актуальним було розв'язання проблеми так званого семантичного розриву, тобто розриву між мовою, якою формулювали задачі фахівці (математики, фізики, інженери чи інженерки) і мовою двійкових чисел, яку «розумів» комп'ютер. Щоб подолати семантичний розрив, застосовували команди, які максимально наближаються до математичних операторів. Такі команди були складними, різної довжини і з великою кількістю форматів, багато з них містили звернення до пам'яті. Загальна кількість команд CISC комп'ютерів набагато перевищує кількість команд у RISC комп'ютері. Перевагою системи команд CISC комп'ютерів є їхня компактність, і тому програми, написані з використанням таких команд, займають мало місця в пам'яті. Однак, зважаючи на розвиток мов високого рівня і прогрес у галузі електроніки, переважну частину складних команд нині дуже рідко застосовують, і це є основним недоліком CISC комп'ютерів.

Сучасні RISC і CISC комп'ютери розвиваються паралельно, враховуючи останні технологічні досягнення і зі взаємними запозиченнями. Щодалі в них стає все більше спільних рис і все менше відмінностей.

Структура машинної команди

Машинна команда складається з двох частин: операційної й операндної.

В *операційній* частині міститься двійковий код операції, яку процесор має виконувати з даними у формі двійкових чисел. Прикладом таких операцій може бути додавання або віднімання двох чисел, переміщення двійкових чисел із регістра процесора до пам'яті або навпаки — з пам'яті до регістра тощо. У багатьох випадках застосовують найпростішу схему кодування: всю сукупність операцій, які здатен виконувати процесор, нумерують двійковими числами, і цей двійковий номер операції і є кодом операції.

В *операндній* частині містяться операнди — двійкові числа або, найчастіше, адресна інформація про їх розташування.

8.3. РЕЖИМИ АДРЕСАЦІЇ

Система адресації — сукупність програмних і апаратних засобів знаходження потрібної інформації в пам'яті комп'ютера або в пристроях введення–виведення. Оскільки пам'ять сучасних комп'ютерів має складну ієрархічну будову, система адресації такої пам'яті також є складною. Застосовують різноманітні засоби для її реалізації.

Режимом адресації (англ. address mode) називають спосіб знаходження адрес команд і даних. Залежно від розташування інформації адресацію поділяють на такі види:

- регістрова;
- адресація основної пам'яті;
- адресація пристроїв введення–виведення.

Кількість регістрів у регістровому файлі процесора набагато менша за кількість елементів основної пам'яті, тому для адресації регістрів потрібна невелика кількість розрядів операндної частини команди. Крім того, тривалість доступу до регістрів набагато менша, ніж до основної пам'яті. Саме через це RISC процесори використовують здебільшого триадресну регістрову адресацію.

До пристроїв введення–виведення процесор звертається за їхніми номерами як до елементів пам'яті, тому ці пристрої і основна пам'ять утворюють єдиний адресний простір.

Залежно від того, як задають адресу, розрізняють *явну* (англ. explicit addressing) і *неявну* (англ. implicit addressing, або inherent addressing) адресацію. За явної адресації адресу задають в операндній частині команди, а за неявної — кодом команди, операндної частини команди взагалі немає. Прикладами неявних команд є такі команди мікропроцесора I8080A: CMA — ін-

вертування розрядів акумулятора, STC — установка розряду переносу в 1 регістра ознак, CMC — інвертування цього ж розряду.

Якщо в операндній частині команди міститься сам операнд (операнди), то таку адресацію називають *безпосередньою* (англ. immediate mode). Її застосовують зрідка для адресації констант. Наприклад, 2-байтна команда **MVI A, byte** мікропроцесора Intel 8080A переміщує операнд, що міститься в другому байті команди, в акумулятор. Перевагою безпосередньої адресації є її простота, оскільки за допомогою одного звернення до пам'яті можна отримати і код операції, і операнд. Недоліком безпосередньої адресації є необхідність переписувати програму в разі зміни даних.

Адресацію за способом визначення адрес поділяють на *пряму*, або *абсолютну* (англ. direct mode), і *непряму* (англ. indirect mode). В операндній частині команди в режимі прямої адресації міститься адреса, за якою в пам'яті зберігається операнд. Для виконання операції за цією командою необхідно двічі звернутися до пам'яті: за першим разом у процесор із пам'яті передається власне команда, а за другим — операнд. Перевагою прямої адресації є просте визначення адреси операнда. Недоліком — необхідність відводити під адресу значну кількість розрядів, адже розмір адресного простору сучасних комп'ютерів дуже великий. Другим недоліком є неможливість переміщувати програми і дані в пам'яті, тимчасом як у сучасних комп'ютерах широко застосовують динамічний розподіл пам'яті між програмами й користувачами, і операційна система постійно переміщує програми для збільшення ефективності.

Щоб подолати ці недоліки, широко застосовують непрямі системи адресації, у яких адреси визначають шляхом складних обчислень. Однак таку складність виправдано тим, що в результаті вдається досягнути високої продуктивності комп'ютера загалом, ефективного використання пам'яті й забезпечити захист пам'яті.

За *регістрової* адресації (англ. register mode) операнд міститься не в пам'яті, а в регістрі процесора. Регістри мають набагато більшу швидкодію, ніж пам'ять, тому команди з регістровою адресацією виконуються набагато швидше. Крім того, адреса регістра набагато коротша, тобто має набагато меншу кількість двійкових розрядів, ніж адреса пам'яті. Недоліком регістрової адресації є обмежений адресний простір, тобто обмежена кількість адрес регістрів.

Непряма регістрова адресація (англ. indirect register mode) подібна до непрямої адресації пам'яті. В операндній частині команди міститься адреса регістра, який зберігає адресу операнда.

Відносна адресація (англ. relative addressing mode). Коди команд програм і дані мають властивість локальності, тобто коди сусідніх команд і даних розміщені якщо не в сусідніх, то у близьких елементах пам'яті. Отже, у процесі виконання програми в адресах команд і даних змінюється тільки незначна частина розрядів, решта ж розрядів залишаються незмінними, тому недоцільно кожного разу при виконанні команд виконувати операції з цією незмінною частиною адрес.

Через це адресу поділяють на дві частини: *незмінна базова* частина адрес зберігається в одному з регістрів процесора, а *змінна* частина адреси (або *зміщення* (англ. displacement)) — в операндній частині команди. Отже, у команді міститься адреса *відносно* певної базової адреси, звідси й назва такого способу адресації — *відносна* (англ. relative addressing mode). Зауважимо, що програміст має справу, як правило, лише з другою частиною адреси, тобто зі зміщенням, операції з базовою частиною адреси покладено на програми операційної системи.

Відносна адресація має ще й інші переваги. Наприклад, можна переміщувати програми в пам'яті, не вносячи змін у команди програми, здійснюючи захист інформації від несанкціонованих змін тощо.

Сторінкову адресацію застосовують у разі, якщо пам'ять має сторінкову організацію, тобто поділена на рівні частини — *сторінки*. Початкові адреси всіх сторінок зведено в *таблицю сторінок,* яка зберігається в пам'яті. Початкову адресу таблиці сторінок розміщено в спеціальному регістрі процесора.

Адресна частина команди в разі сторінкової адресації складається з двох частин: початкової адреси сторінки в таблиці сторінок і зміщення щодо початку сторінки. Виконавчу адресу формує операційна система, складаючи початкову адресу таблиці сторінок, початкову адресу сторінки і зміщення щодо початку сторінки.

Сегментну адресацію використовують за сегментної організації пам'яті, коли пам'ять комп'ютера поділено на сегменти — частини пам'яті довільного розміру. Кожній програмі призначають свій сегмент, тобто початкову адресу сегмента і розмір сегмента. Початкова адреса сегмента міститься в спеціальному сегментному регістрі процесора. В адресній частині команди міститься зміщення щодо початкової адреси сегмента. Виконавча адреса формується додаванням початкової адреси сегмента до зміщення.

8.4. АСЕМБЛЕР

Мова *асемблера* (англ. assembly language) — це алгоритмічна мова вищого ієрархічного рівня, ніж рівень машинних команд. Кожну машинну команду замінюють на команду асемблера. Такий процес називають *асемблеюванням*, а комплекс програм, що здійснює таку заміну, — *асемблером* (англ. assembler). Оскільки кожен тип комп'ютера має свою систему машинних команд, мова асемблера також залежить від типу комп'ютера і тому є *машинно-орієнтованою*. Мова асемблера і мова машинних команд належать до мов низького рівня (англ. low-level programming language).

Необхідність переходу з рівня машинних команд на вищий рівень, тобто на рівень асемблера, було зумовлено такими чинниками. По-перше, машинні команди — це довжелезні колонки двійкових багаторозрядних чисел. Написати програму в машинних кодах — це важка, виснажлива і малопродуктивна робота програміста, оскільки потрібно пам'ятати двійкові коди машинних команд. Крім того, під час написання таких програм є надзвичайно великою ймовірність

помилок, які важко знайти і виправити. По-друге, дані, які обробляє процесор, — це також багаторозрядні двійкові числа, які зовні нічим не відрізняються від кодів машинних команд. Нарешті, адреси пам'яті — також двійкові числа, і програміст мусить пам'ятати, за якою адресою міститься та чи та інформація.

Щоб спростити створення програм і зробити програму зручною для читання і розуміння людиною, у мові асемблера коди команд, дані й адреси, виражені багаторозрядними двійковими числами, замінюють мнемонічними буквеними іменами — *мнемоніками*, тобто словами, які людині легко запам'ятати. Наприклад, команду додавання двох чисел, яка в мікропроцесорі Intel 8080A має двійковий код **10000000**, у мові асемблера замінюють буквеним виразом **ADD**, **B**. ADD — це скорочене англійське слово addition, яке означає математичну дію додавання і є буквеним виразом коду операції.

Крім заміни двійкових кодів операцій на мнемонічні, замінюють також адреси операндів у пам'яті комп'ютера. Наприклад, двійковий код регістра процесора **000**, у якому зберігається один із доданків (другий доданок стандартно зберігається в акумуляторі), замінюють у мові асемблера буквеним ідентифікатором регістра **B**. Двійковий код операції віднімання **10010** замінюють буквеним виразом **SUB** — перші три букви англійського слова subtraction — віднімання.

Крім основної функції — заміни машинних двійкових кодів мнемонічними буквеними виразами, мова асемблера має багато допоміжних засобів, які значно підвищують продуктивність роботи програміста. Одним із таких засобів є макрос — сукупність машинних команд, яка часто повторюється в програмі. Щоб кожного разу не писати частину програми, яка багаторазово повторюється, її виокремлюють у порівняно незалежний модуль, дають цьому модулю ім'я (ідентифікатор) і розташовують у пам'яті, а в програмі розміщують ім'я макросу, за яким викликається сам макрос. Застосування макросів не тільки значно зменшує обсяг пам'яті, а й робить програму на асемблері більш компактною і зрозумілою, що, своєю чергою, зменшує кількість помилок і спрощує їх виявлення.

ТЕСТОВІ ЗАВДАННЯ

МІКРОКОМАНДА — ЦЕ:

1. двійкове число, що зберігається в основній пам'яті комп'ютера і передається в процесор для виконання

2. двійкове число, що зберігається в керівній пам'яті процесора і керує пристроями процесора

3. двійкове число, що зберігається в основній пам'яті комп'ютера і керує пристроями введення–виведення

4. двійкове число, що зберігається в керівній пам'яті процесора і керує пристроями введення–виведення

Розділ 9
МОВИ ПРОГРАМУВАННЯ ВИСОКОГО РІВНЯ

Перші комп'ютери застосовували для виконання обчислень як великий калькулятор. Це була їхня перша «професія». Завдяки бурхливому розвитку електроніки, зокрема розробленню і масовому виробництву інтегральних мікросхем, різко зросла продуктивність комп'ютерів, значно зменшились їхні розміри, маса, енергоспоживання. Такі великі можливості дали змогу не обмежуватися лише обчисленнями, а застосовувати комп'ютери в різних *предметних галузях*, або сферах життєдіяльності людини.

У кожній предметній галузі функціонують відповідні система понять, ідей, методів і прийомів, термінологія. Поняття і терміни, якими оперує лікар, будуть незрозумілими, наприклад, для інженера і навпаки. Крім того, і лікареві, і інженеру, і науковцю потрібно знати мову машинних команд, щоб застосовувати комп'ютер у своїй царині. Існує так званий *семантичний розрив* (англ. semantic gap) між системою понять і термінів у предметній галузі й системою понять у галузі комп'ютерної техніки. Фахівцеві, замість зосередження на розв'язанні проблем у своїй сфері, потрібно думати, як цю проблему описати мовою машинних команд.

Для подолання такого семантичного розриву розробляють *мови програмування високого рівня* (англ. high-level programming language), призначені для підвищення продуктивності праці програміста за рахунок зменшення частки рутинної роботи і збільшення частки інтелектуальної, творчої роботи. Такі мови наближені до мов, якими спілкуються фахівці різних галузей, як-от інженери, економісти тощо. Мови високого рівня є *машинно-незалежними*, тобто програми цими мовами можна застосовувати на комп'ютерах різних типів. Крім того, їх ще називають *проблемно-орієнтованими*, оскільки ці мови зосереджені на розв'язанні задач (англ. problem), незалежно від того, на якому комп'ютері цю задачу виконуватимуть. Команди програми мовою високого рівня — це команди уявного, віртуального комп'ютера, для якого ці команди відіграють таку саму роль, як машинні команди для реальних комп'ютерів.

Мови високого рівня значно спрощують роботу програміста, дають змогу писати складні й великі за обсягом програми, однак такі програми виконуються на комп'ютерах повільніше, мають більший обсяг об'єктного коду, займають більше місце в пам'яті, ніж програми, написані мовою асемблера або мовою машинних команд. За весь період розвитку комп'ютерної техніки було створено понад дві з половиною тисячі мов високого рівня, і цей процес триває.

Перехід від оперування двійковими числами до оперування більш загальними поняттями в предметній галузі називають *абстрагуванням*. Використання *абстрактних*, тобто узагальнених, понять і об'єктів є головною характерною ознакою всього сучасного програмного забезпечення і мов програмування високого рівня зокрема.

9.1. ТРАНСЛЯЦІЯ ПРОГРАМ

Програми, написані мовами високого рівня, не можуть безпосередньо виконуватися процесором комп'ютера, їх потрібно «перекласти» на мову машинних команд, тобто подати у формі двійкових чисел. Перетворення програми, написаної мовою високого рівня, на програму низького рівня — це *трансляція* програми. Її здійснюють не вручну, а за допомогою комплексу спеціальних програм — *трансляторів*.

За способом перетворення програм транслятори поділяють на компілятори й інтерпретатори. *Компілятор* транслює програму в цілому, а потім програма надходить на виконання. *Інтерпретатор* перетворює (транслює) команди вхідної програми послідовно, одна за одною. Відтрансльована команда зразу ж виконується, а потім переходять до трансляції наступної команди. Перевагою інтерпретатора є відсутність попереднього оброблення і зручність у користуванні, особливо в діалоговому режимі. Недолік інтерпретатора — нижча, ніж у компілятора, швидкодія. Останнім часом спостерігається тенденція щодо об'єднання компілятора й інтерпретатора: спочатку програма компілюється в проміжний код (байт-код), який потім інтерпретується і виконується.

Етапи трансляції. Процес трансляції складається з кількох етапів (рис. 9.1). Програма мовою високого рівня, створена за допомогою *текстового редактора*, у формі текстового файлу надходить на вхід транслятора. Таку програму називають *вхідною*, або *вхідним кодом* (англ. source code). За необхідності програму можна також редагувати на цьому етапі.

Далі відбувається попереднє оброблення програми — *препроцесинг*. На цьому етапі виконують текстові вставки, взяті з системних бібліотек або створені програмістом, усувають коментарі тощо. Програму після попереднього оброблення часто називають *розширеним модулем*.

Наступний етап — компіляція, що також має кілька етапів. Спочатку здійснюють *синтаксичний аналіз* (англ. parsing) тексту програми відповідно до синтаксичних правил, прийнятих для цієї мови програмування. У процесі аналізу будують діаграму у вигляді дерева, на якій відображаються зв'язки між словами. Крім того, сучасні аналізатори здатні виявити синтаксичні помилки, допущені під час написання програми.

Після синтаксичного проводять *семантичний аналіз*, під час якого перевіряють відповідність типів змінних і виразів, коректність зв'язків між об'єктами програми тощо.

У деяких компіляторах передбачено *оптимізацію* коду, у процесі якої усувають надлишкові й невдалі конструкції, спрощують складні вирази.

На завершальній стадії компіляції генерується *об'єктний код*, тобто програма мовою низького рівня (мовою асемблера або мовою машинних команд), і формується *об'єктний модуль*.

Об'єктний модуль пов'язаний з іншими модулями, зокрема містить посилання на функції, або підпрограми, які розташовані поза модулем, наприклад у стандартних бібліотеках, тому наступним етапом є *компонування*. *Компону-*

вальник, або *редактор зв'язків* (англ. link editor, linker), — це спеціальна системна програма, яка зв'язує всі об'єктні модулі в одне ціле — *виконуваний модуль* (англ. executable file). Виконуваний модуль може виконуватися або апаратно за допомогою процесора, або програмно за допомогою віртуальної машини.

Для виконання програми виконуваний модуль потрібно завантажити в основну пам'ять. Цю операцію здійснює спеціальна програма — *завантажувальник* (англ. loader), який замінює відносні адреси й відносні посилання на реальні, зчитує вхідну інформацію і розміщує програму в основній пам'яті.

Завантажену програму можна запускати на виконання.

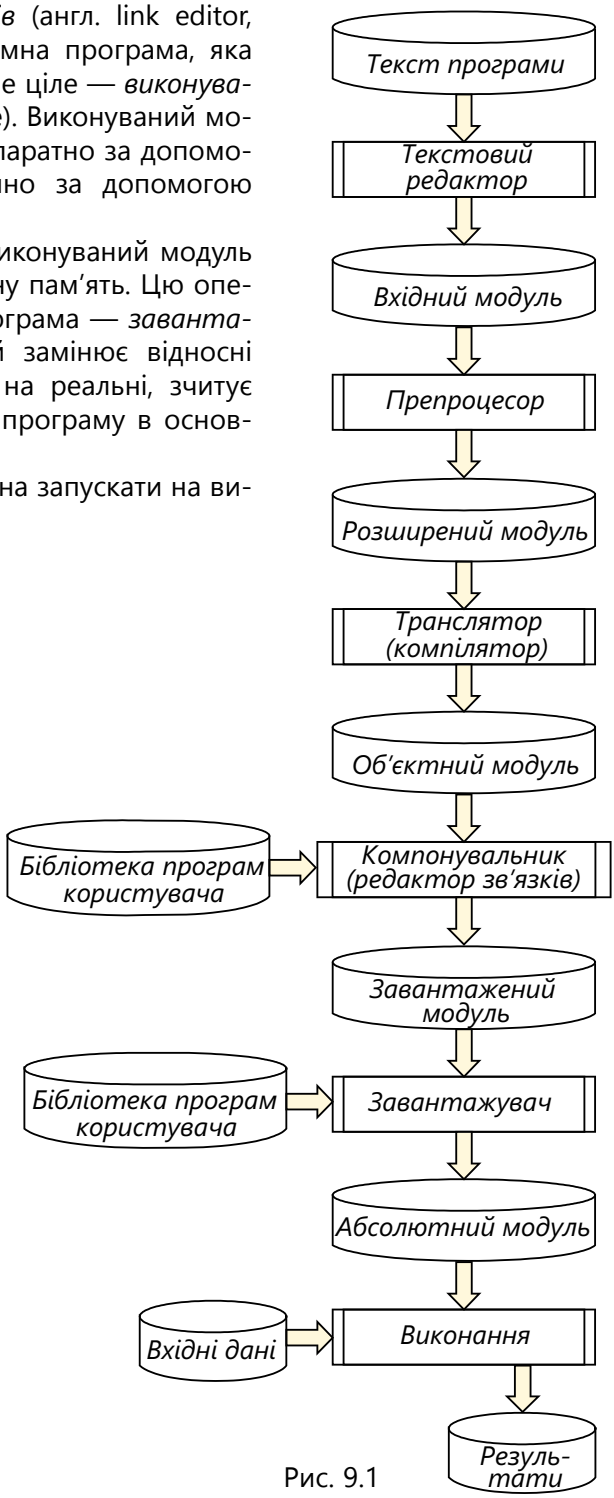


Рис. 9.1

9.2. ВИДИ МОВ ПРОГРАМУВАННЯ

Створення мов програмування високого рівня ґрунтується на **парадигмі програмування** — це система понять, ідей, підходів, методів, іншими словами, спосіб мислення програміста. Відповідно до певної парадигми програмування створюють мови програмування високого рівня. Розглянемо кілька найважливіших видів програмування.

В **імперативних мовах програмування** алгоритми оброблення інформації реалізуються у формі послідовності команд, кожна з них спонукає процесор (CPU) виконати певну операцію, тобто змінює *стан* програми. Імперативна мова програмування схожа на *наказовий спосіб* (англ. imperative mood) в природних мовах, який виражає накази або прохання виконати певні дії. Треба зазначити, що мова машинних команд та інші мови низького рівня за своєю суттю є імперативними, оскільки складаються з кодів команд, які виконує процесор.

Перша мова високого рівня FORTRAN (англ. FORmula TRANslation), яку створив Джон Бекус (John Backus) у 1954 р. на фірмі IBM, була імперативною. Ця мова була призначена для розв'язання задач у сферах науки й техніки і наближена до мови математики. Значення FORTRAN для розвитку комп'ютерної техніки важко переоцінити. Було вперше застосовано такі потужні програмні засоби, як підпрограми, процедури й функції, за допомогою яких створювали складні програми в різних прикладних галузях.

Для спрощення програмування математичних виразів було розроблено імперативну мову високого рівня ALGOL, хоча вона не мала такого широкого застосування, як FORTRAN. Мови COBOL (авторка — Грейс Гоппер (Grace Hopper)) і BASIC, розроблені відповідно у 1960 і 1964 рр., намагалися наблизити до природної англійської мови. Високорівневі мови набули подальшого розвитку в мові PASCAL, яку створив Ніклаус Вірт (Niklaus Wirth), і мові C, яку розробив Деніс Річі (Dennis Ritchie) у 1970-х.

Імперативні мови програмування високого рівня дають змогу обчислювати складні математичні вирази й функції, записані в їхній звичній формі. Обчислені значення цих виразів і функцій за допомогою команд присвоювання ставляться у відповідність змінним і зберігаються в пам'яті. Складні алгоритми оброблення інформації можна реалізувати в імперативних мовах програмування за допомогою команд розгалуження і циклу. Особливо велике значення має застосування в цих мовах *підпрограм* (англ. subroutine) — це сукупність команд, оформлена як одне ціле, порівняно автономне утворення. Підпрограму можна викликати з довільної точки основної програми, отримати вхідні дані й повернути результати своєї роботи. Окремим спеціальним видом підпрограм є *функції*. Ще однією позитивною якістю підпрограм є можливість багаторазового повторного використання (англ. reusability). З цією метою підпрограми доцільно зберігати в спеціально організованих *бібліотеках*.

Поділ програми на підпрограми, тобто *декомпозиція*, є поширеним прийомом, який дає змогу зменшити обсяг програм і спростити їхню структуру.

З головної частини такої програми в процесі роботи викликають підпрограми, тому таке подання програм називають *структурним програмуванням*. Крім структурування програмного коду, здійснюють також структурування даних.

Стиль програмування, за якого оброблення інформації трактують як послідовність окремих операцій із вхідною інформацією (даними), у результаті чого отримують вихідну інформацію, є *процедурним*. Треба зазначити, що програми в машинних кодах і мовою асемблера за своєю суттю є процедурними, оскільки складаються з послідовності машинних операцій, які виконує процесор комп'ютера.

Об'єктно-орієнтоване програмування. З розвитком комп'ютерної техніки з'явилася потреба у створенні складних програм великого обсягу. Для виконання такого складного завдання потрібен колектив програмістів. Важливо, що частини такої програми, написані програмістами різних кваліфікацій і в різний час, мають узгоджено працювати як одне ціле.

Ще однією проблемою є оновлення і підтримка впродовж терміну функціонування, відповідно до зміни умов і нових вимог. Бажано, щоб затрати на таке оновлення були мінімальними і стосувалися не програми загалом, а тільки певних її частин.

З ускладненням програм і збільшенням їхнього обсягу зростає ймовірність помилок, які важко виявити і виправити. З огляду на це було розроблено парадигму об'єктно-орієнтованого програмування, яку реалізовано у відповідних мовах програмування. В об'єктно-орієнтованому програмуванні використовують систему понять, яка ближча до понять предметної галузі, ніж до понять комп'ютерної техніки.

Перша мова, у якій було реалізовано принципи об'єктно-орієнтованого програмування, — мова Simula. Подальший розвиток ці принципи знайшли у мові Smalltalk. До об'єктно-орієнтованих належать такі популярні мови програмування, як C++, Object Pascal, Java, Python та багато інших.

За такого типу програмування програма являє собою сукупність *об'єктів* — порівняно автономних частин. Об'єкт в інформатиці — це абстрактна сутність, яка характеризується станом і поведінкою. *Стан* об'єкта визначає його властивості й залежить від даних, що містяться в об'єкті. *Поведінка* об'єкта окреслюється його діями і функціями, які називають *методами*. Іншими слова, стан визначає, «*який*» це об'єкт і що він собою являє, а поведінка — «*що він може робити*». Об'єкти в програмуванні мають адекватно відображати об'єкти матеріального світу в певній предметній галузі. Такий процес називають *абстрагуванням*, а об'єкт у програмуванні — абстрактним поняттям, *абстракцією*. До уваги беруть лише найважливіші параметри й характеристики реального об'єкта, а другорядні, що не стосуються певної проблеми, відкидають. Це дає змогу зосередитися на головному й не відволікатися на дрібниці.

Оброблення інформації об'єктно-орієнтованою програмою здійснюється в результаті взаємодії об'єктів за допомогою системи спеціальних засобів —

протоколу (англ. protocol), або *інтерфейсу* (англ. interface). Об'єкти можуть взаємодіяти один з одним, зокрема посилаючи один одному *повідомлення* (англ. message), лише через інтерфейс. Це захищає інформацію об'єкта від несанкціонованого доступу і допомагає уникати помилок.

Об'єкти одного типу утворюють *клас*, де кожен об'єкт є *екземпляром*. Класи формують складну ієрархічну структуру. Клас верхнього ієрархічного рівня називають *класом-предком*, а клас нижчого — *класом-нащадком*. Між класами різних ієрархічних рівнів існують відносини успадкування і генералізації. Якщо клас-нащадок зберігає певні властивості й атрибути класу-предка, то такий зв'язок називають *успадкуванням* (англ. inheritance). Відносини між класами, оберненими до успадкування, є *узагальненням*, або *генералізацією*. Вони показують, що певний клас є узагальненням для кількох класів.

Щоб захистити дані й методи об'єкта від несанкціонованого втручання, використовують механізм *інкапсуляції*. Це означає, що для повноцінного використання об'єкта не потрібно знати внутрішню будову і принцип роботи, а достатньо визначити його зовнішні властивості і його поведінку. Іншими словами, все, що міститься всередині об'єкта, закрито оболонкою.

Одним із основних понять об'єктно-орієнтованого програмування є *поліморфізм* (у перекладі з давньогрец. — «той, що має багато форм»). *Перевантаження функцій або методів* (англ. function overloading or method overloading) — вид поліморфізму, який дає змогу функціям із тією самою назвою здійснювати операції з різними типами даних, наприклад функція «перемножити» здійснює операцію перемножування як цілих чисел, так і чисел із рухомою точкою, і навіть матриць.

Шаблон (англ. template) функції — поліморфізм, який дає змогу задати функцію над абстрактним типом даних. Конкретний тип даних задають за виклику цієї функції.

Ще одним видом поліморфізму є застосування методів з однаковими іменами в різних класах. Це означає, що дії, які виконуються однойменними методами, можуть відрізнятися одна від одної, залежно від того, до якого класу належать ці методи.

У **декларативних мовах програмування** визначають, «*що*» в результаті роботи програми маємо отримати, але не описують, «*що робити*», щоб отримати цей результат. Декларативні мови програмування є альтернативою імперативних мов програмування.

Декларативний стиль програмування застосовують для того, щоб усунути або хоча б мінімізувати *побічний ефект* застосування функцій, який притаманний імперативним мовам програмування. Цей ефект полягає в тому, що крім обчислення значення функції в процесі виконання змінюється стан програми, тобто результат залежить не тільки від вхідних даних, а й від попередніх обчислень, тобто від передісторії.

До декларативних мов належать мови запитів до баз даних, мови функціонального і логічного програмування.

У **функціональному програмуванні** оброблення інформації зводиться до обчислення значень функції за значеннями аргументів. Функціональний стиль програмування відрізняється від імперативного, де обчислення значень функції трактують як покрокову зміну *стану* і базовим поняттям є *змінна*, яка змінює своє значення в процесі виконання алгоритму. В імперативному програмуванні функції залежать не тільки від своїх аргументів, а й від зовнішніх змінних, і тому значення функції з тими самими аргументами залежить від того, на якому етапі виконання алгоритму цю функцію було викликано.

Першою мовою функціонального програмування була мова Lisp, згодом було створено мови Scheme, APL, ML, Haskell.

Логічне програмування належить до декларативного програмування, тобто в ньому не використовують оператори присвоювання, розгалуження, циклу. Логічне програмування ґрунтується на апараті математичної (формальної) логіки. Найвідомішою мовою такого типу є Prolog (PROgramming LOGical).

Основним поняттям формальної логіки є *висловлювання* (англ. proposition). Висловлювання може бути *істинним* (англ. true) або *хибним* (англ. false). У формальній логіці розроблено механізм *виведення* (англ. logical deduction) нових висловлювань на основі обмеженої кількості заздалегідь істинних висловлювань (аксіом).

За логічного програмування здійснюють не послідовне виконання команд відповідно до алгоритму, а спроби вивести нове висловлювання на основі заданих істинних висловлювань і довести його істинність або хибність. Логічне програмування застосовують в аналізі природних мов, доведенні математичних теорем, системах штучного інтелекту тощо.

ТЕСТОВІ ЗАВДАННЯ

МОВА ПРОГРАМУВАННЯ ВИСОКОГО РІВНЯ — ЦЕ МОВА ДЛЯ ПРОГРАМУВАННЯ:

1. арифметичних виразів
2. економічних розрахунків
3. у різних предметних галузях
4. комп'ютерних ігор

ЩО ВИКОРИСТОВУЮТЬ ЯК КОМАНДИ МОВ ПРОГРАМУВАННЯ ВИСОКОГО РІВНЯ?

1. Слова природної мови
2. Слова алгоритмічної мови
3. Логічні вирази
4. Математичні вирази

Розділ 10
ОПЕРАЦІЙНІ СИСТЕМИ КОМП'ЮТЕРІВ

Операційна система сучасного комп'ютера — це комплекс спеціальних (системних) програм із різноманітними функціями та характеристиками, тому дати вичерпне визначення таких систем дуже важко.

10.1. КОРОТКА ІСТОРІЯ РОЗВИТКУ ОПЕРАЦІЙНИХ СИСТЕМ

Історія розвитку операційних систем нерозривно пов'язана з розвитком самої комп'ютерної техніки.

Комп'ютери першого покоління на електронних лампах проєктувала, створювала, програмувала й експлуатувала та сама група людей. Програмували такі комп'ютери машинними командами, і тому про операційні системи не могло йти й мови.

Комп'ютери другого покоління було побудовано на основі транзисторів. Значно збільшилася їхня надійність, зменшилися маса, габарити й енергоспоживання, збагатилося й ускладнилося програмне забезпечення. Програми для комп'ютерів створювали із застосуванням мов асемблера і мов високого рівня: мова FORTRAN для наукових та інженерних задач і мова COBOL для економічних і облікових задач. Написані програми набивали на перфокартах, зчитували з них за допомогою спеціального пристрою і надсилали на виконання. Результати роботи програми роздруковували на папері. Оскільки процес введення і виведення був значно тривалішим за процес виконання програми, то процесор часто простоював, чекаючи, поки завантажиться програма. Для економного використання машинного часу застосовували технологію *пакетного оброблення*. Кілька програм, нанесених на перфокарти, збирали в один пакет і вводили у комп'ютер. Крім програм до пакета входили програма-транслятор із мови високого рівня на рівень машинних команд і певна кількість *керівних перфокарт*. Ці перфокарти, що були прообразом операційних систем, давали команди процесору запустити програми на трансляцію, запустити відтрансльовану програму (об'єктний код) на виконання, вивести результати роботи програми. Прикладом такої примітивної операційної системи була FMS (Fortran Monitor System) для програм, написаних мовою FORTRAN.

Третє покоління комп'ютерів було побудовано на мікросхемах малого ступеня інтеграції. Щоб задовольнити вимоги в різних сферах застосування, почали випускати серії комп'ютерів, що мали однакове програмне забезпечення, однак різну продуктивність. Тобто написану програму можна було виконувати на будь-якому типі комп'ютера, що входив до серії, але з різною швидкодією. Прикладом такої серії комп'ютерів є IBM System/360 фірми IBM.

Для програмного забезпечення IBM System/360 було розроблено операційну систему OS/360, у якій реалізовано *багатозадачність* (англ. multitasking) — здатність забезпечувати виконання кількох програм одночасно. Необхідність такого режиму зумовлено тим, що швидкодія процесора набагато перевищує швидкодію пристроїв введення та виведення, і тому під час їх роботи процесор простоює. Особливо це характерно для економічних задач, у яких процесор виконує операції сортування інформації та інші нескладні математичні операції, тимчасом як операції введення і виведення займають основну частину робочого часу. Щоб розв'язати цю проблему, оперативну пам'ять поділяли на частини, у кожну з яких записували свою програму. Коли до однієї з програм вводили інформацію, інші програми по черзі могли використовувати процесор. Створювалася ілюзія, що кожна програма має свій, абстрактний процесор.

Крім багатозадачності було реалізовано також зчитування інформації з перфокарт у міру того, як звільнялися ділянки пам'яті. Такий технічний прийом дістав назву «*підкачка (свопінг)*» (англ. swapping).

Особливим видом багатозадачності є *система розподілу часу* (англ. time-sharing system). На відміну від пакетного режиму, де перемикання між задачами ініціює сама задача після її завершення, у системі розподілу часу кожній задачі відводиться свій проміжок часу, впродовж якого задача користується певним ресурсом комп'ютера. Систему розподілу часу було реалізовано на деяких комп'ютерах третього покоління. До таких потужних на той час пристроїв, які називали *мейнфреймами* (англ. mainframe), було під'єднано близько сотні користувачів через їхні термінали. Для цих комп'ютерів створили операційну систему MULTICS, яка забезпечувала роботу системи розподілу часу.

Ще однією важливою розробкою часів третього покоління були мінікомп'ютери, малопотужні, але дешеві, які мали широке застосування. Для мінікомп'ютерів розробили операційну систему UNIX, різні версії якої використовують дотепер. Щоб написані програми були придатні для будь-якої UNIX операційної системи, IEEE (Institute of Electrical and Electronics Engineers — Інститут інженерів електротехніки і електроніки) випустив стандарт POSIX, якого дотримуються більшість операційних систем UNIX.

Операційна система Linux багато в чому схожа на UNIX. Початкову версію Linux написав фінський студент Лінус Торвальдс (Linus Torvalds). Ця версія є безплатною, доступною для всіх охочих. Крім того, у неї відкритий вихідний код, що дає можливість вносити зміни. За весь період експлуатації Linux багато талановитих програмістів-ентузіастів постійно поліпшували систему. Зараз ця операційна система є однією з найефективніших, її широко застосовують у багатьох сферах.

Четверте покоління комп'ютерів — персональні комп'ютери — з'явилося завдяки прогресу у виготовленні інтегральних мікросхем. Почали масово випускати інтегральні мікросхеми надвисокого ступеня інтеграції (very-large-scale integration, VLSI). Це дало змогу розмістити на поверхні одного кристала не тільки окремі пристрої комп'ютера, а й весь процесор загалом. Такий про-

цесор дістав назву «*мікропроцесор*». На основі мікропроцесорів і мікросхем пам'яті надвисокого ступеня інтеграції почали випускати персональні комп'ютери за доступною для окремих осіб ціною. Перша операційна система, розроблена для персональних комп'ютерів на базі мікропроцесора 8080/8085 фірми Intel, — CP/M (Control Program for Microcomputers). Коли всесвітньо відома фірма IBM почала випуск персонального комп'ютера IBM PC, інша відома фірма Microsoft створила для нього операційну систему MS-DOS (Microsoft Disk Operating System). Команди в MS-DOS вводилися з клавіатури і відображалися у *командному рядку*.

Широке коло користувачів персональних комп'ютерів не мали комп'ютерної підготовки. Щоб зробити «дружнім» інтерфейс для таких користувачів, Стив Джобс (Steve Jobs) застосував *графічний інтерфейс* (graphical user interface, GUI) на персональному комп'ютері Macintosh (скорочено Mac), який випускала фірма Apple. Графічний інтерфейс використовував значки, піктограми, меню, що значно спрощувало роботу з персональним комп'ютером. Зважаючи на великий успіх у користувачів, фірма IBM почала використовувати цей інтерфейс для своїх персональних комп'ютерів. Спеціалісти фірми Microsoft розробили Windows як надбудову (англ. shell — «оболонка») над системою MS-DOS, яка в подальшому розвитку стала самостійною операційною системою із графічним інтерфейсом.

10.2. ПРИЗНАЧЕННЯ ОПЕРАЦІЙНИХ СИСТЕМ

Операційні системи виконують у комп'ютері дві головні функції. З одного боку, вони здійснюють керування пристроями комп'ютера, тобто апаратним забезпеченням, а також певним програмним забезпеченням під час виконання програм. З іншого, операційні системи забезпечують взаємодію користувача з апаратним і програмним забезпеченням, роблять його максимально простим, зручним і ефективним. Розглянемо докладніше ці функції.

Операційна система — менеджер ресурсів комп'ютера

Пристрої у складі сучасного комп'ютера мають складну внутрішню структуру, комплекс параметрів і характеристик, побудовані за різними принципами. Усі ці пристрої становлять апаратне забезпечення комп'ютера. Вручну керувати таким складним «господарством» дуже важко, а в багатьох випадках взагалі неможливо. Крім апаратного, потрібно також узгоджувати роботу програмного забезпечення. Це пов'язано з тим, що в сучасних комп'ютерах одночасно можуть виконуватися декілька програм. Потрібно розподілити між ними пам'ять, процесор та інші пристрої, унеможливити несанкціоноване втручання однієї програми в роботу іншої. Це завдання бере на себе операційна система, однією з основних функцій якої є керування апаратними і програмними засобами, які часто називають *ресурсами* комп'ютера. Операційна система є *менеджером* цих ресурсів.

Операційна система — посередник між користувачем і апаратним забезпеченням

Другою важливою функцією операційних систем є забезпечення взаємодії користувача з такими пристроями комп'ютера, як процесор, пам'ять, пристрої введення і виведення інформації. Операційна система як посередник безпосередньо взаємодіє, з одного боку, з пристроями, що входять до складу апаратного забезпечення, з іншого — з прикладними програмами користувача.

Усі ці пристрої є складними, основані на різних принципах і мають багато типів і модифікацій, які постійно змінюються в процесі розвитку. Наприклад, згідно із законом Мура продуктивність процесорів подвоюється кожні півтора року. Процесори різних виробників мають різні кількість регістрів, структуру і швидкодію. Щоб написати програму, програміст має докладно знати структуру процесора, кількість і особливості його регістрів, систему машинних команд. Програми для процесорів не є взаємозамінними. Аналогічні проблеми виникають, наприклад, для принтерів або зовнішніх пристроїв пам'яті. Принтери, зокрема, побудовані за різними фізичними принципами (лазерні, струменеві, матричні), мають різні структуру, функції і параметри. Створювати індивідуальні програми для такого різноманіття майже неможливо. Крім того, програми користувача здебільшого створюють мовами високого рівня, а команди керування фізичними пристроями є низькорівневими.

Щоб звільнити програміста від потреби знати деталі фізичної реалізації пристроїв апаратного забезпечення і дати йому змогу ефективно працювати з мовами високого рівня, операційна система бере на себе функцію безпосереднього керування фізичними пристроями. Деталі фізичної реалізації пристроїв приховані від програміста, і він може звертатися до них лише за посередництва операційної системи. Взаємодію операційної системи і пристроїв апаратного забезпечення забезпечують програмні й апаратні засоби, які називають *інтерфейсом апаратного забезпечення*.

Операційна система створює також взаємодію з прикладними програмами. Щоб забезпечити ефективне програмування мовами високого рівня, прикладні програми мають справу не з фізичними пристроями, а з узагальненими, *абстрактними об'єктами*, які часто називають просто *абстракціями*. Такі об'єкти є втіленням принципу *абстрагування*, який полягає в наданні можливості програмістові зосередитися на розв'язанні творчих задач вищого рівня у предметній галузі, а деталі конкретної реалізації перекласти на операційну систему. Наприклад, у прикладній програмі однією командою здійснюється запис результату операції в пам'ять, а на операційну систему покладається завдання — знайти ділянку пам'яті, у якій зберігаються дані цієї прикладної програми, далі знайти вільний сектор на диску, сформувати команду на переміщення головки на цей сектор, позиціонувати головку над потрібним сектором, сформувати команду запису і, можливо, перевірити правильність запису. Отже, замість запису на певну ділянку пам'яті на конкретному магнітному диску, програміст має справу із записом до абстрактного об'єкта — *файлу*.

Програмні й апаратні засоби взаємодії операційної системи з прикладними програмами становлять *інтерфейс прикладних програм*.

10.3. РЕЖИМИ РОБОТИ КОМП'ЮТЕРА

Програми, що входять до складу операційної системи, як і прикладні програми, виконуються процесором і зберігаються в тій самій основній пам'яті. Однак, на відміну від прикладних програм, програми операційної системи мають вищий *рівень привілеїв* (англ. privilege level). Зважаючи на це, розрізняють два режими роботи комп'ютера: *привілейований* (англ. privilege mode), у якому виконуються команди програм операційної системи, і режим роботи, у якому виконуються прикладні програми *користувача* (англ. user mode). Привілейований режим роботи називають також у певних операційних системах *захищеним режимом* (англ. protected mode), або *режимом супервізора* (англ. supervisor mode), а режим споживача — *реальним режимом* (англ. real mode). На апаратному рівні привілейований режим реалізується виконанням привілейованих операцій.

Найважливіші програми операційної системи, які завантажені в основну пам'ять і функціонують у привілейованому режимі, є *ядром* операційної системи (англ. operation system kernel).

Взаємодія між прикладними програмами й операційною системою реалізується за допомогою механізму *системного виклику* (англ. system call). Якщо прикладній програмі потрібен певний ресурс комп'ютера, вона надсилає ядру операційної системи запит. Процесор перемикається в привілейований режим роботи, і до роботи долучається та програма операційної системи, яка забезпечує обслуговування цього системного виклику. Завершивши роботу, операційна система передає результат прикладній програмі і перемикає процесор у реальний режим роботи. Прикладна програма далі продовжує свою роботу.

10.4. СТРУКТУРИ ОПЕРАЦІЙНИХ СИСТЕМ

Вимоги до операційних систем залежать від галузі застосування і призначення комп'ютерів. Відповідно до цих вимог змінюються структури операційних систем. Крім того, на розвиток структур великий вплив має прогрес у сфері апаратного і програмного забезпечення. Розглянемо найпоширеніші структури операційних систем.

У *монолітних* операційних системах усі функції зосереджено в ядрі. Системні програми операційної системи можуть викликати одна одну, щоб обслужити системний виклик прикладної програми, і це є перевагою монолітних операційних систем.

У процесі розвитку операційні системи комп'ютерів повсякчас ускладнюються. Кількість системних програм, що входять до сучасних операційних сис-

тем, досягає кількох тисяч. Розміщення всіх цих програм в ядрі робить монолітну операційну систему громіздкою. Зростає ймовірність несанкціонованого доступу однієї системної програми в роботу іншої. З огляду на це, монолітні операційні системи доцільно застосовувати в порівняно простих комп'ютерних пристроях.

Системні програми операційної системи можуть мати різні наслідки унаслідок помилок і несанкціонованого втручання. В одних системних програмах, наприклад, у програмі роботи з процесором, можуть статися катастрофічні наслідки, тимчасом як помилку в програмі роботи з принтером можна легко виправити.

З огляду на це програми операційної системи поділяють за ступенем захисту на кілька рівнів. Зі збільшенням рівня зростає обсяг привілеїв таких програм і забезпечується вищий ступінь захисту. Прикладні програми мають найнижчий рівень привілеїв і можуть звертатися до системних програм нижчого рівня. Якщо програмі нижчого рівня потрібно звернутися до програми вищого рівня, здійснюється виклик, подібний до системного виклику прикладних програм. Ретельно перевіряються права доступу і рівень привілеїв, і в разі позитивного результату програма вищого рівня надає потрібну послугу програмі нижчого рівня. *Багаторівневі* операційні системи мають вищий ступінь захищеності порівняно з монолітними. Однак продуктивність багаторівневих систем знижується через багатоступеневий виклик між рівнями.

Велика кількість програм в ядрі призводить до неефективності і слабкої захищеності операційної системи, тому останнім часом є дуже популярною концепція *мікроядра* (англ. microkernel). За цією концепцією в ядрі операційної системи реалізують лише невелику кількість таких системних програм, несанкціонований доступ або помилки в яких призводять до катастрофічних наслідків. До решти програм операційної системи здійснюється спрощений доступ.

У програмному забезпеченні сучасних комп'ютерів, як системному, так і прикладному, широко застосовують *емуляцію* — створення програми або комплексу програм, які б повністю відтворювали поведінку реального об'єкта. Емулювати можна пристрої апаратного забезпечення комп'ютера, наприклад принтери, сканери тощо. Програму, створену для друкування на одному типі принтера, можна друкувати на іншому фізичному принтері, створивши емуляцію цього пристрою.

Емулювати доцільно не тільки окремі пристрої комп'ютера, а і його операційні системи. Ба більше, на одному фізичному комп'ютері можна емулювати кілька операційних систем, і користувачі комп'ютера можуть одночасно працювати з різними операційними системами. Створюється ілюзія, що кожен користувач працює на своєму комп'ютері, хоча фізичний комп'ютер той самий. Такі уявні комп'ютери, якими користуються різні користувачі, називають *віртуальними машинами*.

10.5. ОСНОВНІ ФУНКЦІЇ ОПЕРАЦІЙНИХ СИСТЕМ

У процесі свого розвитку операційні системи пройшли шлях від примітивних програм, які виконували кілька допоміжних операцій, до сучасних комплексів зі складною ієрархічною структурою, з багатьма різноплановими функціональними можливостями. Функції операційних систем залежать також від їхнього призначення, наприклад функції операційної системи мобільного телефону відрізняються від функції операційної системи персонального комп'ютера і тим більше сервера. Розглянемо найзагальніші функції операційних систем широкого призначення.

Організація і керування виконанням програм

Виконання програм є однією з найважливіших функцій операційних систем. Щоб виконати програму на комп'ютері, потрібно надати певні ресурси — процесор на певний термін, ділянку пам'яті, пристрої введення і виведення, а також організувати їх взаємодію у процесі виконання програми.

Сучасні комп'ютери здебільшого багатозадачні, тобто вони одночасно виконують кілька програм. На деяких типах комп'ютерів, наприклад серверах, працюють кілька користувачів. Організовуючи роботу комп'ютера в цих умовах, потрібно з максимальною ефективністю розподілити ресурси комп'ютера між програмами.

Ще однією важливою обставиною є неоднакова швидкодія процесора, пам'яті та пристроїв введення і виведення. Під час одного звернення до пам'яті процесор може виконати десятки або сотні операцій. Своєю чергою, під час одного звернення до пристроїв введення і виведення, наприклад одного введення одного символу з клавіатури, процесор може здійснити десятки й сотні звернень до пам'яті.

Зважаючи на всі ці аспекти, можна констатувати, що в сучасних комп'ютерах програми майже ніколи не виконуються цілком, від початку до кінця, а поділяються операційною системою на частини, і ці частини виконуються в такому порядку, щоб забезпечити максимальну ефективність комп'ютера. Для визначення таких процесів широко використовують поняття «процес» і «потік».

Процес (англ. process) — це система дій пристроїв комп'ютера для виконання програми. Для кожного процесу, тобто для виконання програми, передбачено певні ресурси. По-перше, кожній програмі відведено окрему ділянку пам'яті, так званий окремий *адресний простір*, захищений від несанкціонованого доступу. По-друге, надано доступ до процесора на певний час. Зважаючи на важливість цього ресурсу, доступ до процесора надають не на весь термін виконання програми, а частинами, щоб забезпечити максимальне завантаження процесора і уникнення його простоїв. І, нарешті, передбачено доступ до пристроїв введення і виведення для введення вхідних даних і виведення результатів.

Із метою ефективного використання ресурсів комп'ютера, передусім процесора, доцільно процес поділити на частини — *потоки* (англ. thread — «нитка») — і виконувати їх паралельно. Потоки, на відміну від процесів, використовують спільний захищений адресний простір. У найпростішому випадку процес може складатися з одного потоку.

Із поняттями процесу і потоку тісно пов'язане поняття «*задача*» (англ. task) — програмний код, який має виконати процесор. Задача є одиницею роботи для процесора. У деяких операційних системах задачі об'єднують у більшу одиницю роботи — *завдання* (англ. job).

Дескриптор і контекст. Для керування процесами в операційній системі передбачено *дескриптор* (англ. description — «описувач»), що містить таку інформацію:

- ідентифікатор;
- тип процесу, який визначає правила надання певних ресурсів; рівень пріоритету процесу;
- інформація про стан, у якому перебуває процес;
- інформація про захищену ділянку пам'яті, відведену для процесу;
- засоби взаємодії з іншими процесами.

Дескриптори процесів входять до складу інформаційної структури, яка дає змогу операційній системі керувати процесами. Така структура має назву *блок керування задачею* (англ. task control block, TCB), або *блок керування процесом* (англ. process control block, PCB), і розміщується в оперативній пам'яті. У деяких операційних системах, наприклад у популярній Windows, для відкритих процесів відведено сегмент оперативної пам'яті — *сегмент стану задач* (англ. task state segment, TSS), а адреса цього сегмента зберігається в спеціальному регістрі процесора (англ. task register, TR).

Стани процесів і потоків. Оброблення інформації в мультипрограмних операційних системах — це сукупність процесів. Зрозуміло, що в комп'ютерній системі з одним процесором у кожний момент часу може виконуватися лише один процес, а паралельне оброблення здійснюється перемиканням процесора з одного процесу на інший. Поки один процес виконується, інші чекають своєї черги, отже процеси можуть перебувати в стані виконання і невиконання. Своєю чергою, якщо процес не виконується, то він може перебувати в стані очікування або в стані готовності. Отже, процеси й потоки можуть перебувати в таких станах:

- створення — операційна система готує ресурси для створення нового процесу;
- виконання — всі необхідні ресурси надано, зокрема найважливіший — процесорний час, і процесор за командами програми обробляє інформацію;
- очікування — необхідні ресурси надано, однак процес або потік очікує певної події, наприклад закінчення процедури введення або виведення даних, без яких неможливо перейти у стан виконання;

- готовність — необхідні ресурси надано, дані отримано, однак процесор зайнятий виконанням команд програми, яка була в черзі першою;
- завершення — операційна система завершує виконання цього процесу.

Перемикання контексту й оброблення переривань. Щоб ефективно використовувати процесор і уникати його простоїв, потрібно перемикати процесор з одного процесу на інший. Такий перехід називають *перемиканням контексту.*

Отже, один і той самий процес може кілька разів переходити з одного стану в інший, поки не завершиться. Переходи між станами процесів здійснюють за допомогою *системи переривань*, що є важливою частиною операційної системи і реалізується як апаратними, так і програмними засобами. *Переривання* — це призупинення виконання одного процесу і перехід до виконання іншого процесу. Процес переривання складається з кількох послідовних кроків.

Щоб отримати такий важливий ресурс, як процесорний час, процес формує спеціальний сигнал — *запит на переривання* (англ. interrupt request, IRQ) і надсилає його процесору. Запити на переривання можуть надходити від кількох процесів. Операційна система приймає і фіксує сигнали запитів і ідентифікує програми або пристрої, що їх надіслали.

Процес, який у цей час виконувався на процесорі, призупиняється. Щоб у подальшому відновити виконання цього процесу без втрат, потрібно зберегти інформацію про стан процесу на момент призупинення, а саме: вміст лічильника команд, який зберігає інформацію про адресу наступної команди, вміст регістрів процесора та інші цінні дані. Для цього керування передають *оброблювачу переривань* — спеціальній програмі операційної системи, яка реалізує процес переривання. Після закінчення роботи оброблювача переривань формується сигнал підтвердження переривання і керування передається програмі, що надіслала запит.

Відновлюється інформація про стан процесу, який був на момент попереднього переривання, тобто *контекст*, і програма продовжує виконуватися.

Планування процесів і потоків. У мультипрограмній системі, де одночасно відбуваються кілька процесів, операційна система має визначити, у якій послідовності виконуватимуться процеси, у який момент часу треба призупинити виконання одного процесу і відновити виконання іншого. Такі дії операційної системи, пов'язані з організацією паралельного виконання кількох процесів, називають *плануванням*.

В операційних системах використовують два види планування: *без витіснення* (англ. non-preemptive) і *з витісненням* (англ. preemptive). За планування без витіснення процес перебуває в стані виконання доти, доки він за власною ініціативою не передасть керування операційній системі, яка вибере з черги інший процес у стані готовності. Планування з витісненням — такий спосіб планування, за якого рішення про перемикання процесора з однієї задачі на іншу приймає операційна система.

За планування дотримуються таких критеріїв:

- справедливість — гарантувати кожному процесу або потоку надання певної частини процесорного часу, без монопольного «захоплення» процесора одним процесом;
- ефективність — звести до мінімуму простої процесора в очікуванні чергового процесу;
- тривалість очікування (англ. waiting time) — мінімізувати тривалість перебування процесів у стані готовності;
- тривалість відгуку (англ. response time) — мінімізувати інтервал часу від моменту надходження запиту від процесу до моменту надання йому процесорного часу.

Щоб забезпечити планування відповідно до цих суперечливих критеріїв застосовують різні алгоритми. Розглянемо найпопулярніші алгоритми планування.

За найпростішим алгоритмом *FCFS* (англ. first come, first served — «першим прийшов, першим обслуговується») здійснюється планування без витіснення. Процес, що отримав у своє розпорядження процесор, використовує його до свого завершення. Після цього для виконання вибирається наступний процес, що стоїть у черзі першим. Перевагою цього алгоритму є простота його реалізації. Недоліком є те, що середня тривалість очікування і середня тривалість виконання істотно залежать від послідовності розміщення процесів у черзі.

Алгоритм *RR* (англ. round robin — «дитяча карусель») належить до алгоритмів із витісненням. Якщо уявити, що всі процеси сидять на дитячій каруселі, що рівномірно обертається, то кожен процес певний інтервал (квант) часу перебуває поблизу процесора і на цей відтинок отримує його у своє розпорядження. Після закінчення цього інтервалу часу процесор перемикається на черговий процес. Ефективність алгоритму залежить від розміру кванта часу. Якщо квант часу сумірний або перевищує тривалість виконання окремого процесу, то цей алгоритм вироджується в алгоритм FCFS. Якщо ж квант часу істотно менший за середню тривалість виконання процесів, то створюється ілюзія, що кожному з n процесів надається окремий віртуальний процесор, продуктивність якого у n разів менша за продуктивність реального процесора. Недоліком цього алгоритму є можливість ситуації, коли сумарна тривалість перемикань процесора істотно перевищує сумарну тривалість оброблення інформації.

Алгоритм *SJF* (англ. shortest job first — «спершу найкоротша робота») організує чергу процесів так, що першими в черзі стоять процеси із найкоротшим терміном виконання. Це дає змогу зменшити кількість перемикань, оскільки короткотермінові процеси можуть взагалі обійтися без перемикань процесора. Недолік цього алгоритму полягає в тому, що в багатьох випадках складно оцінити тривалість виконання кожної роботи.

Взаємодія процесів і потоків. За паралельного виконання кількох процесів або потоків у комп'ютерній системі постає проблема взаємодії між

ними. Процеси й потоки поділяють на два види: *незалежні* (англ. independent processes), які не взаємодіють між собою, і *взаємозалежні* (англ. cooperating processes). Незалежні процеси не впливають один на одного і тому не створюють проблем. Паралельне виконання процесів і потоків, що взаємодіють між собою, має, з одного боку, забезпечувати раціональне використання ресурсів комп'ютера, а з іншого — унеможливлювати негативний вплив одного процесу на інший.

У багатьох випадках результат роботи програм за їх паралельного виконання залежить від послідовності процесів і потоків. Процес узгодження послідовності їх виконання в часі називають *синхронізацією*.

Взаємодію процесів і потоків зумовлено *спільним використанням* ресурсів комп'ютерної системи. Наприклад, потоки спільно використовують єдиний адресний простір. У спільному користуванні можуть бути також файли з даними, пристрої введення і виведення, як-от принтер, тощо.

За умови спільного використання ресурсів комп'ютера загострюється *конкурентна боротьба* процесів і потоків за доступ до певного ресурсу, і тому в операційній системі має бути механізм, щоб розв'язувати цю проблему. Операційна система, наприклад, може організовувати черги з урахуванням пріоритетів для доступу до ресурсів.

У конкурентній боротьбі за доступ до певного ресурсу може виникнути ситуація, яку називають «*змагання наввипередки*» (англ. race condition). Проблема полягає в тому, що операційна система надає процесам і потокам доступ до процесора, відповідно до критерію раціонального використання процесорного часу за прийнятим алгоритмом планування, тому моменти перемикання контексту і тривалість доступу дуже складно передбачити. Наприклад, два процеси або потоки спільно використовують інформацію, записану за певною адресою. Якщо перший процес, прочитавши інформацію, встигне записати за цією адресою нову інформацію, то результат роботи програми буде правильний. Якщо ж, після прочитання інформації, перший процес буде перервано й почне виконуватися другий процес, який за тією самою адресою запише свою інформацію, то це призведе до непоправних наслідків. Отже, правильність роботи програми залежить від відносної тривалості обох процесів, яка є випадковою величиною і яку складно передбачити.

Ділянку програмного коду, на якій може виникнути така аварійна ситуація, називають *критичною секцією* (англ. critical section). Щоб уникнути небажаних наслідків під час виконання критичних секцій програм, застосовують механізм *взаємного виключення* (англ. mutual exclusion), який унеможливлює доступ до ресурсів спільного користування інших процесів під час виконання критичної секції певного процесу. На практиці використовують багато способів взаємного виключення: змінні блокування, семафори, мютекси, монітори тощо.

Одним із негативних наслідків застосування взаємного виключення є *патова ситуація* (англ. deadlock), яка полягає в тому, що два процеси використовують два ресурси спільного користування і обидва перебувають на стадії

виконання критичних секцій. Перший процес блокує перший ресурс і перебуває в стані очікування, коли звільниться другий ресурс. Водночас другий процес блокує другий ресурс і очікує на звільнення першого ресурсу. Таке взаємне очікування двох процесів може тривати нескінченно довго, паралізуючи роботу всього комп'ютера.

Розв'язати проблему патових ситуацій можна такими способами:

- розпізнавання глухих кутів;
- запобігання патовим ситуаціям;
- відновлення системи;
- ігнорування.

Розпізнавання патових ситуацій є складним завданням, оскільки глухі кути треба відрізняти від черг, які часто виникають за спільного використання ресурсів. Розпізнавання ґрунтується на аналізі таблиць розподілу ресурсів і таблиць запитів до зайнятих ресурсів.

Щоб запобігти виникненню таких ситуацій, потрібно ретельно аналізувати умови виникнення й не допускати їх.

У разі патової ситуації можна зняти з виконання частину процесів, звільнивши ресурси, на які очікують інші процеси; повернути деякі процеси в буферну пам'ять; зробити «відкат», тобто повернути деякі процеси назад до контрольної точки, у якій здійснювалося збереження інформації з метою подальшого відновлення. Усі ці методи потребують додаткових затрат.

Якщо виникнення патових ситуацій є малоймовірним, збитки від них сумірні зі збитками від інших відмов апаратного і програмного забезпечення, а додаткові затрати на недопущення глухих кутів занадто великі, то їх можна просто ігнорувати.

Керування пам'яттю

Зі зростанням складності програм, зумовленим новими сферами застосування комп'ютерів, зростають і вимоги до таких параметрів пам'яті комп'ютерів, як обсяг, швидкодія, вартість. На сучасному етапі розвитку не існує такого виду пам'яті, який би цілком їм відповідав. Пам'ять сучасних комп'ютерів має складну ієрархічну будову, і на кожному рівні ієрархії застосовується свій тип пам'яті. Щоб ефективно використовувати пам'ять, операційна система має забезпечити адекватне керування цією складовою комп'ютера. Для цього передбачено комплекс програмних і апаратних засобів — *пристрій керування пам'яттю* (англ. memory management unit, MMU).

Віртуальна пам'ять — технологія керування пам'яттю в сучасних комп'ютерах для підтримки багатокористувацького і багатопрограмного режиму роботи. Ця технологія полягає в тому, що для написання прикладних програм використовують не адреси фізичних пристроїв пам'яті, а так звані логічні адреси, тобто адреси уявної, віртуальної пам'яті. Пам'ять реальних пристроїв пам'яті, що входять до апаратних засобів комп'ютера, називають *фізичною пам'яттю*. Роботу з перетворення логічних адрес на фізичні бере

на себе операційна система, закриваючи від програміста деталі фізичної організації пам'яті і звільняючи його від низькорівневої рутинної роботи.

Технологія віртуальної пам'яті створює ілюзію, що кожній прикладній програмі надано в розпорядження власну пам'ять, яка є незалежною і не перекривається з пам'яттю інших програм. Сукупність адрес цієї пам'яті утворює *адресний простір*, що є *неперервним*, тобто містить всі адреси без винятків від нульової до максимальної.

Фактично всі прикладні й системні програми користуються тією самою оперативною пам'яттю. Сукупний обсяг пам'яті, який потрібен усім прикладним і системним програмам, набагато перевищує обсяг фізичної оперативної пам'яті, тому в цій основній пам'яті міститься тільки незначна, найнеобхідніша на конкретний момент частина програм. Решта програм зберігаються на зовнішній дисковій пам'яті. Операційна система, а точніше — пристрій керування пам'яттю, переміщує програми частинами в оперативну пам'ять за потреби. На цей процес програміст не впливає.

Розглянемо найпоширеніші засоби і прийоми, які застосовують для реалізації технології віртуальної пам'яті.

Сторінкова організація пам'яті. Частина програми під час її виконання перебуває в оперативній (основній) пам'яті, а частина — у вторинній дисковій пам'яті. На пересилання інформації з дискової на оперативну пам'ять витрачається багато часу, оскільки тривалість доступу до дискової пам'яті на кілька порядків більша за оперативну. Для збільшення ефективності пам'яті загалом доцільно пересилати інформацію з дискової пам'яті не окремими байтами, а цілими блоками. Такі блоки інформації стандартного обсягу називають *сторінками* (англ. page). Наприклад, для 32-розрядних комп'ютерів обсяг сторінки дорівнює 4 кілобайти, тобто байт інформації на сторінці адресується 12-розрядною адресною шиною. У сучасних комп'ютерах застосовують також сторінки більшого обсягу.

Оперативну пам'ять також поділяють на частини — *сторінкові блоки*, або *фрейми* (англ. frame), обсяг яких дорівнює обсягу сторінок. Сторінки нумерують, тому адреса байта інформації складається з двох частин: номера сторінки і зміщення всередині сторінки. Інформація про початковий номер, права доступу та інші важливі дані про властивості сторінки розміщується в *спеціальних елементах пам'яті* (англ. page table entry, PTE), які, своєю чергою, зведені в *таблицю сторінок* (англ. page table). Таблиці сторінок займають значний обсяг оперативної пам'яті. Наприклад, для 32-розрядної адресної шини й обсягу однієї сторінки в 4 кілобайти для таблиць сторінок необхідно відвести $2^{32-12}=2^{20}=1$ мільйон байтів пам'яті. Щоб зменшити обсяг оперативної пам'яті, відведеної для таблиці сторінок, систему таблиць роблять двоступінчастою. Для цього сторінки з таблицями зводять у *каталог таблиць сторінок,* де зберігаються елементи пам'яті (англ. page directory entry, PDE). Кожен такий елемент містить інформацію про таблицю сторінок, так само, як елемент пам'яті *PTE* містить інформацію про сторінку. Адреса каталогу таблиць сторінок міститься у спеціальному регістрі.

Двоступінчаста організація таблиць сторінок дає змогу зменшити обсяг пам'яті для них. Водночас це збільшує тривалість доступу до пам'яті, оскільки за потрібною інформацією потрібно тепер звертатися до пам'яті тричі: перший раз — до каталогу сторінок, другий — до таблиці сторінок, і лише третє звернення дає змогу отримати з пам'яті потрібний байт інформації.

Щоб зменшити тривалість доступу до оперативної пам'яті за сторінкової організації, застосовують *буфер асоціативної пам'яті* (англ. translation look aside buffer, TLB). Це пам'ять невеликого обсягу, але великої швидкодії, у якій зберігаються адреси, ймовірність виклику яких є значною. Перш ніж звернутися до оперативної пам'яті, процесор звертається до буфера. Якщо адреса потрібної сторінки вже міститься в буфері, то за цією адресою здійснюється безпосереднє звернення до пам'яті, без витрат часу на звернення за адресою каталогу і за адресою таблиці сторінок. Якщо потрібна адреса міститься в буфері, таке звернення називають *влученням* (англ. hit). За невдалого звернення, якщо потрібної адреси в буфері немає (англ. missing), генерується помилка звернення і запускається повна процедура визначення потрібної адреси оперативної пам'яті, відтак ця адреса заноситься в буфер. У сучасних комп'ютерах застосовують ефективні алгоритми заповнення буфера, які дають змогу довести кількість вдалих звернень від сотень до тисячі.

Сегментна організація пам'яті. За сторінкової організації пам'яті вся пам'ять поділена на частини фіксованого розміру — сторінки, але має єдиний адресний простір, тобто нумерація всіх елементів пам'яті є неперервною — від нульового до максимального. Якщо в основній пам'яті розміщено кілька програм, що виконуються процесором, доцільно поділити віртуальну пам'ять на *сегменти* — частини змінного розміру, які мають власний адресний простір. Різні види інформації — код програми, дані, стек — доцільно розміщувати в різних сегментах, надаючи їм різні права доступу. Крім того, різні програми матимуть свій набір сегментів.

Із кожним сегментом пов'язаний елемент пам'яті — *дескриптор*, у якому зберігається інформація про цей сегмент: його номер, розмір, права доступу тощо. Дескриптори сегментів зводять у таблиці, які зберігаються в основній пам'яті: *глобальну* (англ. global descriptor table, GDT) і *локальну* (англ. local descriptor table, LDT). У глобальній таблиці зібрано дескриптори сегментів, спільні для всіх програм. Локальна таблиця дескрипторів може бути в кожної програми. Адреси глобальної і локальних таблиць дескрипторів містяться в спеціальних системних регістрах процесора.

Логічна адреса за сегментної організації складається з двох частин: номера сегмента і зміщення адреси всередині сегмента. За номером сегмента із відповідної таблиці вибирається дескриптор сегмента, у якому міститься інформація про початкову адресу сегмента, його розмір, права доступу. За цією інформацією здійснюється перевірка кожного запиту на доступ до пам'яті. Якщо всіх вимог дотримано, доступ до потрібного сегмента дозволяється і видається адреса цього сегмента. Якщо ж ні — генерується помилка доступу

й операційна система обробляє цю помилку. За адресою сегмента і зміщенням усередині сегмента формується повна адреса потрібної інформації.

Сегментно-сторінкова організація пам'яті. Оскільки сегментна і сторінкова організації пам'яті мають свої переваги і недоліки, у багатьох сучасних операційних системах, зокрема у Windows, ці два підходи об'єднують. За номером сегмента і зміщенням формується лінійна адреса, яка далі за допомогою сторінкового механізму адресації перетворюється на фізичну адресу.

Уведення і виведення інформації

Для керування пристроями введення і виведення інформації на рівні двійкового коду потрібно докладно знати будову і принципи дії цих пристроїв. Праця програміста під час створення таких програм була важкою і малопродуктивною. Програми були машинозалежними, тобто в разі зміни пристрою треба було змінювати всю програму. Програми керування пристроями введення–виведення у двійкових кодах, створені висококваліфікованими програмістами, почали заносити до спеціальних системних бібліотек. Це дало змогу не писати цей код у кожній програмі заново, а тільки звертатися до готової програми. Саме введення і виведення інформації стали однією з основних причин створення операційних систем.

Пристрої уведення і виведення мають різноманітні характеристики й параметри:

- швидкість обміну інформацією змінюється в діапазоні від кількох байтів за секунду для клавіатури до кількох гігабайтів у секунду для мережних або відеокарт;
- одні пристрої уведення і виведення можуть використовуватися прикладними програмами паралельно, а інші — лише однією програмою;
- не всі пристрої мають здатність запам'ятовувати інформацію;
- за здатністю обмінюватися інформацією пристрої поділяють на *блочні*, які обмінюються цілими масивами (блоками) інформації, і *символьні*, що передають інформацію по одному байту послідовно;
- є пристрої, призначені лише для уведення інформації або лише для її виведення; є й пристрої, які можуть і вводити, і виводити інформацію.

Система введення–виведення інформації складається з апаратного і програмного забезпечення.

Контролери пристроїв уведення і виведення. Одним із найпоширеніших пристроїв, що входять до складу апаратного забезпечення, є *контролер* (англ. controller) — електронний цифровий пристрій, виконаний або у вигляді мікросхеми, або як окрема частина материнської плати і призначений для керування пристроями уведення і виведення. Контролер бере на себе здійснення низькорівневих операцій керування пристроями уведення і виведення і звільняє програміста від необхідності ними займатися. Для взаємодії із системними програмами операційної системи до складу контролерів входять такі регістри:

- регістр вхідної інформації, у який записується інформація, призначена для виведення;
- регістр вихідної інформації, у який записується інформація, призначена для введення;
- регістр стану, у який записується інформація про стан пристрою, наприклад готовність до операції уведення або виведення, зайнятість пристрою, наявність помилок при здійсненні операції тощо;
- регістр керування, у якому записується керівна інформація, задаються режими роботи тощо.

Перед здійсненням операцій уведення і виведення контролер програмується операційною системою на потрібний режим роботи через регістр керування. Далі читається інформація з регістру стану і перевіряється готовність пристрою виконувати операції уведення і виведення. Якщо пристрій готовий, то до регістру вхідної інформації записується потрібна інформація. Контролер формує команди керування, під дією яких пристрій здійснює операцію введення або виведення. Під час цього процесу до регістру стану заноситься інформація про зайнятість пристрою. Після завершення операції інформація про зайнятість скидається і встановлюється інформація про готовність пристрою до наступної операції.

Робота з пристроями уведення і виведення здійснюється у двох режимах: режимі опитування і режимі переривань.

Режим опитувань. Опитування пристрою уведення або виведення (англ. polling) здійснюється у такій послідовності:

- процесор циклічно читає інформацію з регістру стану контролера і перевіряє зайнятість пристрою, яка засвідчується встановленням одиниці біта зайнятості в регістрі стану. Якщо біт зайнятості встановлено, то це означає, що пристрій ще не завершив попередню операцію, і процесор через певний час здійснює чергове опитування. Після завершення операції біт зайнятості скидається, пристрій готовий до виконання наступної операції;
- перевіривши, що біт зайнятості скинуто, процесор записує код команди до регістру керування, а дані — до регістру вхідних даних. Після цього процесор встановлює біт готовності команди і переходить до виконання інших робіт;
- перевіривши, що біт готовності команди встановлено, контролер встановлює біт зайнятості й переходить до виконання команди введення або виведення;
- після завершення операції контролер скидає біт готовності команди;
- якщо операція завершилася успішно, то контролер скидає біт помилки, а якщо невдало — встановлює цей біт; на останньому кроці контролер скидає біт зайнятості.

Режим переривань. Режим опитування доцільно застосовувати, якщо швидкодія процесора і пристрою уведення або виведення сумірні. У більшості випадків ці швидкодії розрізняються на кілька порядків, тому в режимі опи-

тування процесор багато часу простоюватиме, циклічно опитуючи повільний пристрій. Для ефективного використання процесорного часу в сучасних комп'ютерах застосовують систему *переривань* — призупинення роботи процесора з однією програмою і перехід на виконання іншої.

Система переривань складається з апаратної і програмної частин. Основним пристроєм апаратної частини є *контролер переривань*. Якщо пристрій уведення або виведення під час роботи процесора має переслати чергову порцію інформації, формується сигнал — *запит на переривання*. Цей сигнал надходить на контролер переривання, який з'єднаний з усіма пристроями введення і виведення сигнальними лініями, якими надходять запити. Відповідно до ступеня терміновості обслуговування пристрої уведення і виведення поділяють за *рівнем пріоритетності*.

Прийнявши черговий запит на переривання, контролер ставить цей пристрій у чергу на обслуговування і перевіряє рівень пріоритетності. Якщо рівень пріоритетності прийнятого запиту нижчий за поточний, то пристрій переходить у режим очікування обслуговування. Якщо ж він вищий — обслуговування поточного запиту призупиняється і контролер формує сигнал переривання, який надходить на процесор. Процесор аналізує сигнал переривання, і якщо він дозволений (не замаскований), формує сигнал підтвердження переривання, який надходить на контролер. Після цього процесор читає інформацію з регістру контролера і визначає, який саме пристрій надіслав запит. Таку інформацію називають *вектором переривань*.

Визначивши, який пристрій надіслав запит, процесор розпочинає обслуговування запиту. Для обслуговування пристроїв уведення і виведення в режимі переривань у складі операційної системи є системне програмне забезпечення — *оброблювачі* переривань. Після завершення роботи оброблювача переривань процесор повертається до перерваної програми і продовжує роботу. Тож у режимі переривань процесор не витрачає свій час на очікування, а виконує свою роботу з іншою програмою.

Прямий доступ до пам'яті. У режимах опитування і переривань процесор бере участь у пересиланні інформації між пристроями введення–виведення і пам'яттю. Враховуючи те, що швидкодія оперативної пам'яті є набагато меншою, ніж швидкодія процесора, останній витрачатиме багато часу на очікування, особливо в разі пересилання інформації великими масивами. Для звільнення процесора від цієї малопродуктивної роботи в сучасних комп'ютерах застосовують режим прямого доступу до пам'яті (англ. direct memory access, DMA). У цьому режимі функцію пересилання інформації між пристроями введення–виведення і пам'яттю покладено на спеціальний процесор — *контролер прямого доступу до пам'яті*. Він захоплює шину даних і здійснює пересилання масивів інформації. Центральний процесор на цей час відключений від шини даних і може виконувати команди, не пов'язані з доступом до пам'яті.

Програмне забезпечення системи введення і виведення інформації є частиною операційної системи і має складну ієрархічну будову. Нижній рівень

цього програмного забезпечення посідають *драйвери* — програми, які безпосередньо керують апаратною частиною пристроїв уведення і виведення. Апаратна частина складається з контролера пристрою і самого пристрою уведення–виведення. Тип драйвера залежить від призначення відповідних пристроїв. Наприклад, для магнітного диска потрібно враховувати сектори, доріжки, циліндри магнітного диска, керувати електродвигунами для обертання диска і переміщення магнітної головки; для миші потрібно визначати, яку кнопку натиснуто, на яку відстань і у якому напрямі переміщено пристрій тощо.

З іншого боку, потрібно забезпечити уніфіковану взаємодію, тобто уніфікований інтерфейс між драйвером і вищим ієрархічним рівнем операційної системи. Зважаючи на це, розроблення драйверів покладено на виробників пристроїв уведення і виведення.

Базова система введення і виведення — комплекс системних програм операційної системи, що здійснюють взаємодію між прикладними програмами і драйверами:

1. *Уніфікований інтерфейс для всіх пристроїв уведення і виведення.* Якби операційна система з кожним пристроєм уведення і виведення взаємодіяла по-різному, то для кожного нового пристрою або його модифікації доводилося би вносити зміни до операційної системи. Щоб уникнути цього, для кожної групи пристроїв уведення і виведення визначають набір функцій, який має підтримувати драйвер цього пристрою. Виробники пристроїв, розробляючи для них драйвери, враховують ці вимоги з боку операційних систем. У результаті інтерфейс пристроїв стає уніфікованим, тож пристрої можна під'єднувати до комп'ютера без будь-яких змін в операційній системі.

2. *Буферизація.* Обмін інформацією між прикладними програмами і пристроями уведення–виведення призводить до значних простоїв процесора і затрат часу на багаторазове переривання роботи прикладних програм, зумовлене низькою швидкодією пристроїв уведення і виведення порівняно з процесором. Якщо до складу операційної системи увести невелику за обсягом пам'ять — буфер, то процесор може виконувати свою роботу, допоки буфер повільно заповнюється даними з пристрою уведення і виведення під керуванням драйвера. Після заповнення буфера вмикається процесор і на великій швидкості здійснюється пересилання інформації.

3. *Повідомлення про помилки.* З усіх пристроїв комп'ютера саме під час роботи пристроїв уведення і виведення найчастіше трапляються помилки і збої. За наслідками помилки можна поділити на кілька видів, зокрема такі, що можуть призвести до фатальних наслідків, й такі, що не становлять серйозної загрози і їх можна виправити. У будь-якому випадку помилку потрібно виділити і визначитися з наступними діями. Деякі типові помилки може виправити драйвер пристрою, а якщо ні — рішення переадресовується вищим рівням операційної системи.

4. *Розподіл пристроїв уведення і виведення між прикладними програмами.* Такі пристрої є спільним ресурсом для кількох прикладних і системних

програм, які одночасно виконуються на сучасних комп'ютерах. Для раціонального використання цього ресурсу операційна система має спланувати доступ до нього з урахуванням особливостей пристроїв. У багатьох випадках це реалізується за допомогою черг із прикладних програм з урахуванням їхнього рівня пріоритету.

Безпека і захист інформації

У сучасному суспільстві інформація є одним із найважливіших ресурсів. Загальною тенденцією стає глобальний доступ до інформації у масштабах планети. У наш час через глобальні комп'ютерні мережі стала доступною інформація про події в будь-якому куточку планети, зокрема студенти можуть ознайомитися з розкладом занять і програмою навчання провідних університетів світу. За такої доступності інформації гостро постає питання про її захист. У комп'ютерах безпеку і захист інформації покладено на відповідну систему, яка є невід'ємним складником операційної системи комп'ютера.

Система захисту інформації має забезпечувати такі можливості:

- *конфіденційність* (англ. confidentiality) — здатність приховати інформацію від стороннього доступу. Доступ до інформації має лише той, хто має на це відповідні права, — *права доступу.* Для забезпечення конфіденційності інформацію шифрують методами криптографії;
- *цілісність* (англ. integrity) — спроможність системи захистити інформацію від вилучення або несанкціонованої зміни;
- *доступність* (англ. availability) — здатність забезпечити доступ до інформації тим, хто має на це відповідні права.

Будь-яку дію, спрямовану на порушення конфіденційності, цілісності й доступності інформації, а також нелегальне, несанкціоноване використання інших ресурсів називають *загрозою*. Реалізована загроза — це *атака*.

Для забезпечення захисту інформації і підтримання належного рівня безпеки застосовують різноманітні програмні й апаратні засоби. Розглянемо найпоширеніші з них.

Шифрування (англ. encryption) — це технологія перетворення зрозумілого повідомлення на незрозуміле, недоступне для сторонніх. Зворотне перетворення — це *дешифрування* (англ. decryption). Шифроване повідомлення передають відкритими каналами зв'язку, отримувач приймає його і дешифрує. Шифрування і дешифрування об'єднують загальним поняттям «*криптографія*». Алгоритм, за яким здійснюють шифрування і дешифрування, є *криптографічним*. Для успішного застосування такого алгоритму потрібно мати *ключ* (англ. key) — спеціальну інформацію, здебільшого секретну. Ключ є найважливішим параметром криптографічного алгоритму, який визначає його *стійкість*.

На практиці використовують два види ключів: *симетричні*, або *секретні*, й *асиметричні*, або *відкриті*.

У криптографічних алгоритмах із симетричним ключем застосовують той самий ключ як для шифрування інформації, так і для її розшифрування. Такий

ключ потрібно зберігати в таємниці й передавати секретними, закритими каналами зв'язку. Для підвищення стійкості криптографічних алгоритмів потрібно частіше змінювати ключі. Інтенсивне використання закритих, дорогих каналів зв'язку для передавання симетричних ключів значно збільшує затрати, що є одним із найбільших недоліків симетричних ключів.

Криптографічні системи з асиметричними ключами використовують два типи ключів: відкритий, несекретний — для шифрування інформації, може передаватися відкритими каналами зв'язку; закритий, секретний — для дешифрування.

У криптографічних системах підтримання безпеки широко застосовують також *односторонні функції шифрування,* або *хеш-функції, дайджест-функції.* Якщо вжити хеш-функцію до повідомлення, то отримаємо зашифроване повідомлення — *дайджест*, яке передають відкритими каналами зв'язку разом із незашифрованим повідомленням. Отримавши повідомлення разом із дайджестом, користувач застосовує таку саму секретну хеш-функцію до незашифрованого повідомлення і порівнює отриманий дайджест із переданим. Якщо дайджести однакові, це означає, що передане повідомлення не було змінено.

Автентифікація (англ. authentication) — процедура встановлення достовірності, тобто доказ того, що користувач є справжнім і тим, за кого він себе видає. Автентифікація запобігає доступу до системи небажаних осіб і дозволяє вхід легальним користувачам. У цьому процесі беруть участь дві сторони: одна доводить свою автентичність, надаючи докази, а інша — перевіряє ці докази. Доказом автентичності можуть бути різні матеріальні й нематеріальні речі, але найпростішим і логічним способом є *паролі*. Недоліком системи паролів є той факт, що складно зберегти баланс між простотою і зручністю пароля і його надійністю. Паролі можна вгадати, підібрати, випадково розкрити або нелегально передати стороннім особам. Одним із можливих способів угадування пароля є перебір усіх можливих комбінацій букв, цифр, розділових знаків — так звана словникова атака. Звісно, вручну це зробити неможливо, однак за допомогою сучасних комп'ютерних засобів із гігантською продуктивністю можна перепробувати всі можливі варіанти за прийнятний час.

Авторизація доступу — це процедура контролю доступу легальних користувачів до ресурсів комп'ютера, за якої кожному легальному користувачеві надають саме ті права на окремі види ресурсів, які для нього визначив адміністратор. Розрізняють вибірковий і мандатний способи керування доступом.

За *вибіркового* способу певні операції з конкретним ресурсом (об'єктом) дозволяють або забороняють суб'єктам або групам суб'єктів: стан прав доступу описується матрицею, у рядках якої розміщені суб'єкти, у стовпчиках — об'єкти, а в клітинках матриці — операції, які суб'єкт може виконувати з об'єктом. У таких матрицях багато клітинок будуть пустими, тому на практиці застосовують рідко. Якщо таку матрицю розкласти по стовпчиках, то кожен стовпчик буде *списком прав доступу* (англ. access control list, ACL), який буде

компактнішим за матрицю, оскільки пусті клітинки не потрібно додавати до списку.

Мандатний спосіб керування доступом полягає у тому, що всі об'єкти поділяють на групи за *рівнем секретності*, а всіх суб'єктів — за *рівнем допуску* до секретної інформації.

Аудит систем захисту — це відслідковування і фіксація подій, пов'язаних із доступом до захищених ресурсів. Аудит дає змогу виявити спроби вторгнень і слабкі місця в системі захисту.

ТЕСТОВІ ЗАВДАННЯ

ЩО ТАКЕ ОПЕРАЦІЙНА СИСТЕМА?

1. Комплекс програм, що дають змогу користувачеві спілкуватися з комп'ютером, керувати пристроями комп'ютера, програмами та інформацією, що зберігається в пам'яті комп'ютера
2. Комплекс програм, що організовують роботу з файлами та папками
3. Комплекс програм, які забезпечують роботу в Інтернеті
4. Комплекс програм, які оперативно виконують запити користувачів

ЯКІ ОСНОВНІ ФУНКЦІЇ ВИКОНУЄ ОС?

1. Опрацювання графічної інформації
2. Оброблення текстової інформації
3. Керування програмами, встановленими на комп'ютер
4. Керування роботою пристроїв комп'ютера
5. Забезпечення діалогу користувача та комп'ютера

Розділ 11
ОБРОБЛЕННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ. ПРОГРАМА Microsoft Word

Застосування інформаційних технологій для створення, оброблення, передавання на відстань і подання текстової інформації має велике значення. Нині триває етап активного впровадження *безпаперових технологій*, що передбачає, зокрема, створення і функціонування документів в електронній формі.

11.1. ТЕКСТОВА ІНФОРМАЦІЯ

Текстова інформація є одним із найпоширеніших видів інформації. Від часу винайдення друкування і донедавна основним носієм текстової інформації був *папір* (книжки, газети, журнали тощо).

Основною одиницею текстової інформації є *документи*, наприклад стаття в газеті чи журналі, доповідь, замітка, лекція, літературний твір (оповідання, вірш), лист тощо. Документ, створений за допомогою комп'ютерних засобів, називають *електронним документом*. Він зберігається в комп'ютері у вигляді *текстового файлу.*

Найменшим неподільним елементом текстової інформації є символ. Символи поділяють на такі види, як букви, розділові знаки, цифри, спеціальні символи.

Основними символами є *букви (літери)*. Сукупність букв, які вживають у певній мові, називають *азбукою*, *алфавітом* або *абеткою*. Алфавіт української мови складається з 32 букв і базується на *кирилиці* — азбуці, створеній великими слов'янськими просвітителями Кирилом і Мефодієм. Більшість народів світу використовують *латиницю* — азбуку, винайдену у Стародавньому Римі.

Для зв'язку слів у реченні застосовують такі символи, як розділові знаки: крапка, кома, крапка з комою, знак оклику, знак запитання тощо. Числа на письмі передають за допомогою *цифр*.

У деяких текстах, зокрема науково-технічних, широко застосовують *спеціальні символи*: букви грецького алфавіту, знаки математичних операцій, спеціальні математичні символи тощо.

Із букв складається більша структурна одиниця тексту — *слова*. На письмі слова відокремлюють *пробілами*. Послідовність слів утворює *речення*. Декілька речень, пов'язаних за змістом, формують *абзац*.

За традицією, текст розбивають на *сторінки*.

Кодування текстової інформації. Як уже зазначено, текстову інформацію для оброблення її комп'ютером потрібно подати у двійковому вигляді — *закодувати*.

До широкого застосування комп'ютерів також застосовували кодування символів, зокрема букв алфавіту, наприклад код *Бекона*, код *Брайля* для сліпих, інтернаціональний морський код за допомогою прапорців, код *Морзе* тощо.

Найбільшого поширення для кодування текстової інформації в комп'ютерах набув код *ASCII* (American Standard Code for Information Interchange — Американський стандартний код для обміну інформацією), який було впроваджено на практиці в 1963 р. Це 7-розрядний двійковий код для кодування великих і малих літер латиниці, цифр, розділових знаків та деяких керівних символів для пристроїв, призначений лише для англомовних користувачів.

У 1991 р. некомерційна організація *Консорціум Юнікоду* представила розроблений стандарт кодування символів *Юнікод* (англ. Unicode), який забезпечує цифрове подання символів усіх писемностей світу та спеціальних символів. На кодування одного символу відводиться від 1 до 4 байтів, причому перший байт збігається з кодом ASCII.

Сучасні комп'ютерні засоби виконують такі види *оброблення текстової інформації*:

- введення тексту за допомогою клавіатури або з паперового оригіналу за допомогою сканера з подальшим розпізнаванням тексту спеціальними програмами, збереження його в пам'яті;
- редагування тексту (виправлення, заміна, копіювання, переміщення, вилучення тощо);
- попередній перегляд перед друком і друкування;
- редагування декількох документів одночасно;
- форматування тексту (оформлення зовнішнього вигляду тексту, зміна його параметрів);
- перевірка правопису;
- використання графічних зображень у тексті;
- використання таблиць у тексті;
- використання макросів (програмних кодів) у документах.

11.2. ЗАСОБИ ОБРОБЛЕННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ

Для оброблення текстової інформації широко застосовують різноманітні програмні засоби, які можна поділити на три категорії: текстові редактори, текстові процесори і настільні видавничі системи. Розглянемо їх докладніше.

Текстові редактори (англ. text editors) призначені для створення невеликих простих документів (англ. plain text) із розширенням **.txt.** Для їх створення використовують ASCII-символи без форматування. На практиці широко застосовують прості текстові редактори *Блокнот* і *WordPad* (які входять до складу операційної системи Windows), а також *AkelPad, ConTEXT, Crimson Editor, E Text Editor, EditPlus, EmEditor, Metapad, MS-DOS Editor.*

Текстові редактори мають такі типові *функціональні можливості:*

- знайти і замінити (англ. find and replace) — знайти слово або словосполучення і замінити його на нове;
- вирізати, скопіювати і вставити (англ. cut, copy, and paste) — вилучити або скопіювати частину тексту чи весь текст і вставити потім його в будь-яке місце того самого або іншого файлу;
- форматування тексту (англ. text formatting) — застосувати найпростіші види форматування: перехід на новий рядок, автоматичний відступ, формування простого списку, підсвічування синтаксичних помилок тощо;
- відміна операції (відкочення назад) і відміна відкочення (англ. undo and redo) — відмінити останню операцію і повернутися до попередньої і перейти від передостанньої операції назад до останньої;
- можливість обробляти тексти у форматі Unicode.

Крім текстових редакторів загального призначення широко використовують також *текстові редактори спеціального призначення*, зокрема:

- редактори програмного коду з додатковими функціональними можливостями для полегшення створення програмного коду. Такі редактори можуть, наприклад, виявляти синтаксичні помилки;
- гіпертекстові редактори — призначені для створення вебсторінок із гіпертекстовими посиланнями мовою HTML;
- текстові редактори для створення статей, журналів, книжок у науковій сфері, зокрема у галузях математики, фізики, комп'ютерних наук (наприклад, *TeX* і *LaTeX*).

Текстові процесори (англ. word processors) мають значно більше засобів для оброблення текстової інформації, ніж текстові редактори. Наприклад, вони підтримують режим WYSIWYG (what you see is what you get — «що бачиш, те й отримаєш»).

Найпоширенішими текстовими процесорами є *Microsoft Word,* що входить до складу *Microsoft Office*, і *OpenOffice.org Writer* вільного програмного забезпечення *OpenOffice*. Застосовують також текстові процесори з відкритим кодом *LibreOffice Writer, AbiWord, KWord, LyX, Office Web Apps* та *Google Docs* у вебмережі.

Типовими функціями текстових процесорів є:

- пакетне укладання листів за допомогою шаблонів і адресної бази даних;
- індексація ключових слів та їхніх сторінок;
- автоматичне укладання змісту документа і його секцій із відповідними сторінками;
- перехресні посилання між секціями із вказівкою сторінок;
- оформлення посилань за номерами;
- варіантність документа за допомогою змінних (наприклад, номер моделі, артикул тощо);
- підтримка версій документа;
- групова робота з документом;

- коментарі й анотації до документів;
- підтримка малюнків, ілюстрацій і діаграм;
- підтримка внутрішнього взаємопосилання.

Мовна підтримка з боку текстового процесора зазвичай передбачає:

- перевірку орфографії;
- повідомлення про граматичні помилки, де такий висновок може зробити програма;
- тезаурус — словник, тобто пропозицію варіантів правильного написання для слів, які програма вважає набраними помилково.

Більшість текстових процесорів пропонують можливість зібрати статистику про редагований документ, а саме:

- кількість символів, слів, речень, рядків, абзаців, параграфів, сторінок;
- довжина слів, речень і абзаців;
- час редагування.

Настільні видавничі системи (англ. desktop publishing) — це комплекс програм, призначених для комп'ютерної верстки документів, тобто створення макета документа, який можна видати в друкарні або надрукувати на принтері. Організації, підприємства, окремі користувачі мають змогу отримувати для своїх потреб широкий спектр друкованої продукції. На практиці широко застосовують настільні видавничі системи *Adobe PageMaker, Adobe InDesign, QuarkXPress*.

11.3. ТЕКСТОВИЙ ПРОЦЕСОР Microsoft Word

Текстовий процесор *Microsoft Word* входить до пакета офісних програм *Microsoft Office*. Основним призначенням Word є створення текстових документів високої поліграфічної якості з ілюстраціями, таблицями, діаграмами та посиланнями. Microsoft Word можна також застосовувати як настільну видавничу систему для створення книжок і брошур невеликого обсягу.

Елементи вікна Microsoft Word

Для запуску програми Word необхідно натиснути:

- на кнопку *Пуск* у лівому нижньому куті екрана та вибрати *Програми — Microsoft Office — Microsoft Office Word;*
- на значок *Word* на Панелі завдань;
- правою кнопкою миші вибрати команду *Створити — Документ Microsoft Office Word.*

Після завантаження Microsoft Word на екрані з'являється вікно програми (рис. 11.1).

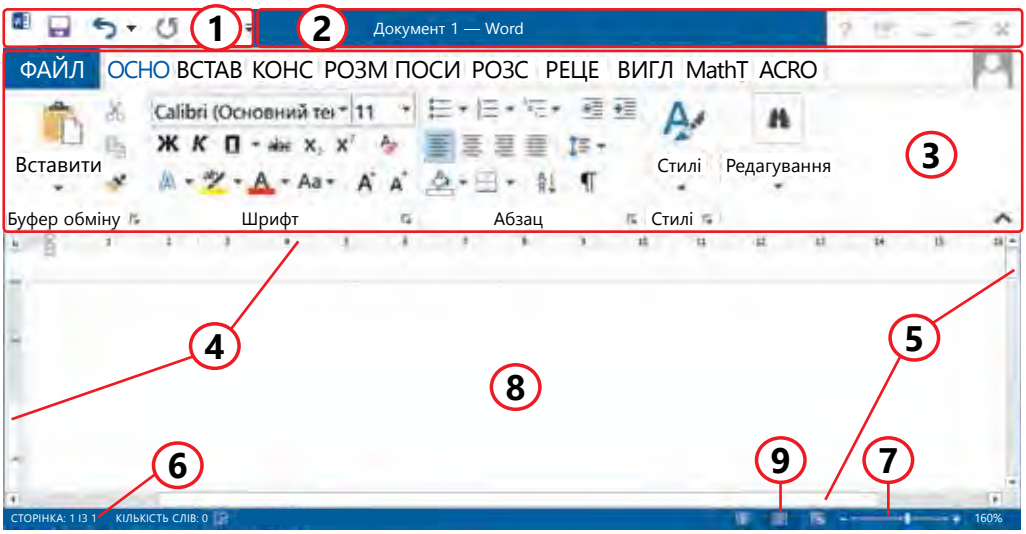


Рис. 11.1

1. Панель швидкого доступу стандартно розташована у верхній частині вікна і призначена для швидкого доступу до функцій, які часто використовують: *Зберегти, Скасувати, Повторити* та *Настроїти панель швидкого доступу* (рис. 11.2).

Рис. 11.2

2. Рядок заголовка містить назви програми й документа (стандартне ім'я документа Документ 1 — Word), кнопки для керування розмірами вікна (*Згорнути, Розгорнути* і *Закрити*).

3. Стрічка — це набір інструментів у верхній частині вікна програми, яка допомагає швидко знаходити потрібні команди. Команди впорядковані у групи.

У вікні відображається десять постійних *вкладок*:

- **Файл** містить команди для роботи з документами (*Створити, Зберегти, Зберегти як, Відкрити, Відомості, Закрити, Експорт, Друк* і *Параметри* для налаштування інтерфейсу програми);
- **Основне** — основні інструменти для форматування та редагування тексту;
- **Вставлення** містить інструменти, за допомогою яких можна додати до документа різноманітні об'єкти;
- **Конструктор** здійснює налаштування основних елементів дизайну усього документа: теми, палітри кольорів, шрифти тощо;

- **Макет** — різноманітні дії зі сторінкою: *Абзац, Параметри сторінки, Упорядкування* та ін.;
- **Посилання** здійснює налаштування змісту документа, гіперпосилань, міток тощо;
- **Розсилки** дає змогу налаштувати та здійснити масове розсилання документа вибраним користувачам;
- **Рецензування** містить інструменти, за допомогою яких можна додати до документа примітки, налаштувати опції перекладу, перевірки орфографії та граматики згідно зі встановленими словниками тощо;
- **Подання** налаштовує вигляд вікна (або кількох) програм, їхніх елементів, розташування тощо;
- **Розробник** допомагає створювати програми, записувати і виконувати макроси, використовувати команди, елементи керування і форми; цю вкладку типово приховано.

Отже, на кожній вкладці є декілька *груп* із командами, що мають спільне призначення (рис. 11.3).

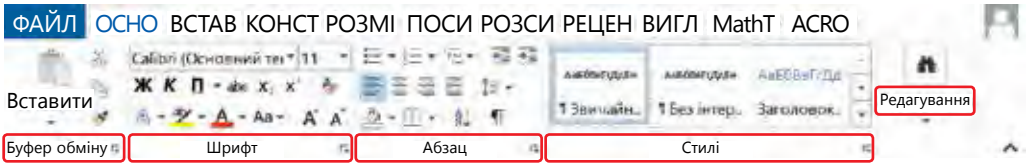


Рис. 11.3

1. *Діагональна стрілка* — кнопка виклику діалогового вікна ↘, яка розташована в нижньому правому куті більшості груп, призначена для відображення додаткових можливостей, пов'язаних із цією групою.

2. *Команди* — це елементи керування, кнопки, списки, що розкриваються, прапорці, лічильники та інші.

3. Замінити, видалити стрічку не можна, але для того, щоб збільшити робочу ділянку, стрічку можна сховати (згорнути): треба натиснути на кнопку й вибрати команду *Автоматично приховувати стрічку* (рис. 11.4).

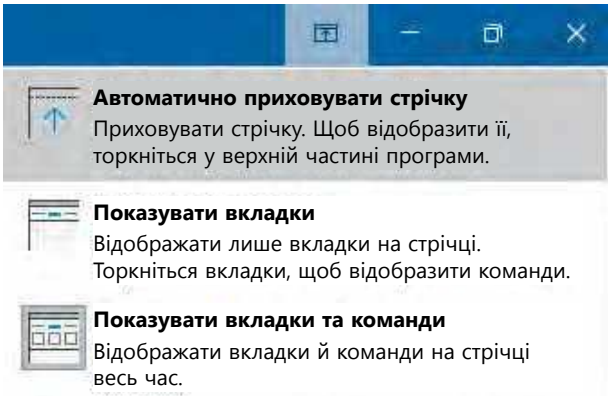


Рис. 11.4

4. Горизонтальна й вертикальна лінійки призначені для вирівнювання тексту, малюнків, таблиць та інших елементів документа. У верхній частині вікна розташована *горизонтальна* лінійка з маркерами (рис. 11.5), а в лівій — *вертикальна* лінійка. За допомогою маркерів і позначок на цих лінійках можна змінювати значення деяких властивостей об'єктів текстового документа (розміри полів, відступи абзаців тощо). Для відображення або приховування лінійки потрібно вибрати у вкладці **Подання** ☑ Лінійка.

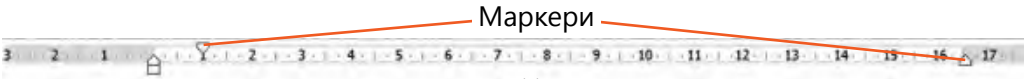


Рис. 11.5

5. Вертикальну й горизонтальну смуги прокручування використовують для тих частин документа, які розташовані поза екраном. Смуг немає, якщо текст документа міститься на екрані повністю.

6. У рядку стану, що в нижній частині вікна програми, виводяться такі повідомлення: номер поточної сторінки, кількість слів у тексті, тип режиму перевірки правопису, мова тексту тощо (рис. 11.6). Користувач може установити потрібний перелік повідомлень (додати або приховати елементи) за допомогою контекстного меню цього рядка:

Сторінка: 15 з 28	Слів: 5 466	українська

Рис. 11.6

- клацнути правою кнопкою миші в будь-якому місці рядка стану;
- у контекстному меню *Настроювання рядка стану* поставити або прибрати позначку напроти вибраного елемента (рис. 11.7).

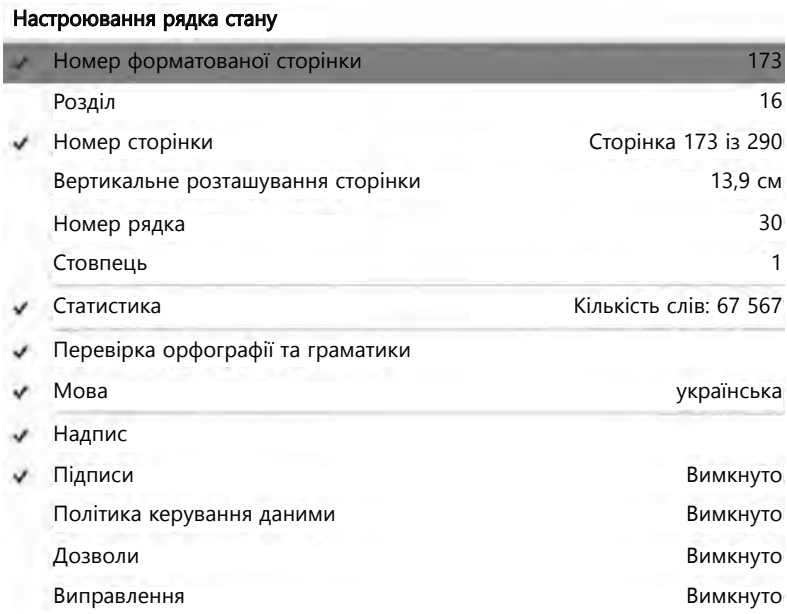


Рис. 11.7

7. Елемент **Масштаб** дає змогу змінити масштаб зображення одним із таких способів:

- перемістити повзунок праворуч для збільшення масштабу або ліворуч для зменшення;
- на стрічці скористатися командами вкладки **Подання** — група *Масштаб*;
- здійснити один клік лівою кнопкою миші на число відсотка біля повзунка

. Відкриється діалогове вікно *Масштаб*, у якому потрібно вибрати один із режимів перегляду або зазначити масштаб у відсотках (рис. 11.8). Натиснути на клавішу *CTRL*, прокрутити коліщатко миші вперед для збільшення масштабу, назад — для зменшення.

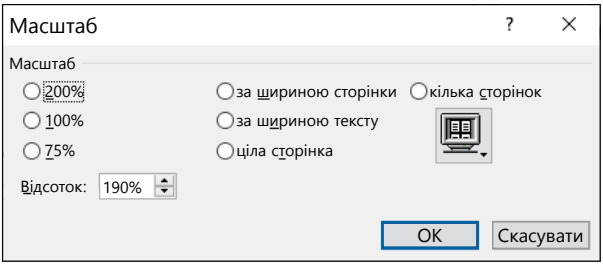


Рис. 11.8

8. Робоча ділянка займає найбільшу центральну частину вікна і призначена для відображення робочого документа.

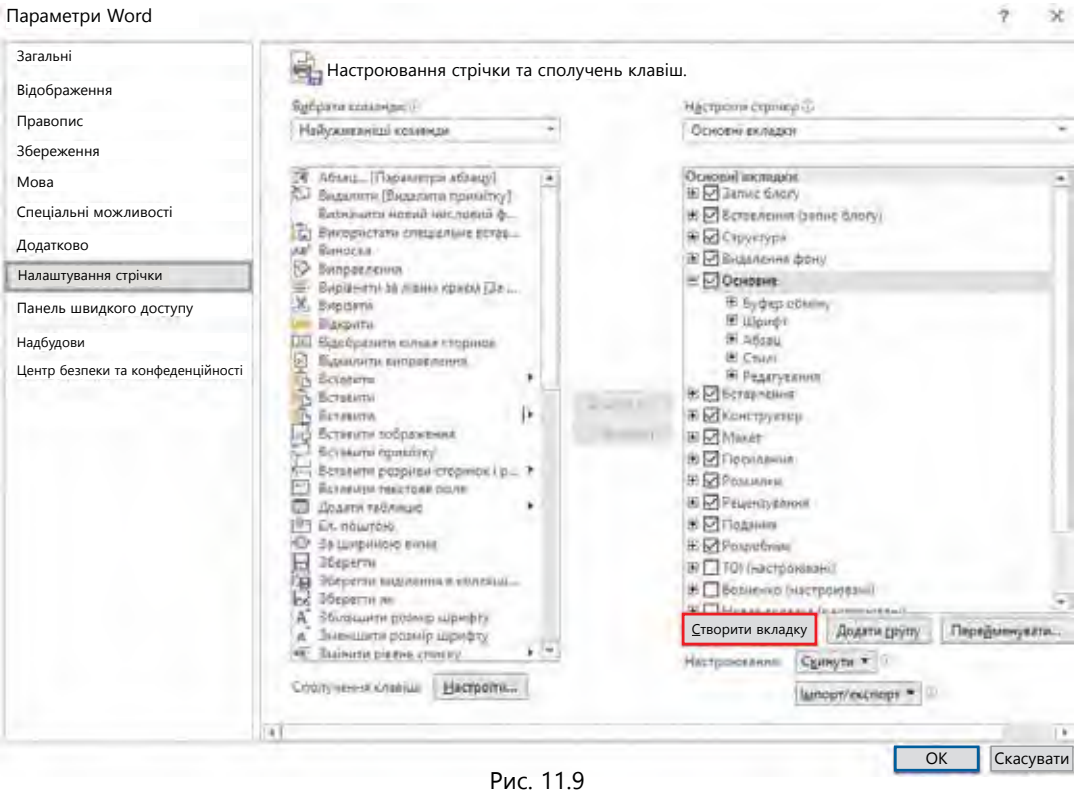


Рис. 11.9

9. Кнопки *Режими перегляду* розташовані праворуч від елементів *Масштаб* . Читати текст зручно в *Режимі читання:* приховуються всі інструменти, а документ займає все вікно. У режимі *Розмітка сторінки* відображається як сам документ, так і інструменти для роботи з ним. Режим *Вебдокумент* відкриває документ у режимі вебсторінки. Для перемикання між режимами також можна скористатися вкладкою **Подання**.

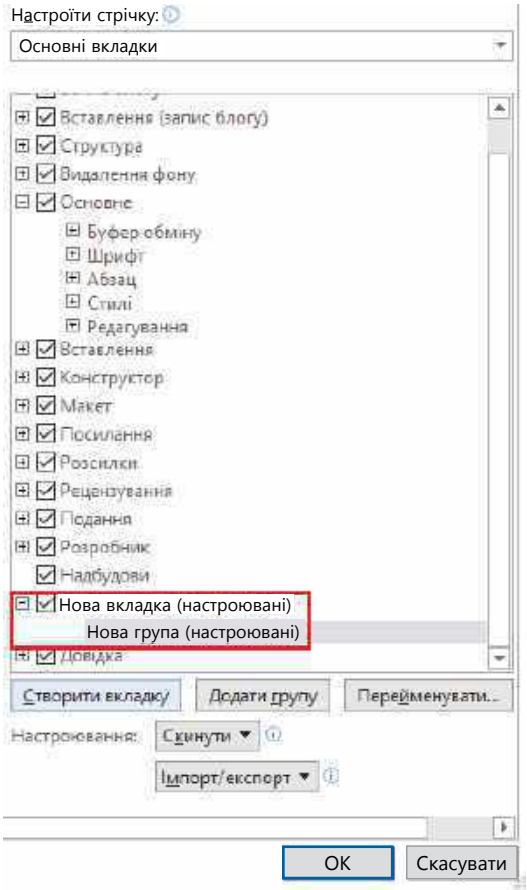


Рис. 11.10

Налаштування стрічки. Користуватися вкладками в більшості випадків досить зручно, однак іноді для роботи з різними об'єктами того самого документа необхідно перемикатися між стрічками, що потребує додаткових витрат часу. Щоб підвищити ефективність роботи, користувач може створити додаткові вкладки і розмістити на них будь-які групи командних кнопок, які використовує найчастіше. Для створення нової вкладки треба виконати команду *Настроювання стрічки* та сполучення клавіш *Настроювання стрічки* з контекстного меню. Відкриється діалогове вікно *Параметри Word,* у якому необхідно натиснути на кнопку *Створити вкладку* (рис. 11.9).

У правій частині вікна в ділянці *Основні вкладки* (рис. 11.10) відобразиться нова вкладка з ім'ям *Нова вкладка (настроювані),* яка містить одну групу *Нова група (настроювані).*

Ім'я вкладки та групи доцільно одразу змінити. Для цього необхідно їх виділити й натиснути кнопку *Перейменувати*, у діалоговому вікні зазначити ім'я вкладки, наприклад IT або групи Word. Далі командні кнопки з лівої частини вікна потрібно перетягнути в праву частину до нової створеної групи, вони там автоматично закріпляться. Можна також використати кнопку *Додати*. Після завершення створення нової вкладки та закриття діалогу налаштувань ця вкладка відобразиться на стрічці (рис. 11.11).

Для налаштування інтерфейсу програми використовують меню *Файл — Параметри.*

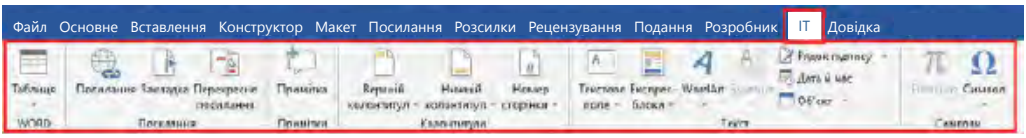


Рис. 11.11

Введення та редагування тексту

Як уже зазначено, текст складається зі структурних елементів: символів, слів, абзаців, сторінок. Розглянемо докладніше введення, редагування і форматування складників документа Word.

Введення символів. Символи вводять, як правило, з клавіатури на місце курсора — темної вертикальної риски.

Операції редагування, як і інші, можна скасовувати, використовуючи в панелі *Швидкого запуску* кнопку *Скасувати (Ctrl+Z)*. Список, що з'являється поруч із кнопкою скасування, дає змогу скасувати одразу кілька дій.

Щоб ввести у текст символ, якого немає на клавіатурі, використовують інструмент *Символ (Symbol)* у групі *Символи* на вкладці **Вставлення** (рис. 11.12).

За допомогою кнопки *Інші символи* відкривається діалогове вікно *Символ*, що має дві вкладки: *Символи і Спеціальні символи* (рис. 11.13). У полі *Шрифт* вибирають шрифт, що містить потрібні символи. Зазвичай це шрифт *Symbol*, хоча в деяких випадках можна скористатися іншими типами шрифту.

Рис. 11.12

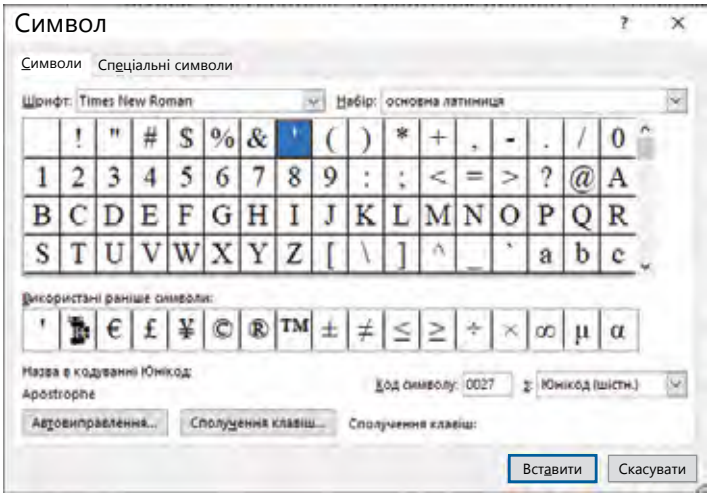


Рис. 11.13

Щоб вставити потрібний символ на місце курсора в тексті, треба двічі клацнути мишею на символі або натиснути на кнопку *Вставити*. Символ можна також ввести, набравши його порядковий номер на цифровій клавіатурі за натиснутої клавіші *Alt*. За допомогою кнопки *Сполучення клавіш* і відповідного діалогового вікна можна призначити символу комбінацію клавіш і відтак вводити цей символ із клавіатури.

Для вилучення символу ліворуч від курсора потрібно натиснути клавішу *Backspace,* а праворуч від курсора — клавішу *Delete*.

Переміщення в документі. Для швидкого переміщення текстом у процесі редагування можна використовувати такі клавіші й сполучення клавіш:

- клавіші керування курсором;
- смуги прокручування;
- *Home/End* — початок/кінець рядка;
- *Ctrl+Home* — до початку тексту; *Ctrl+End* — до кінця тексту;
- *Shift+F5* — повернення до попереднього місця редагування.

У процесі введення й редагування тексту можна вмикати режим відображення прихованих символів форматування, натиснувши кнопку *Відобразити всі знаки* ¶ в групі *Абзац* на вкладці **Основне**. У відповідних місцях тексту з'являться спеціальні позначки (табл. 11.1). До прихованих відносять символи, що використовуються для форматування тексту (абзац, пропуск, табуляція) і не відображаються під час друку.

Таблиця 11.1

Приклади недрукованих знаків

Недрукований знак	Клавіші введення	Позначення	Недрукований знак	Клавіші введення	Позначення
Пропуск	Пропуск	•	Нерозривний пропуск	*Ctrl+Shift+* пропуск	°
Кінець абзацу	*Enter*	¶	Нерозривний дефіс	*Ctrl+Shift+* дефіс	—
Табуляція	*Tab*	→	Розрив рядка	*Shift+Enter*	↵
М'який перенос	*Ctrl+*дефіс	¬	Розрив сторінки	*Ctrl +Enter*	·····Розрив сторінки·····

Перевірка правопису. Текстовий процесор Microsoft Word під час уведення тексту автоматично перевіряє орфографію та граматику. Це стандартні режими в програмі. Пошук орфографічних помилок у тексті проводиться за словником, який встановлено в Microsoft Office. Слова з помилками підкреслюються *червоною хвилястою лінією.* У разі граматичної помилки фрагмент тексту підкреслюється *зеленою хвилястою лінією*. Для виправлення ситуації треба відкрити контекстне меню цього фрагмента, встановити причину помилки й усунути її.

Для перевірки правопису тексту потрібно в групі *Правопис* на вкладці **Рецензування** вибрати кнопку *Правопис і граматика* або натиснути на функціональну клавішу *F7* (рис. 11.14). У вікні, що відкриється, програма виводить повідомлення про знайдені помилки та можливі варіанти їх усунення. Користувач може внести запропоновані виправлення до тексту або ж не зважати на вказівки і поради програми.

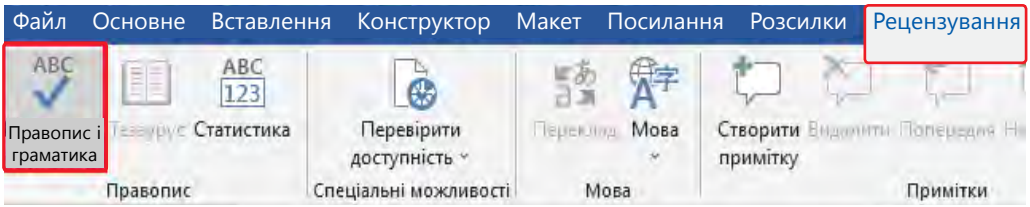


Рис. 11.14

Щоб вибрати мову перевірки в тексті, потрібно натиснути на кнопку *Мова* в групі *Мова* на вкладці **Рецензування** або в *Рядку стану,* де з'явиться діалогове вікно *Мова* (рис. 11.15).

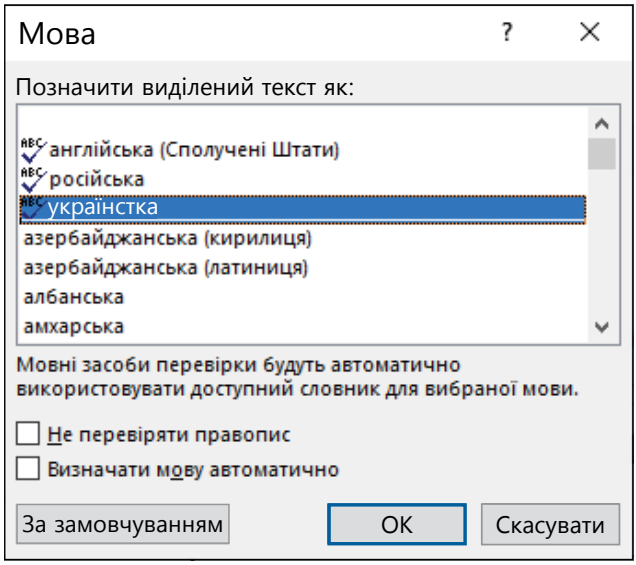


Рис. 11.15

Заміна тексту. Щоб замінити фрагменти тексту в документі, треба вибрати команду *Замінити* на вкладці **Основне** або виконати комбінацію клавіш *CTRL+H*. Відкриється діалогове вікно *Пошук і замінювання* з активною вкладкою *Замінити* (рис. 11.16).

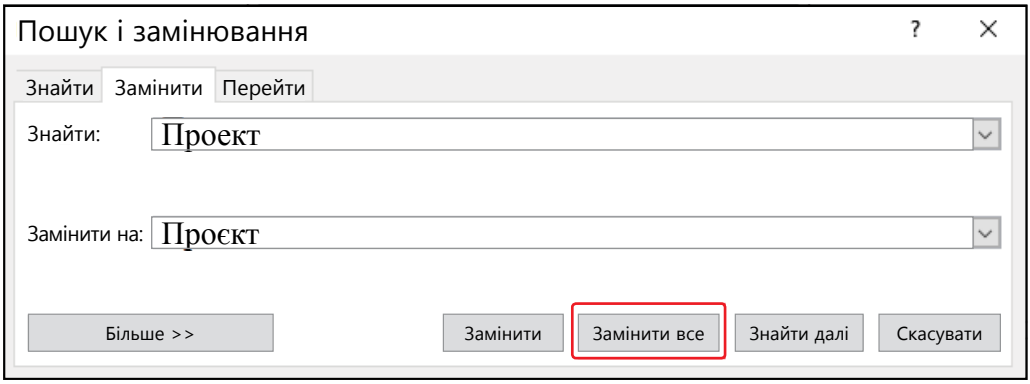


Рис. 11.16

У рядку *Знайти* набрати текст із документа, який необхідно замінити, а в рядку *Замінити* — текст, на який необхідно замінити (новий текст). У разі повної заміни — натиснути на кнопку *Замінити все*, якщо необхідно виконати вибіркову заміну — на кнопку *Замінити* або *Знайти далі*.

Форматування тексту

Виділення елементів тексту. Форматування, копіювання або переміщення здійснюється для виділених фрагмента тексту, символів, слів тощо. Виділяють різні складники документа за допомогою миші і/або клавіатури (табл. 11.2).

Таблиця 11.2

Способи виділення фрагментів тексту в документі Word

Елемент тексту	Спосіб виділення
Слово	Двічі клацнути на слові лівою кнопкою миші
Рисунок	Клацнути рисунок
Рядок тексту	Перемістити курсор до лівого краю рядка й клацнути лівою кнопкою миші
Кілька рядків тексту	Перемістити курсор до лівого краю першого чи останнього з виділених рядків і перетягнути курсор вгору або вниз
Речення	Тримаючи натиснутою клавішу *Ctrl,* клацнути в будь-якій частині речення
Абзац	Перемістити курсор до лівого краю абзацу й двічі клацнути кнопкою миші. Інший спосіб: тричі клацнути в будь-якій частині абзацу
Великий блок тексту	Клацнути початок фрагмента, прокрутити документ так, щоб на екрані з'явився кінець фрагмента, а потім клацнути його, тримаючи натиснутою клавішу *Shift*

Виділення тексту за допомогою клавіші *F8* здійснюють так:

- натиснути один раз — для ввімкнення режиму виділення (у рядку статусу активується індикатор ВДЛ);
- натиснути двічі — для виділення слова;
- натиснути тричі — для виділення речення;
- натиснути чотири рази — для виділення абзацу;
- натиснути п'ять разів — для виділення всього документа.

Після виділення фрагмента тексту поруч із ним автоматично з'являється напівпрозора мініпанель, на якій розміщено деякі елементи керування, що використовують для форматування фрагментів тексту (рис. 11.17).

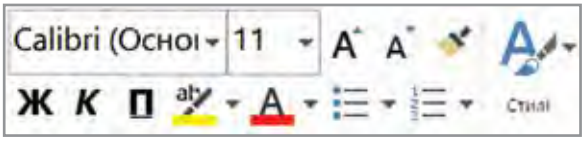


Рис. 11.17

Якщо потрібних інструментів форматування на мініпанелі немає, слід скористатися елементами керування груп *Шрифт* і *Абзац* вкладки **Основне** (рис. 11.18).

Елементи для форматування символів розміщено в групі *Шрифт* на вкладці **Основне** (рис. 11.19). Можна встановити такі параметри символів:

- шрифт (гарнітура), наприклад Times New Roman, Arial, Courier New;

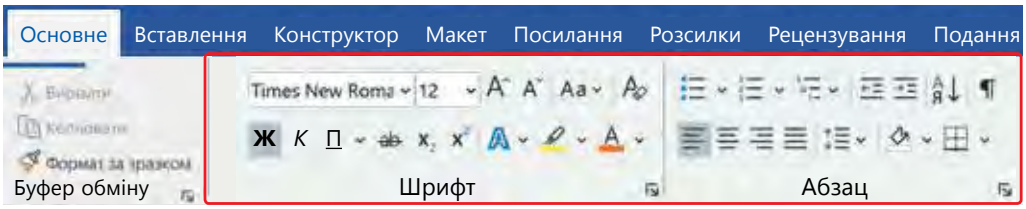


Рис. 11.18

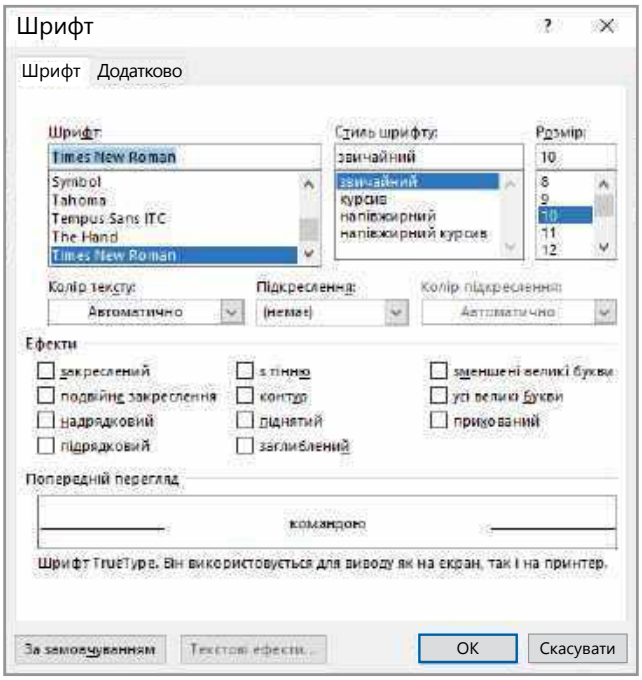


Рис. 11.19

- стиль шрифту — Звичайний, **Напівжирний**, *Курсив*, ***Напівжирний курсив;***
- розмір шрифту, що за традицією вимірюється в пунктах (пт);
- різні види підкреслень;
- колір символів;
- спеціальні ефекти (закреслення, подвійне закреслення, верхні та нижні індекси, тінь, контур).

На вкладці *Додатково* — *Інтервал* можна задати проміжок між символами у слові: звичайний, ущільнений або розріджений.

За допомогою поля *Зміщення* можна вибрати зміщення символів вгору і вниз щодо базової лінії.

Щоб надати словам, у яких трапляються деякі сполучення букв, кращого зовнішнього вигляду, застосовують *кернінг*, тобто штучне збільшення або зменшення інтервалу між деякими буквами у сполученнях (рис. 11.20).

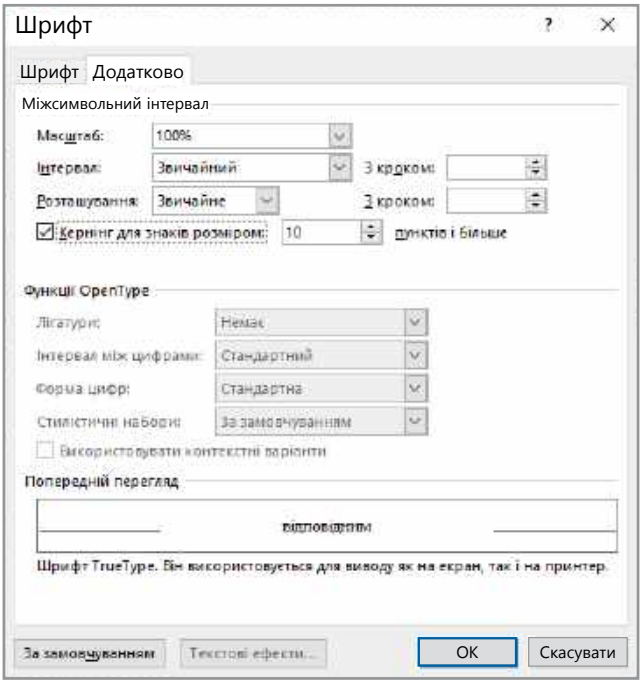


Рис. 11.20

Для керування регістром введених символів використовують команду *Регістр* із меню **Шрифт** або комбінацію клавіш *Shift+F3*. У діалоговому вікні *Регістр*, яке з'являється під час виконання відповідної команди, можна встановити такі режими:

- Як у реченнях — перший символ у першому слові речення переводиться у верхній регістр;
- всі малі — всі символи у вибраному фрагменті переводяться в нижній регістр;
- Всі великі — всі символи у вибраному фрагменті переводяться у верхній регістр;
- починати З Великих — у верхній регістр переводиться перший символ кожного слова виділеного фрагмента;
- змінити регістр — символи у верхньому регістрі переводяться у нижній і навпаки.

Форматування абзаців. *Абзац* — це частина текстової інформації, обмежена символом ¶ *(Кінець абзацу)*. Щоб під час введення текстової інформації перейти до нового абзацу, потрібно натиснути клавішу *Enter*, а щоб перейти на новий рядок у межах абзацу — комбінацію клавіш *Shift+Enter*.

Основні параметри форматування абзаців встановлюють у діалоговому вікні, що з'являється після вибору команди *Абзац* меню **Основне** (рис. 11.21). Аналогічну команду можна вибрати із контекстного меню, яке відкривається, якщо клацнути в довільному місці правою кнопкою миші.

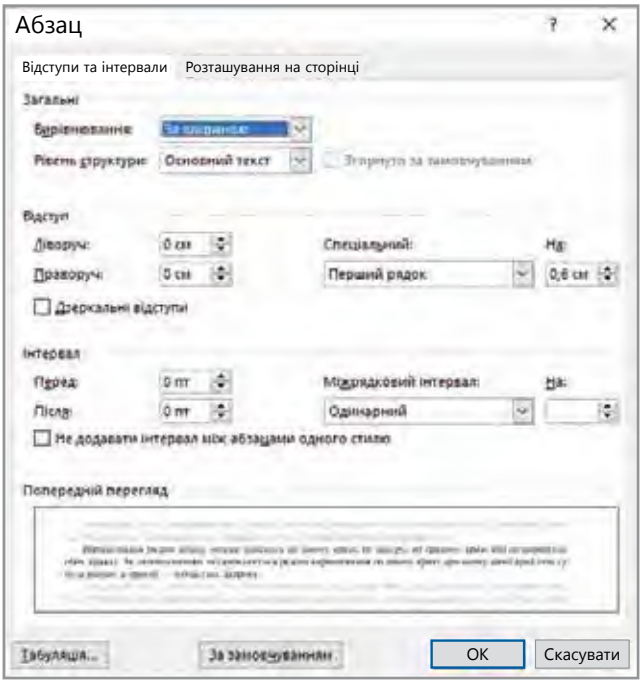


Рис. 11.21

Діалогове вікно *Абзац* має дві вкладки: *Відступи та інтервали; Розташування на сторінці.*

Вирівнювання рядків абзацу можна здійснити за лівим або правим краєм, за центром або за шириною. Типово встановлюється режим вирівнювання за лівим краєм, при цьому лівий край тексту буде рівним, а правий — зубчастим. Вибрати спосіб вирівнювання можна за допомогою списку *Вирівнювання* вкладки *Відступи і інтервали* діалогового вікна Абзац, кнопками або клавішами *Ctrl+L, Ctrl+R, Ctrl+E, Ctrl+S.*

Відступи абзацу ліворуч і праворуч від робочого поля сторінки і відступ першого рядка задають за допомогою відповідних регуляторів вкладки *Відступи та інтервали* діалогового вікна *Абзац*.

Для визначення відступів швидко й наочно використовують горизонтальну лінійку, пересуваючи відповідні маркери (рис. 11.22).

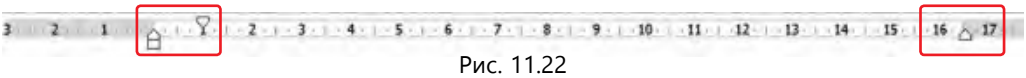


Рис. 11.22

Інтервал між рядками тексту абзацу вибирають зі списку *Міжрядковий інтервал:*

- *одинарний* — відстань, що дорівнює висоті шрифту найбільшого розміру, що використовується в рядку;
- *1,5 рядка* — інтервал, який у 1,5 разу перевищує відстань за одинарного інтервалу;

- *подвійний* — удвічі більше за одинарний інтервал;
- *мінімум* — мінімальний інтервал, який вибирають для дуже великих шрифтів;
- *точний* — інтервал, який точно дорівнює значенню, вказаному в полі *Значення*;
- *множинний* — формується перемножуванням одинарного інтервалу на значення, вказане в полі *Значення*.

Встановити запропоновані міжрядкові інтервали або вибрати інший можна за допомогою кнопки *Міжрядковий інтервал та інтервал між абзацами* .

Щоб поліпшити зовнішній вигляд тексту, задають інтервали перед і після абзацу за допомогою відповідних регуляторів вкладки *Відступи і інтервали* діалогового вікна *Абзац*.

Для надання тексту більшої виразності окремі абзаци і заголовки іноді вкладають у рамки з тінню і тлом: команда *Межі та тіні* на вкладці **Основне**, група *Абзац,* кнопка *Межі,* яка викликає на екран однойменне вікно *Межі й заливка* з вкладками: *Межі, Сторінка* та *Заливка* (рис. 11.23). На вкладці *Межі* встановлюють вид рамки, тип, колір і ширину ліній. Результат цієї процедури відображається у вікні *Зразок*. У разі використання вкладки *Сторінка* характер дій залишається той самий, але вони поширюються на всю сторінку. На вкладці *Заливка* можна вибрати колір заливання і тла, тип візерунка.

Під час формування сторінок тексту деякі абзаци розриваються на частини, що інколи буває небажано. Щоб уникнути цього, треба встановити перемикач *Не розривати абзац* вкладки *Положення на сторінці* вікна *Абзац*.

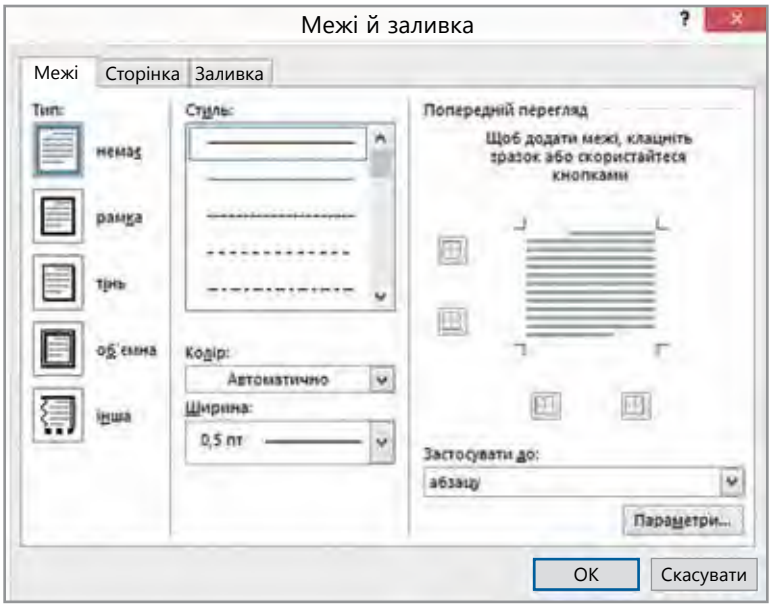


Рис. 11.23

Форматування сторінок. Поділ документа на сторінки здійснюється автоматично, якщо увімкнено режим *Фоновий поділ на сторінки*, розміщений на вкладці *Загальні* діалогового вікна *Параметри*, вкладка *Файл*. У багатьох випадках користувачі здійснюють такий поділ вручну: встановлюють курсор на місце поділу і вибирають команду *Розриви* на вкладці **Макет**, внаслідок чого на екрані з'являється однойменне діалогове вікно (рис. 11.24). Поділити документ на сторінки можна також за допомогою клавіатури комбінацією клавіш *Ctrl+Enter*.

Робота зі сторінками документа здійснюється в режимі *Розмітка сторінки*, який задається за допомогою однойменної команди на вкладці **Подання**.

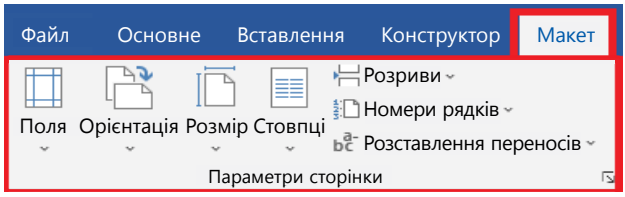


Рис. 11.24

Розглянемо найпоширеніші елементи форматування сторінок документа, які задаються за допомогою діалогового вікна *Параметри сторінки* на вкладці **Макет** (рис. 11.25).

Розмір сторінки вибирають на вкладці *Папір* зі списку *Розмір паперу* або за допомогою регуляторів *Ширина і Висота*. На вкладці *Поля* за допомогою перемикачів вибирають орієнтацію сторінки: книжкову або альбомну.

Розміри полів визначають за допомогою відповідних регуляторів на вкладці *Поля*. Щоб встановити дзеркальні поля на сторінці, треба поставити відповідну мітку.

Змінювати поля на сторінці можна легко й наочно, переміщуючи індикатори полів на лінійці форматування.

Межі тексту можна зробити видимими, якщо по-

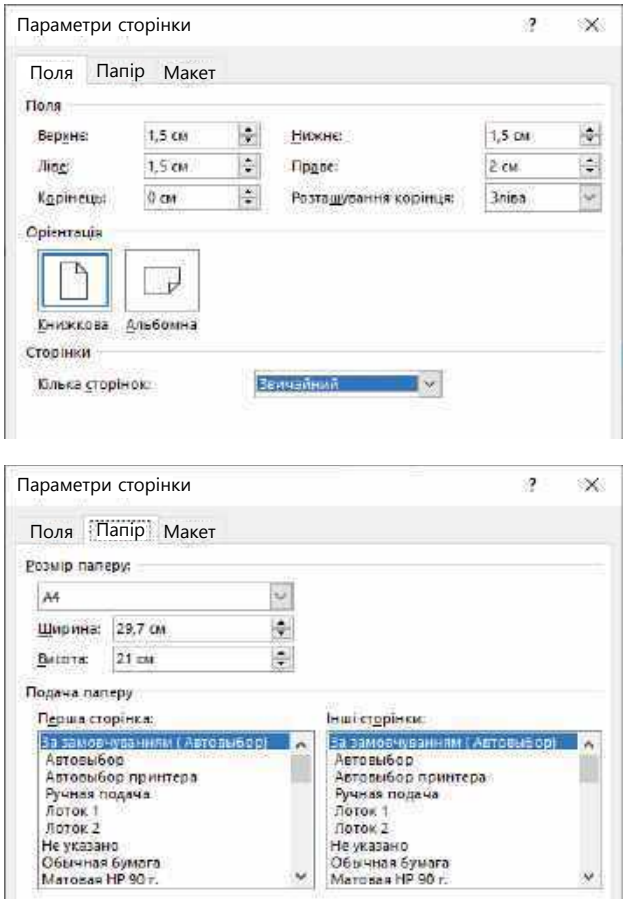


Рис. 11.25

ставити мітку *Відображати межі тексту* в меню **Файл**: команда *Параметри* в діалоговому вікні *Параметри*.

У колонтитулах (верхньому й нижньому) зазвичай подають інформацію про текст (автор, загальна назва, назви розділів, параграфів тощо, номери сторінок). Оформлюють ці елементи за допомогою діалогових вікон *Верхній колонтитул* і *Нижній колонтитул* на вкладці **Вставлення**.

Нумерацію сторінок можна додати разом із колонтитулами або окремо за допомогою діалогового вікна *Номера сторінок*, викликаного однойменною командою з групи *Колонтитули* на вкладці **Вставлення**.

Таблиці

Таблиця — зручна й наочна форма подання інформації, тому її дуже широко застосовують у текстах, особливо економічного і соціального спрямування. Інформацію в таблицях розміщують у *клітинках*, які утворюють *стовпці* й *рядки*. Стовпці позначають зліва направо буквами латиниці (A, B, C, ... , X, Y, Z), а рядки — зверху вниз цифрами (1, 2, 3, ...). У клітинках таблиці можуть розміщуватися дані будь-якого виду: числа, текст, графіка.

Створити таблицю і вставити її в текст можна різними способами.

Перед вставкою в документ таблиці потрібно встановити курсор у відповідне місце документа. Після натискання кнопки *Таблиця* на вкладці **Вставлення** відображаються опції всіх способів вставлення і створення таблиць.

У вікні *Вставлення таблиці* подано трафарет таблиці розміром 4×5 клітин. Притиснувши ліву кнопку миші і пересуваючи вказівник миші праворуч і вниз, вибираємо потрібну кількість стовпців і рядків (рис. 11.26).

За допомогою кнопки *Вставити таблицю* також можна в діалоговому вікні вибрати кількість стовпців і рядків, ширину стовпців (рис. 11.27).

Після створення або виділення таблиці на стрічці вікна програми з'являється група інструментів під загальною назвою **Робота з таблицями**,

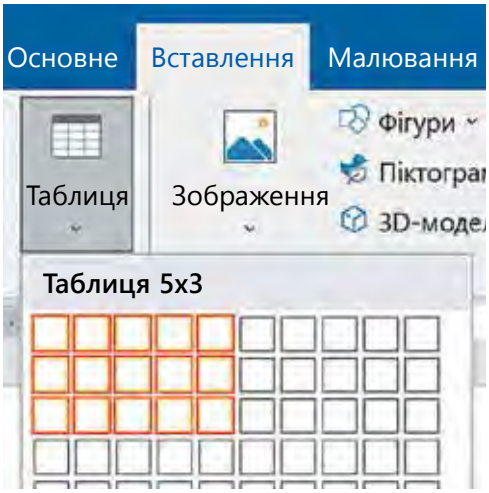


Рис. 11.26

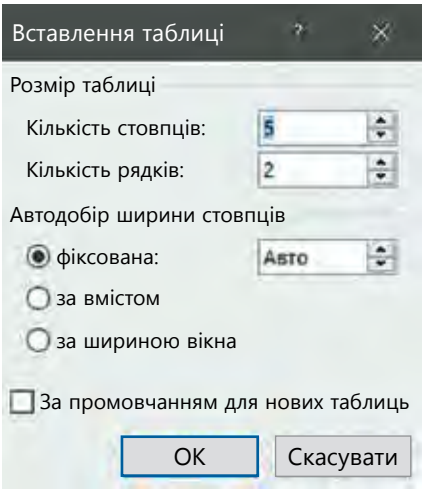
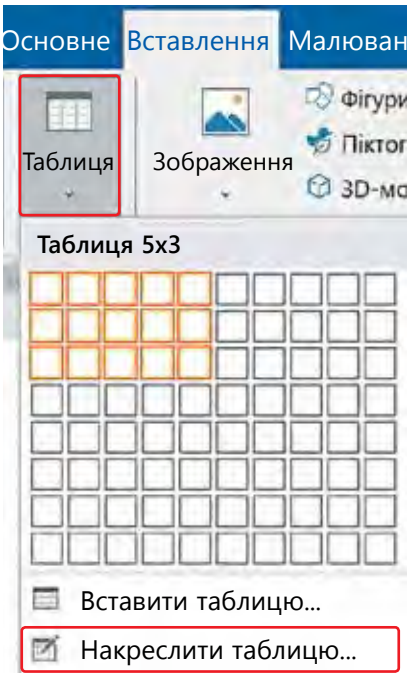


Рис. 11.27



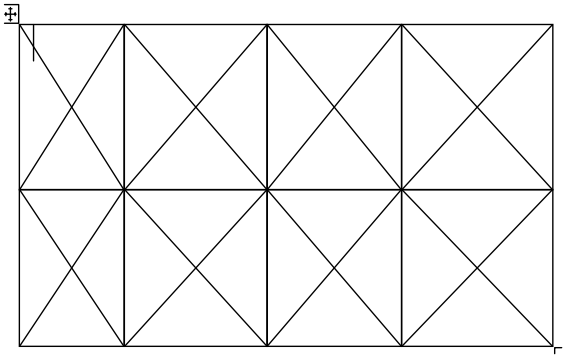


Рис. 11.28

яка відображається в рядку заголовка і містить дві вкладки *Конструктор таблиць* і *Макет*.

Креслення таблиці виконують у довільній формі. У меню *Таблиця* потрібно вибрати команду *Накреслити таблицю* (рис. 11.28) і переміщувати вказівник миші, який набирає вигляд олівця, вниз і праворуч до отримання потрібного розміру.

Таблицю можна створити також на основі вже наявного тексту, виконавши такі операції:

- поділити текст на стовпці за допомогою символу абзацу (*Enter*), табуляції, крапки з комою або будь-якого іншого символу;
- виділити перетворений текст і виконати команду *Перетворити на таблицю* в групі *Таблиці* на вкладці **Вставлення**. У діалоговому вікні, що з'явиться, задати потрібну кількість стовпців і рядків.

Дані в таблицю вводять у клітинки, починаючи з позиції курсора. У міру заповнення клітинки її розміри по вертикалі автоматично збільшуються.

Переміщення між клітинками таблиці здійснюють за допомогою клавіатури такими комбінаціями клавіш:

- *Tab, Shift+Tab* — перехід до наступної і попередньої клітинки;
- *Alt+Home, Alt+End* — перехід до першої і останньої клітинки рядка;
- *Alt+PgUp, Alt+PgDn* — перехід до першої і останньої клітинки стовпця.

Як і під час роботи зі звичайним текстом, перед тим, як застосувати *операцію* до певного об'єкта або групи об'єктів таблиці, їх потрібно виділити.

Щоб об'єднати клітинки, потрібно виділити групу клітинок і виконати команду *Об'єднати клітинки* з меню *Таблиця* (рис. 11.29).

Для поділу клітинки на кілька в рядку використовують команду *Розділити клітинки* з меню *Таблиця*. У діалоговому вікні, що відповідає цій команді, у полі *Кількість стовпців* потрібно зазначити відповідне число.

Щоб додати новий рядок до таблиці, потрібно виділити один із рядків, а потім виконати команду *Вставити рядок вище* з мініпанелі (рис. 11.30, *а*)

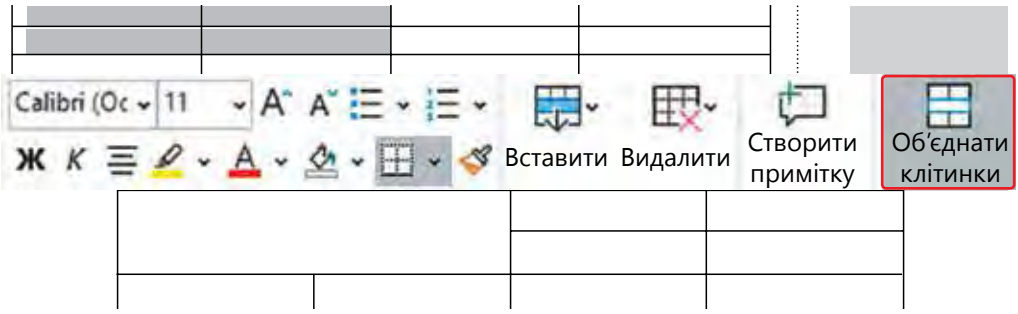
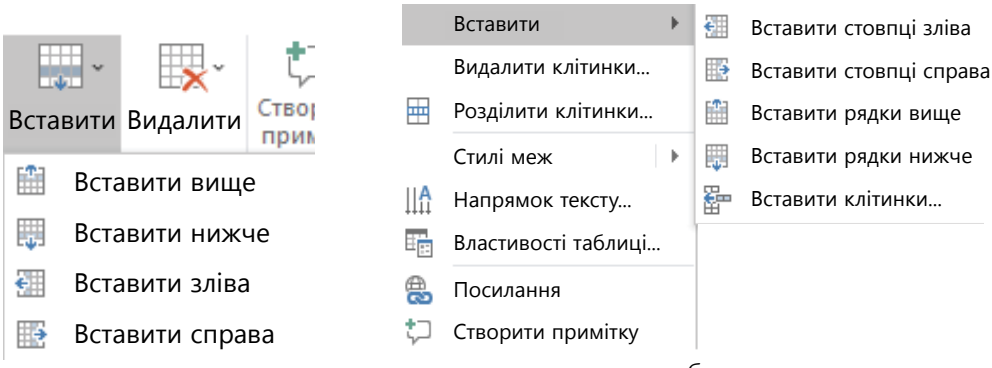


Рис. 11.29

або з контекстного меню *Вставити рядки вище* (рис. 11.30, *б*). Аналогічно виконують команду *Вставити стовпці зліва* чи *справа*.



а *б*

Рис. 11.30

Попередньо вибрані клітинки таблиці вилучають командою *Видалити* у групі *Рядки і стовпці* на вкладці *Макет* у **Робота з таблицями**. У діалоговому вікні цієї команди потрібно вибрати одну з чотирьох операцій (рис. 11.31):

- зі зсувом ліворуч — вибрані клітинки вилучаються, а на їхнє місце зсуваються клітинки, розміщені праворуч;
- вилучити весь рядок;
- зі зсувом угору — вибрані клітинки вилучаються, а на їхнє місце зсуваються клітинки розміщені внизу;
- вилучити весь стовпець.

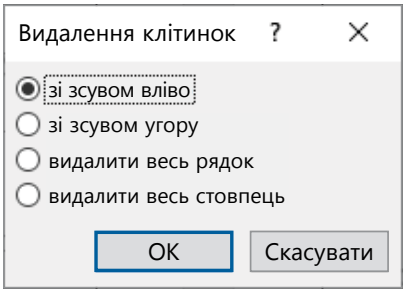


Рис. 11.31

Щоб виконати *сортування* в таблиці, треба помістити курсор у довільну клітинку і вибрати команду *Сортування* на вкладці *Макет* у **Робота з таблицями**.

Відкриється діалогове вікно *Сортування*, за допомогою якого можна виконати сортування в кількох колонках, але не більше ніж у трьох.

У полі *Сортувати* вказують колонку, у якій сортування має здійснюватися в першу чергу. У полях *Потім* зазначають, якщо це потрібно, решту колонок, у яких виконуватиметься сортування. Для кожної з колонок потрібно вказати тип інформації в полі *Тип*: текст, число, дата. Напрям сортування вибирають за допомогою параметра *За зростанням* і *За спаданням* (рис. 11.32).

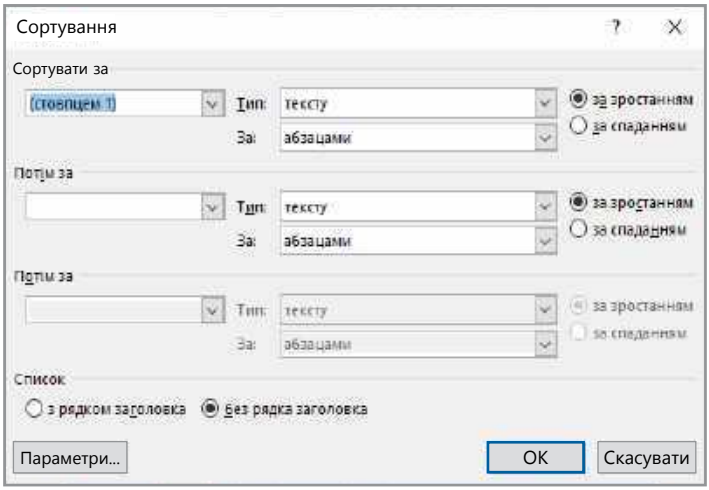


Рис. 11.32

Форматування тексту всередині таблиці здійснюється за допомогою тих самих засобів, що і для звичайного тексту.

Під час створення таблиці або під час роботи з уже наявними таблицями можна скористатися автоматичним форматуванням у групі *Стилі таблиць* на вкладці **Конструктор**, зокрема вибрати один із стилів рамок, а також скористатися готовими розробками з оформлення клітинок (рис. 11.33).

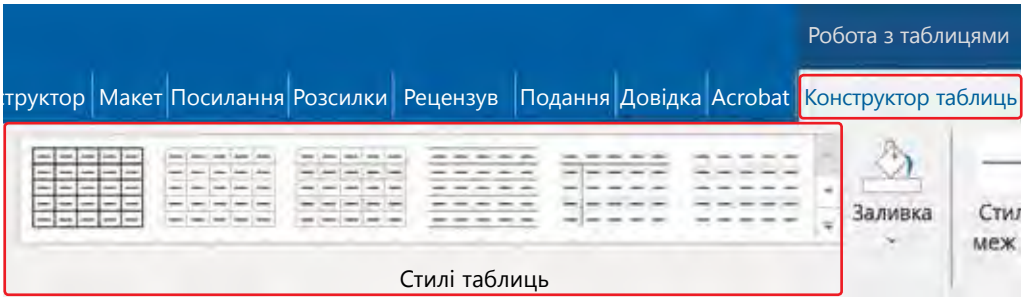


Рис. 11.33

Ширину колонки або висоту рядка зручно змінювати безпосередньо в таблиці за допомогою миші. Якщо підвести вказівник миші до лінії, що розділяє колонки чи рядки, то курсор змінює свій зовнішній вигляд на дві паралельні

лінії. Клацнувши в цей момент лівою кнопкою миші й тримаючи її в натиснутому стані, можна перемістити лінію розмежування в потрібну позицію, формуючи тим самим висоту рядка або ширину колонки.

За допомогою миші можна змінювати розміри клітинок, переміщуючи спеціальні елементи на вертикальній і горизонтальній лінійках форматування (рис. 11.34).

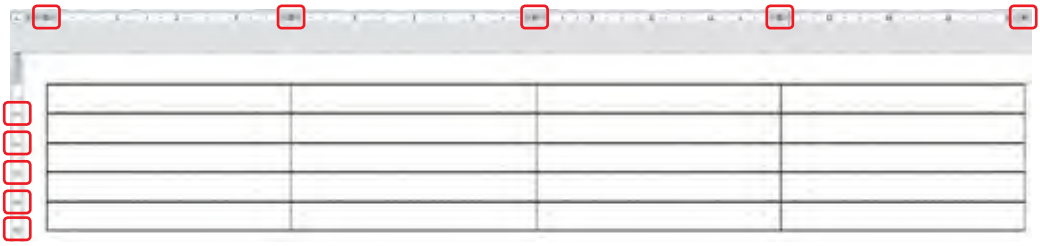


Рис. 11.34

Ще один спосіб — вкладки *Рядок* і *Стовпець* команди *Висота* і *Ширина* в діалоговому вікні *Властивості таблиці* (рис. 11.35). Параметри колонок таблиці встановлюють за допомогою вкладки *Стовпець*. Як і при форматуванні рядків, якщо потрібно відформатувати одразу кілька колонок, потрібно їх попередньо вибрати. Кнопки *Попередній стовпець* і *Наступний стовпець* слугують для переходу між стовпцями. Висоту рядка визначають у полі *Висота* рядка.

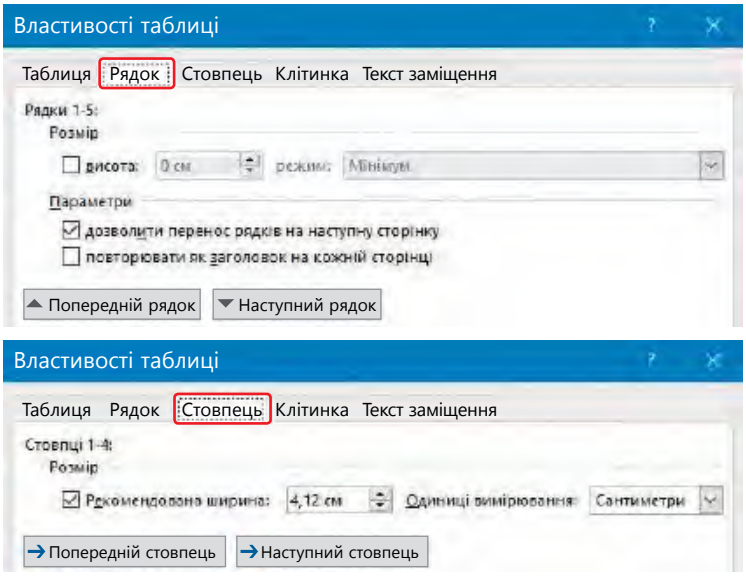


Рис. 11.35

Ширину колонки (або кількох колонок) вказують у полі *Рекомендована ширина* стовпця. Перехід між рядками здійснюють за допомогою кнопок *Попередній рядок* і *Наступний рядок*.

Параметр *Вирівнювання* визначає спосіб вирівнювання рядка щодо полів сторінки: за лівим краєм, по центру, за правим краєм. Потрібно розрізняти вирівнювання тексту всередині клітинок і вирівнювання рядків таблиці щодо полів сторінки.

Деякі великі таблиці доводиться розміщувати на кількох сторінках. Заголовки таких таблиць повторюють із переходом на нову сторінку: команда *Повторити рядки заголовків* на вкладці *Макет*.

Напрямок тексту в таблиці вибирають за допомогою команди *Напрямок тексту* на вкладці *Макет* (рис. 11.36).

Рис. 11.36

Формули у тексті

Багато текстів, зокрема науково-технічні, містять символи, яких немає не тільки на клавіатурі, а й на вкладках діалогового вікна *Символ*, яке викликається однойменною командою з меню **Вставлення**. Для введення таких складних символів, передусім математичних, передбачено конструктор формул. Щоб вставити у текст математичну формулу, рівняння або вираз, треба скористатися відповідною вкладкою в меню **Вставлення** — *Символи* (рис. 11.37). Щоб ввести формулу з нуля, натисніть клавіші *Alt* + = на клавіатурі.

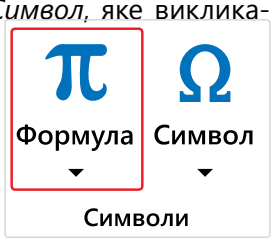


Рис. 11.37

Команда *Вставити нове рівняння* виводить на екран нову панель інструментів — *Засоби для роботи з рівнянням* (рис. 11.38).

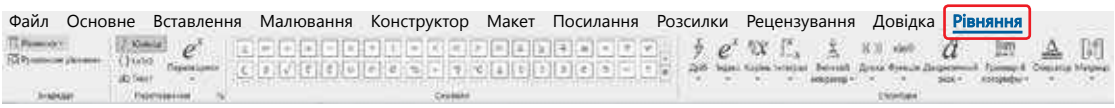


Рис. 11.38

Щоб додати формули до колекції формул, треба виділити формулу (рис. 11.39), натиснути клавішу зі стрілкою вниз і вибрати команду *Зберегти виділений фрагмент у колекції рівнянь*.

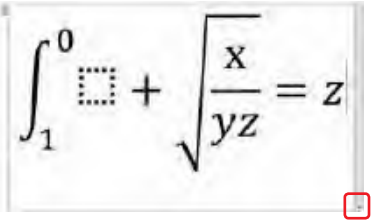


Рис. 11.39

У діалоговому вікні *Створення стандартного блока* ввести ім'я формули (типово відображатиметься виділена формула). У списку колекції вибрати *Рівняння* і в категорії *Алгебра* натиснути OK (рис. 11.40).

Для створення рукописного рівняння треба вибрати команду *Рукописне рівняння* й натиснути на інструмент *Записати* (рис. 11.41).

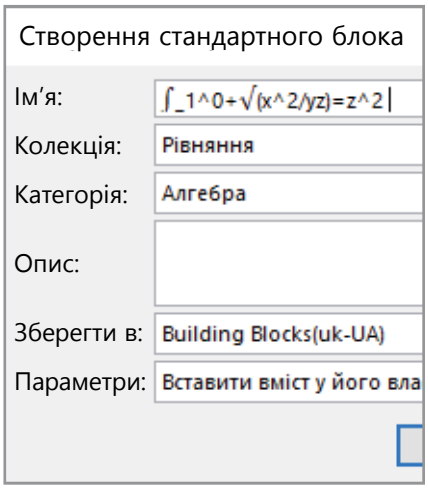


Рис. 11.40

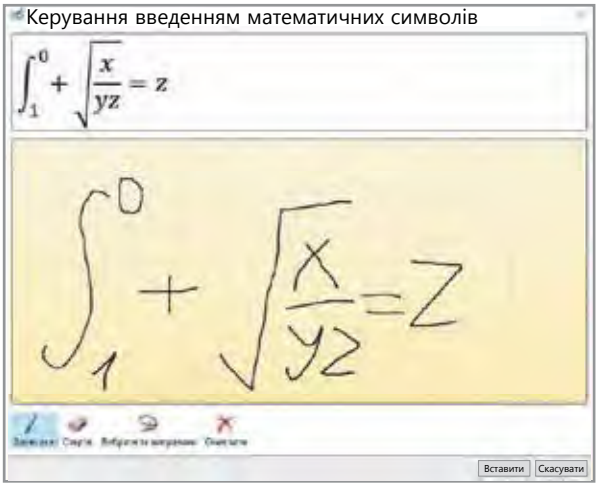


Рис. 11.41

Щоб вийти з *Редактора формул*, потрібно натиснути клавішу *Esc* або клацнути мишею поза полем формули.

ТЕСТОВІ ЗАВДАННЯ

ЗМІНА ЗОВНІШНЬОГО ВИГЛЯДУ ТЕКСТУ — ЦЕ:

1. редагування тексту
2. форматування тексту
3. створення проєктів
4. внесення змін до тексту

ЯК МОЖНА ВИДІЛИТИ ВЕСЬ ТЕКСТ?

1. Ctrl+C
2. Ctrl+X
3. Ctrl+Z
4. Ctrl+A

У ТЕКСТОВОМУ РЕДАКТОРІ ПРИ ВСТАНОВЛЕННІ ПАРАМЕТРІВ СТОРІНКИ ВИБИРАЮТЬ:

1. гарнітуру, розмір, накреслення
2. відступ, інтервал, вирівнювання
3. поля, орієнтацію, колонтитули
4. стиль, шаблон

У ПРОЦЕСІ ФОРМАТУВАННЯ ТЕКСТУ ЗМІНЮЮТЬ:

1. параметри абзацу
2. послідовність символів, слів, абзаців
3. параметри сторінок
4. нумерацію малюнків

У ТЕКСТОВОМУ РЕДАКТОРІ ГОЛОВНИМИ ПАРАМЕТРАМИ ПРИ ВСТАНОВЛЕННІ ПАРАМЕТРІВ АБЗАЦУ Є:

1. гарнітура, розмір, накреслення
2. відступ, інтервал, вирівнювання
3. поля, орієнтація
4. стиль, шаблон

Розділ 12
ОБРОБЛЕННЯ ГРАФІЧНОЇ ІНФОРМАЦІЇ. ПРОГРАМА Microsoft Visio

12.1. ОСНОВНІ ПОНЯТТЯ КОМП'ЮТЕРНОЇ ГРАФІКИ

Графічні зображення є важливою формою інформації. Комп'ютерна графіка — це галузь інформатики, яка вивчає методи і засоби створення та оброблення зображень за допомогою програмно-апаратних обчислювальних комплексів. За способом формування зображень комп'ютерну графіку поділяють на растрову, векторну, фрактальну і тривимірну (3D).

Растрове зображення являє собою набір пікселів (англ. picel picture element — «елемент зображення»), кожний із яких має певний колір. Розмір растрового зображення — це кількість пікселів за горизонталлю й вертикаллю, наприклад растрові зображення операційної системи Windows на екрані монітора мають такі розміри: 640×480, 1024×768, 1240×1024 і більше. Чим більше пікселів має графічне зображення, тим кращою є якість зображення.

Роздільна здатність — це величина, що показує, скільки пікселів розміщується на одиниці довжини зображення. Розрізняють такі види роздільної здатності:

- роздільна здатність зображення — вимірюють у точках на дюйм (англ. dpi — dots per inch), залежить від вимог до якості зображення та розміру файлу, способу оцифрування або методу створення початкової ілюстрації, формату файлу та інших параметрів. Чим вищі вимоги до якості, тим більшою має бути роздільна здатність зображення;
- роздільна здатність екранного зображення — вимірюють в елементарних точках растра, які називають *пікселями* (англ. pixel, — picture element). Розмір пікселя варіюється залежно від вибраної екранної роздільної здатності (з діапазону стандартних значень), роздільної здатності зображення і масштабу відображення;
- роздільна здатність друкованого зображення залежить від застосованого методу та параметрів растрування. Щільність растра, що вимірюється кількістю ліній на дюйм, — це *лініатура*.

Растрова графіка дає змогу точно передавати найменші деталі, тому растрові графічні зображення мають високу якість. Разом із тим, для збереження таких зображень потрібен великий обсяг пам'яті. Наприклад, для збереження зображення на екрані дисплея системи з розмірами 1024×768, за умови, що колір кодується трьома байтами, необхідно 1024×768×3=2,3 Мбайтів пам'яті. Крім того, досить складно змінювати масштаб растрового зображення і редагувати його.

Для **векторної графіки** базовим елементом зображення є *лінія*. Як і будь-який об'єкт, лінія має такі властивості: форму (пряма, крива), товщину, колір, шрифт (суцільна, пунктирна).

Векторне графічне зображення складається з простих графічних елементів (прямих і кривих ліній) та геометричних фігур (трикутників, прямокутників, кіл, еліпсів, дуг). Для побудови кожного такого елемента складають програму, в якій задають відповідні параметри: довжину, товщину, орієнтацію, колір лінії, радіус і координати центра кола тощо.

Об'єкти векторної графіки мають різні способи подання.

- Точку на площині задають двома числами (координатами x, y), що показують її положення щодо початку координат.
- Пряму лінію описують рівнянням $y = kx + b$. Вказавши параметри k і b, завжди можна відобразити нескінченну пряму лінію в системі координат, тобто для опису прямої лінії достатньо двох параметрів.
- Для відрізка прямої потрібно ввести координати початку і кінця.
- Криву другого порядку (параболу, гіперболу, еліпс, окружність) описують рівняннями, які містять степені не вище, ніж другий. Така крива не має точок перегину. Для опису нескінченної кривої другого порядку достатньо п'яти параметрів, а для частини кривої потрібно ще два — координати кінцевих точок. Формула кривої другого порядку в загальному випадку має вигляд: $x^2 + a1 \cdot y^2 + a2 \cdot x \cdot y + a3 \cdot x + a4 \cdot y + a5 = 0$.
- Крива третього порядку відрізняється від кривої другого порядку можливою наявністю точки перегину. Для її опису потрібно дев'ять параметрів, а для опису її частини — ще два. В загальному випадку рівняння кривої третього порядку має такий вигляд: $x^3 + a1 \cdot y^3 + a2x^2 \cdot y + a3 \cdot x \cdot y^2 + a4 \cdot x^2 + a5 \cdot y^2 + a6 \cdot x \cdot y + a7 \cdot x + a8 \cdot y + a9 = 0$.

Перевагою векторних зображень порівняно з растровими є те, що обсяг пам'яті для збереження векторних у десятки, сотні й навіть тисячі разів менший, ніж для таких самих растрових зображень.

Векторна графіка має особливі переваги для передавання штучних графічних об'єктів: технічних креслень, схем, графіків, діаграм тощо. Природні об'єкти мають, здебільшого, плавний перехід ліній і кольорів, тому є певні труднощі під час їх зображення засобами векторної графіки.

Фрактальна графіка, як і векторна, основана на математичних обчисленнях. Однак її базовим елементом є сама математична формула, тобто жодних об'єктів у пам'яті комп'ютера не зберігається, зображення будується суто за рівняннями.

Тривимірна графіка поєднує, як правило, векторний і растровий способи формування зображень. Цей різновид графіки вивчає прийоми й методи побудови об'ємних моделей об'єктів у віртуальному просторі. 3D-графіку широко застосовують для наукових розрахунків, інженерного проєктування, комп'ютерного моделювання фізичних об'єктів.

12.2. ГРАФІЧНИЙ РЕДАКТОР Microsoft Visio І ЙОГО ПРИЗНАЧЕННЯ

Microsoft Visio (MS Visio) — потужний графічний редактор ділової та інженерної векторної графіки, призначений для створення векторних графічних зображень будь-якої складності. За допомогою вбудованих шаблонів, трафаретів і стандартних модулів надано можливість створювати найпростіші схеми, плани, креслення, графічні та організаційні діаграми і роботи з даними без художніх або технічних навичок. Це універсальний креслярський засіб, який буде корисний кожному, кому потрібно створити власні методи організації інформації, діаграми, пов'язані з проєктуванням систем, інженерним проєктуванням, розробленням додатків, обслуговуванням, технічною підтримкою, бізнес-процесами тощо та передати інформацію про процеси, інфраструктуру.

MS Visio дає змогу зберігати створені файли в різних графічних форматах: GIF (Graphic Interchange Format), JPEG (Joint Photographic Expert Group), креслення Auto CAD тощо.

Усі Visio-файли мають той самий формат. Однак розширення файлу Visio-документа визначається тим, як саме файл було відкрито і як було збережено після внесення до нього змін.

У загальному випадку створений Visio-документ може бути записаний у вигляді файлу, що має одне з розширень:

* **.vsd** (рисунок, діаграма або схема) — основне розширення, яке застосовують для більшості створених користувачем документів. Із версії Visio 2013 типово використовують **.vsdx** формат;

* **.vss** (фігура) — розширення файлів, у яких збережено стандартні або призначені для користувача бібліотеки Visio-фігур або Visio-рішень. Об'єкти, що зберігаються в цих бібліотеках, — це *майстри*, а тематично пов'язана група таких майстрів, що зберігається на спеціальній панелі, — *трафарет*;

* **.vst** (шаблон) — файл із таким розширенням містить ескізи (шаблон) майбутнього готового документа, хоча принципової різниці файлів із розширенням * .vsd і * .vst немає;

* **.vdx** — діаграма у форматі XML;
* **.vsx** — фігура XML;
* **.vtx** — шаблон XML;
* **.vsl** — надбудова;
* **.vsdx** — OPC/XML діаграма;
* **.vsdm** — OPC/XML діаграма, що містить макрос.

Елементи вікна Microsoft Visio

Щоб запустити програму Visio, потрібно:

- натиснути на кнопку *Пуск*, яка розташована в лівому нижньому куті екрана, та вибрати *Програми — Microsoft Office — Microsoft Office Visio;*
- правою кнопкою миші клацнути значок *Visio* на Панелі завдань, вибрати команду *Створити — Microsoft Visio Drawing*.

Після завантаження Visio на екрані з'являється вікно *Створити* зі списком восьми основних категорій: *Інженерний відділ, Бізнес, Блок-схема, Карти та плани поверхів, Загальні, Мережа, Програми і бази даних, Планування* (рис. 12.1). Після категорії шаблонів у центральній частині вікна відображатимуться піктограми відповідних шаблонів:

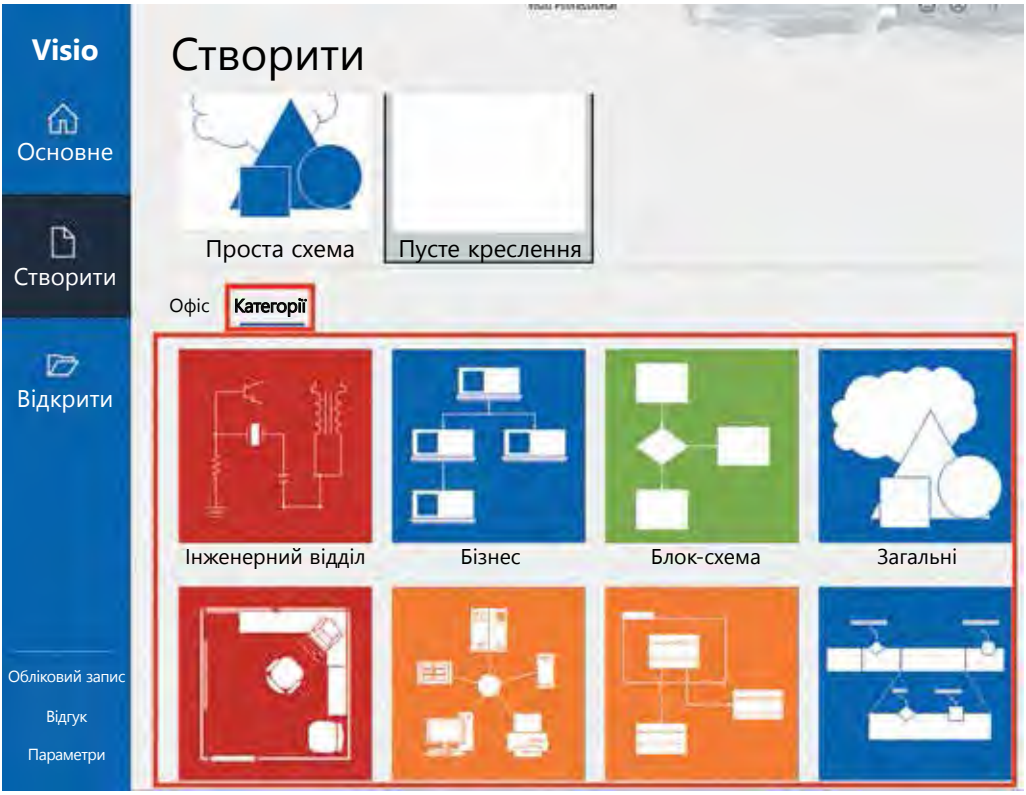


Рис. 12.1

- *Інженерний відділ* — шаблони для проєктування інженерних та електричних схем, технічних установок тощо (рис. 12.2);
- *Бізнес* — шаблони для побудови діаграм і графіків, організаційних діаграм, схеми аудиту, схеми керування якістю, дерева помилок, схеми потоку створення вартості та ін. (рис. 12.3);
- *Блок-схема* — шаблони для побудови схем (рис. 12.4);
- *Карти та плани поверхів* — шаблони для побудови планів будівель, ділянок, приміщень, робочих місць, комунікацій, дорожніх схем, планів місцевості та ін. (рис. 12.5);
- *Загальні* — шаблони з основними графічними примітивами, які використовують для створення рисунків (рис. 12.6);
- *Мережа* — шаблони для проєктування схем локальних або глобальних комп'ютерних мереж (рис. 12.7);

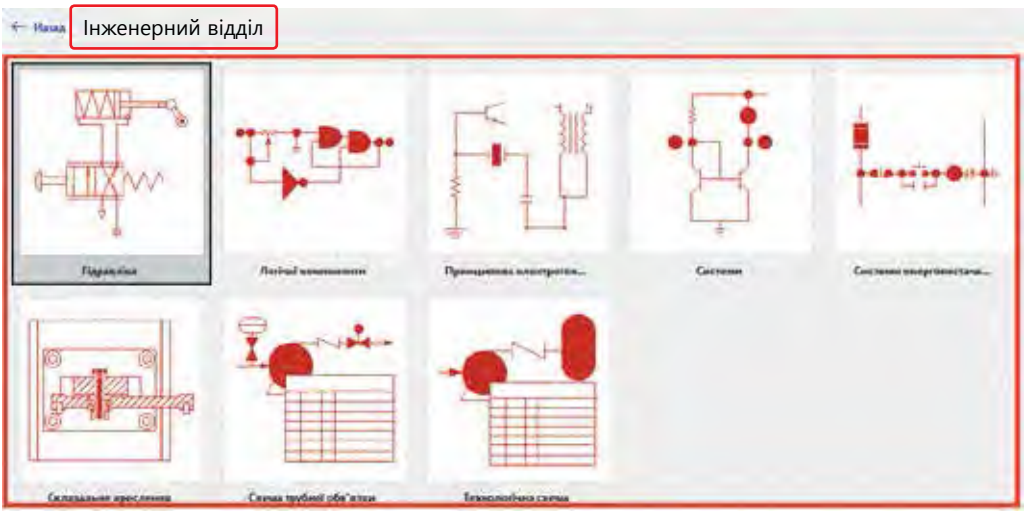


Рис. 12.2

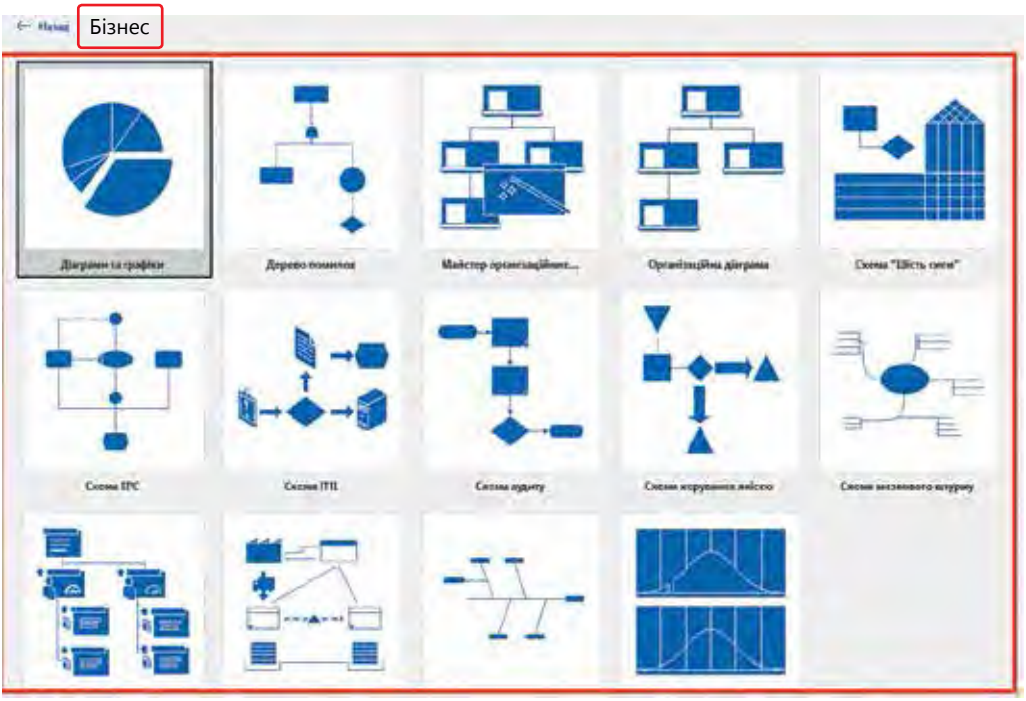


Рис. 12.3

- *Програми і бази даних* — шаблони для створення схем, пов'язаних із роботою програмного забезпечення, візуалізації структури баз даних, карт і структури вебсайтів тощо (рис. 12.8);
- *Планування* — містить шаблони для побудови схем процесів, залежних від часу (рис. 12.9).

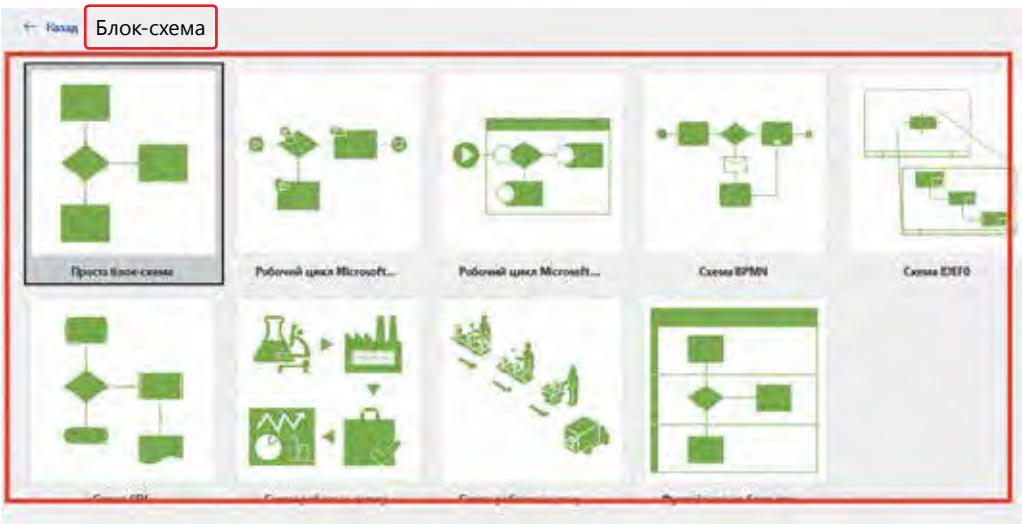


Рис. 12.4

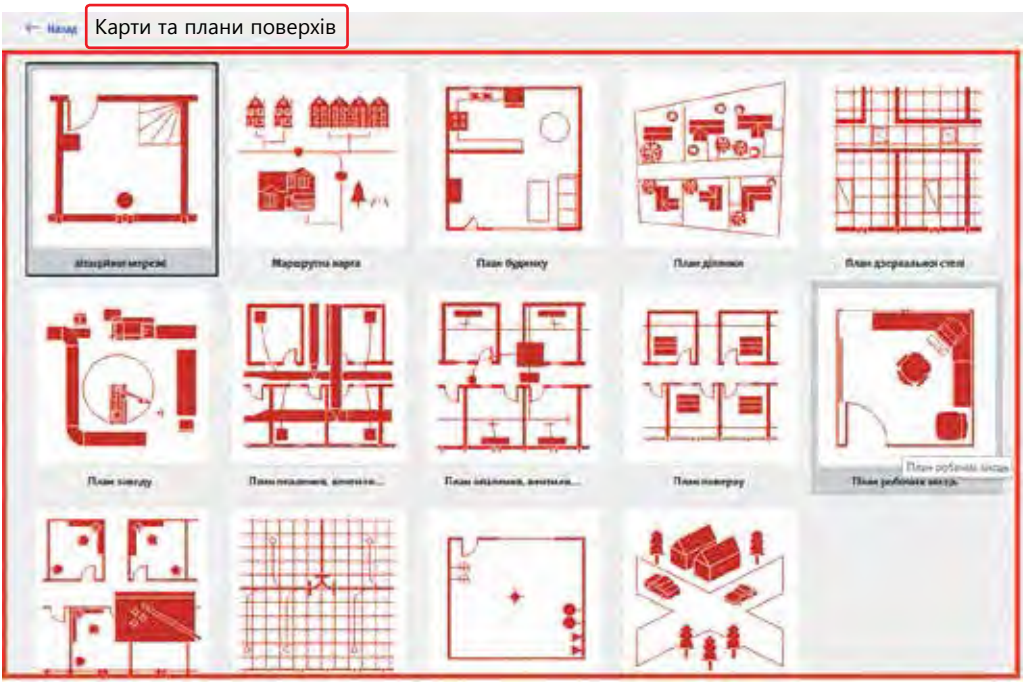


Рис. 12.5

Вікно програми Visio містить такі елементи (рис. 12.10):

1. Панель швидкого доступу до функцій, які часто використовують.

2. Рядок заголовка **з кнопками керування розмірами вікна.**

Рис. 12.6

Рис. 12.7

3. Вкладки: *Файл, Основне, Вставлення, Конструктор, Дані, Процес, Рецензування, Подання, Довідка.*

4. Стрічка — набір інструментів у верхній частині вікна програми.

5. Поле трафаретів містить усі відкриті для цього документа набори фігур і команди для відкривання нових.

6. Робоче поле (аркуш рисунка), в яке можна вставляти фігури, графічні елементи і текстові блоки.

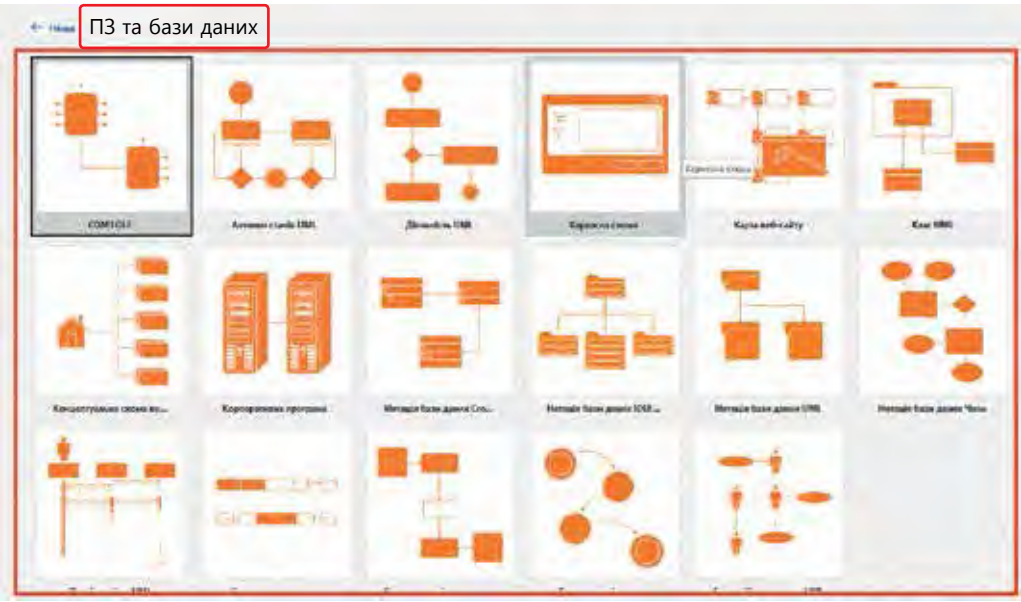


Рис. 12.8

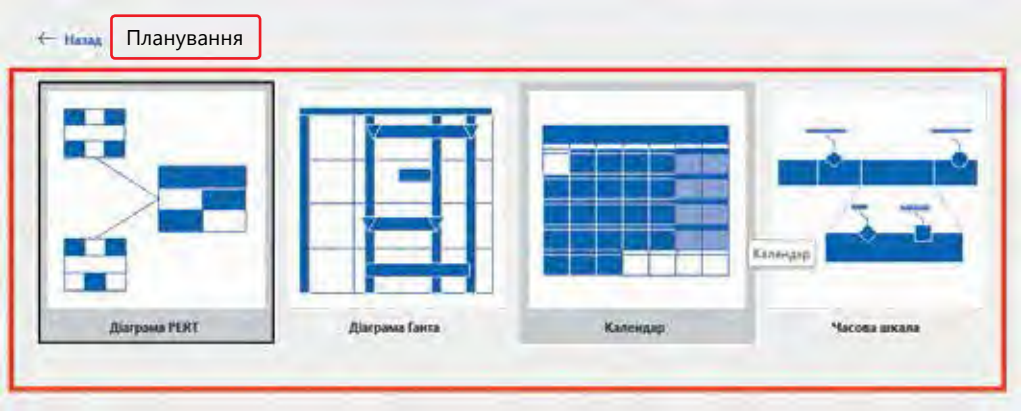


Рис. 12.9

7. Лінійки — для визначення положення зображення щодо координатних осей, початок якого завжди розташований в лівому нижньому кутку сторінки.

8. Сторінки — внизу аркуша рисунка розташовано ярлички аркушів Visio, зліва — кнопки прокручування ярличків аркушів.

9. Рядок стану — нижній рядок екрана MS Visio, у лівій частині якого подано короткі характеристики виділених об'єктів: ширина, висота, кут нахилу, зміни за X і Y, а в правій — номер сторінки і кількість сторінок у документі.

10. Смуги прокручування розташовані праворуч і внизу аркуша рисунка.

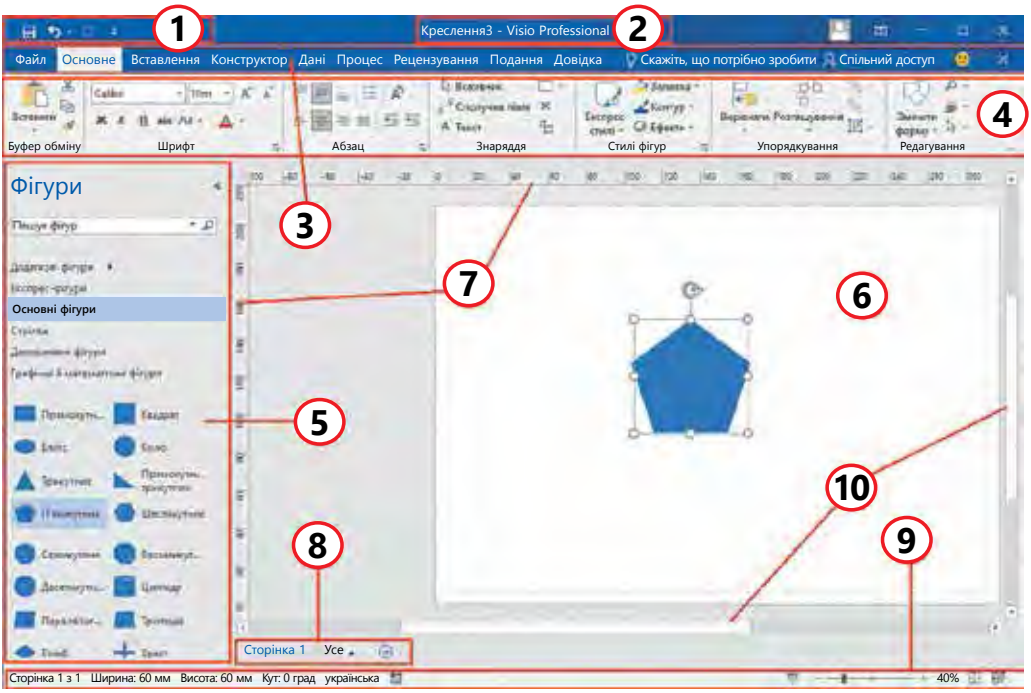


Рис. 12.10

Типово у вікні відображаються дев'ять постійних вкладок:

- **Файл** — команди для роботи з документом: створювати, відкривати, записувати, змінювати властивості документа, друкувати тощо;
- **Основне** — основні інструменти для форматування та редагування тексту, команди, що дають змогу виконувати різні корисні дії з однією фігурою або групою фігур та їхніми елементами. Тут можна змінювати порядок розташування фігур, групувати, об'єднувати, нахиляти, перевертати;
- **Вставлення** — команди, що дають змогу вставити нову сторінку, ілюстрації, коментар, частини схеми, гіперпосилання, об'єкт;
- **Конструктор** — для керування параметрами сторінки та налаштування основних елементів дизайну усього документа;
- **Дані** — команди для роботи з даними;
- **Процес** — команди для перевірки, імпорту та експорту схеми;
- **Рецензування** — інструменти, за допомогою яких можна додати до документа примітки, налаштувати переклад, перевірити орфографію та граматику за встановленими словниками тощо;
- **Подання** — команди, що дають змогу керувати відображенням на екрані зображення та елементів інтерфейсу, макроси;
- **Довідка** — містить довідкову інформацію.

Для полегшення роботи з фігурами та зручності орієнтації на *робочому полі* рисунка відображено сітку. Частота сітки змінюється автоматично у разі

збільшення або зменшення масштабу відображення рисунка. Вздовж лівого і верхнього країв робочого поля розташовано лінійки, які слугують для визначення положення графічних об'єктів щодо координатної сітки, початок якої стандартно міститься в лівому нижньому куті аркуша.

У робочому полі може відображатися тільки одна сторінка, типовий розмір якої відповідає формату А4 (210×297 мм). Однак документ може містити кілька сторінок (як-от книга MS Excel), перехід між якими здійснюють за допомогою ярличків. Додати нову сторінку можна одним зі способів:

- натиснути на кнопку зі знаком +;
- натиснути комбінацію клавіш *Shift+F11;*
- вкладка *Вставлення — Створити сторінку;*
- клацнути правою кнопкою миші на ярличку наявної сторінки і вибрати в контекстному меню команду *Вставити*, у вікні *Параметри сторінки* зазначити ім'я сторінки, фон, одиниці вимірювання.

Кожна нова сторінка документа стандартно переймає розмір, орієнтацію, масштаб, одиниці виміру, зсув тіні й параметри сітки поточної сторінки, яка відображається у вікні документа. Будь-який із цих параметрів можна змінити за допомогою діалогового вікна *Параметри сторінки,* яке завжди відкривається після застосування команди *Вставити,* перш ніж нову сторінку буде додано до документа. Крім того, у контекстному меню є команди: *Видалити, Перейменувати, Дублювати* і *Змінити порядок сторінок*. Порядок сторінок можна також змінити, перетягнувши ярлички за допомогою миші. Для швидкого переходу до потрібної сторінки документа потрібно натиснути на відповідний ярличок лівою кнопкою миші.

Щоб використовувати один з інструментів, треба його активізувати — натиснути ліву кнопку миші на відповідній символьній піктограмі. Відтак активізований інструмент можна застосовувати для операцій у робочій ділянці сторінки. Біля деяких кнопок інструментів є трикутник (▼), який показує, що з кнопкою пов'язаний не один, а кілька інструментів. Потрібно клацнути кнопкою миші на трикутнику, щоб на екрані розкрилася панель конкретного інструменту.

Вікно редагування являє собою аркуш із накладеною на нього сіткою з вертикальних і горизонтальних ліній. Сітка є зручним засобом для позиціонування готових фігур або малювання, її розмір відповідає налаштуванням сторінки (вкладка *Параметри сторінки*). Частота сітки змінюється автоматично, як тільки змінюється масштаб відображення рисунка, а в надрукованому документі її не видно.

Сторінка Visio-документа є основним робочим простором для розроблення документів. Visio-документ може містити необмежену кількість таких сторінок. Це дає змогу збирати в один файл кілька графічних зображень, тематично пов'язаних між собою, причому кожна сторінка може мати різні параметри.

Сторінка документа має набір основних властивостей: ім'я, розміри, фон, одиниця виміру, орієнтація. Щоб визначити ці параметри, потрібно у вкладці

Конструктор вибрати *Параметри сторінки* або натиснути комбінацію клавіш *Shift+F5* (рис. 12.11).

Вікно *Параметри сторінки* містить макі вкладки:

- *Параметри друку* — установки поточного принтера; також можна вибрати розміри аркуша, його орієнтацію і масштаб (рис. 12.12);

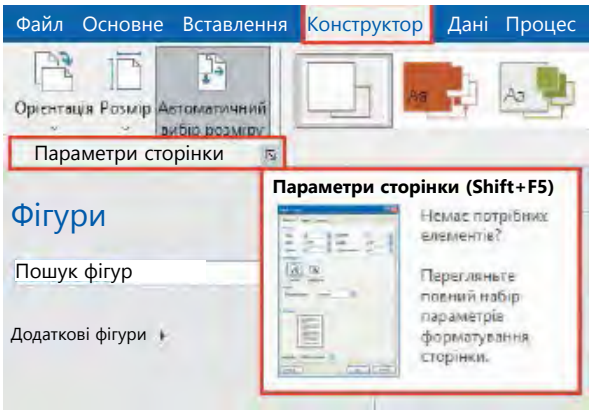


Рис. 12.11

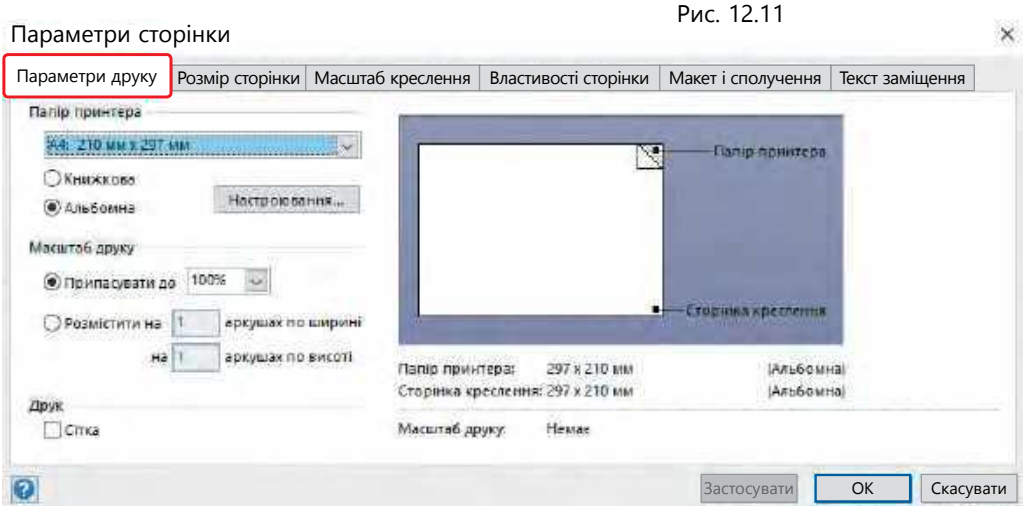


Рис. 12.12

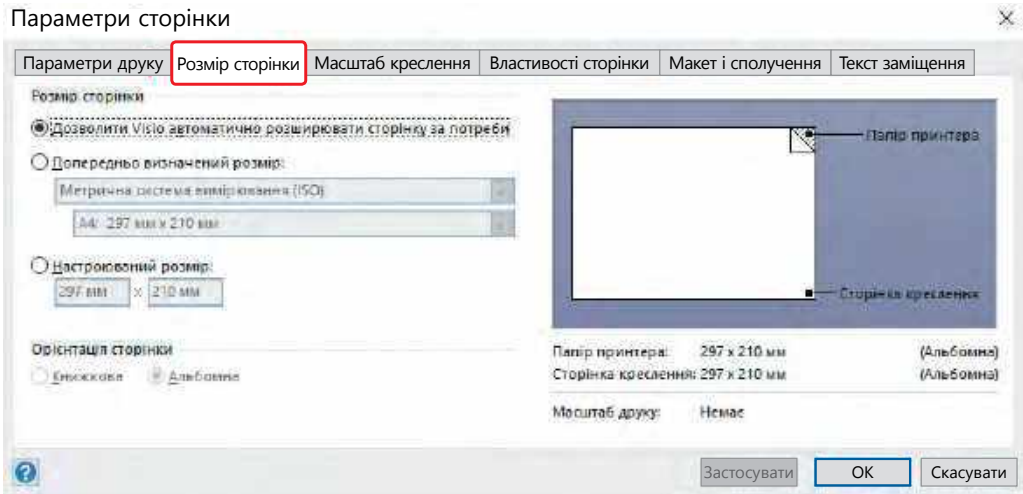


Рис. 12.13

- *Розмір сторінки* — потрібні розміри та орієнтація сторінки (книжкова або альбомна) (рис. 12.13);
- *Масштаб креслення* — масштаб зображення, можна вибрати зі стандартних або встановити користувацький формат (рис. 12.14);

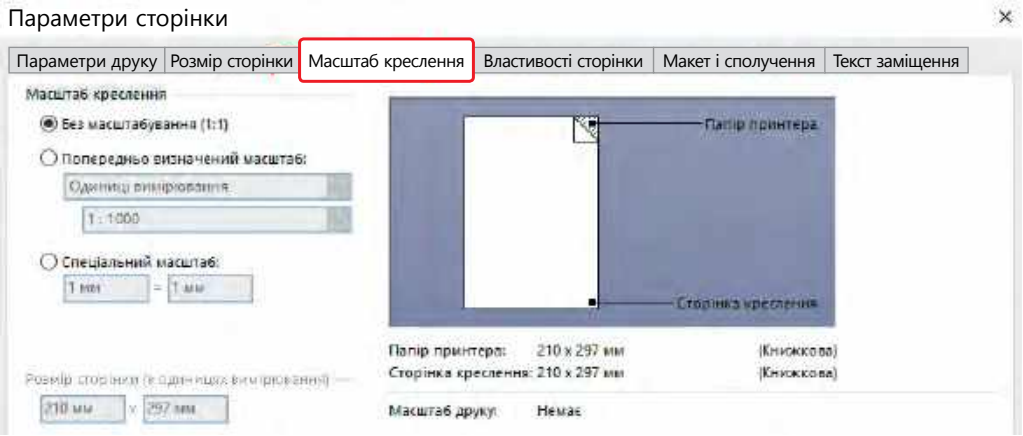


Рис. 12.14

- *Властивості сторінки* — ім'я та фон сторінки, одиниці вимірювання (рис. 12.15);

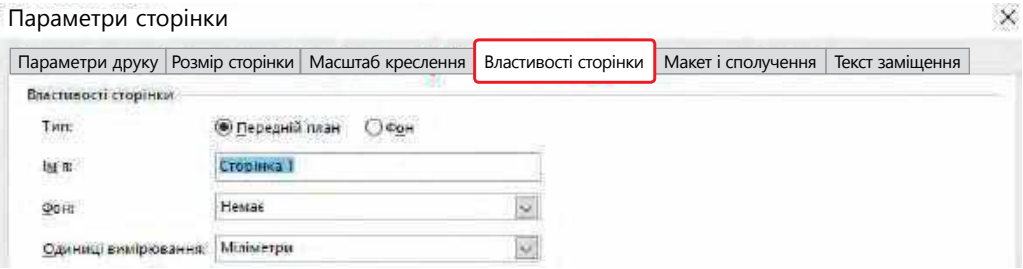


Рис. 12.15

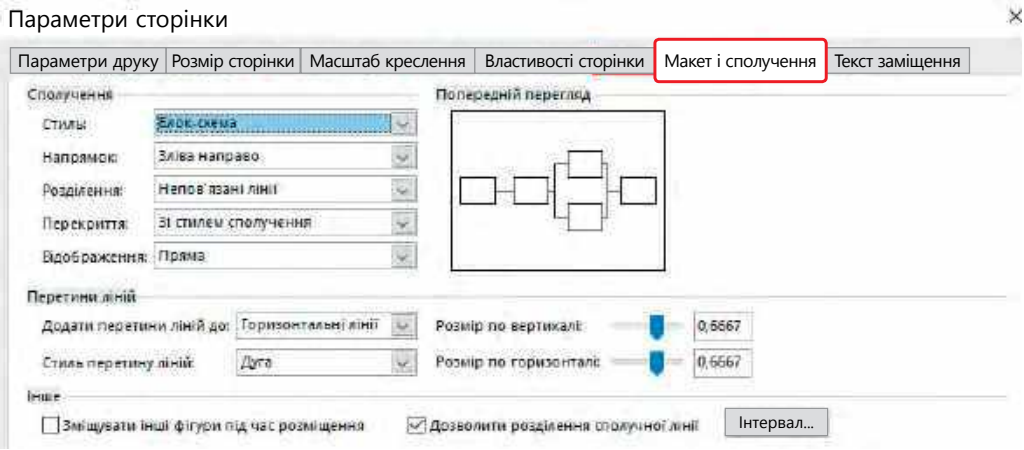


Рис. 12.16

- *Макет і сполучення* — стиль з'єднання фігур і об'єктів у документі (рис. 12.16);
- *Текст заміщення* — назва та опис тексту (рис. 12.17).

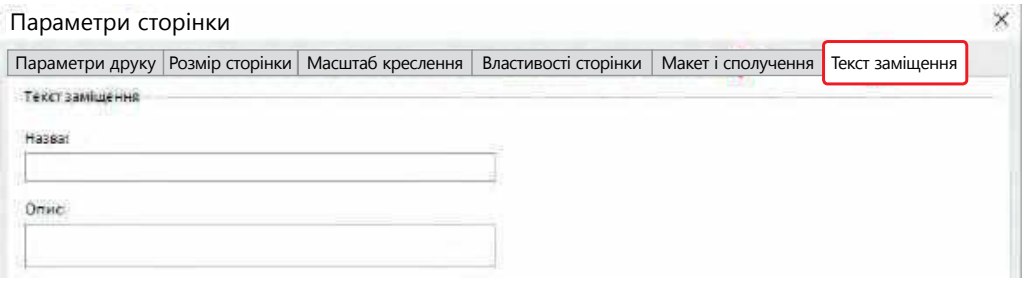


Рис. 12.17

Керування масштабом зображення

Інструменти MS Visio, призначені для керування масштабом зображення, є традиційними для графічних редакторів. Уміле використання цих інструментів дає змогу значно полегшити створення документа: коригування зображень, малювання дрібних деталей рисунка, збільшення точності прив'язування та багато іншого.

У MS Visio передбачено кілька способів зміни масштабу. У вікні діалогу *Масштаб* (рис. 12.18) (команда *Масштаб* на вкладці **Подання**) можна вибрати один із його варіантів або визначити в полі *Відсотки* будь-який інший зручний для роботи масштаб перегляду (в діапазоні від 1 до 2914 %).

Рис. 12.18

Команди, що керують параметрами масштабу, містяться на вкладці **Подання** — *Масштаб:*

- *Фактичний розмір* — сторінка відображається у фактичному розмірі та відповідає масштабу 100 %;
- *За шириною сторінки* — сторінка відображається у свою повну ширину в межах вікна редагування;
- *Сторінка повністю* — сторінка відображається повністю в межах вікна редагування.

Для продуктивнішої роботи з документами в MS Visio передбачено низку можливостей для зміни масштабу зображення за допомогою клавіатури й миші. Зокрема, якщо натиснути і тримати комбінацію клавіш *Ctrl+Shift,* вказівник миші перетвориться на лупу, а права і ліва кнопка миші стануть інструментами ступеневої зміни масштабу зображення. Натиснувши ліву кнопку миші, можна збільшити масштаб зобра-

ження, натиснувши праву — зменшити (так само, як за допомогою інструментів *Зменшення масштабу* і *Збільшення масштабу,* але в іншій пропорції). Треба також зазначити, що незалежно від того, збільшується або зменшується масштаб, зображення буде у тому місці, де стоїть вказівник миші. Тож можна докладніше розглянути частину зображення у віддаленому місці сторінки.

Трафарети

Одним із найцікавіших засобів MS Visio, його «родзинкою» є *Трафарет* (англ. stencil). Це особлива бібліотека, в якій зібрано тематично пов'язані фігури, створені для подальшого їх використання під час роботи з іншими документами. Кожен трафарет містить різні *майстри* фігур, графічні й допоміжні елементи.

Після версії MS Visio 2013 замість назви «трафарет» частіше вживають терміни «набір фігур», «фігури» або «колекція елементів».

Кількість і склад відображених трафаретів залежать від вибраного під час створення документа шаблону. Саме трафарет формує основні елементи призначеного для користувача інтерфейсу, оскільки він містить фігури, за допомогою яких інші користувачі зможуть конструювати свої рисунки. Трафарети подано з лівого боку вікна в полі *Фігури*. Черговість розташування трафаретів — в алфавітному порядку. У кожен момент часу видно фігури тільки одного трафарету (внизу поля *Фігури*). Заголовок цього трафарету виділено кольором (рис. 12.19).

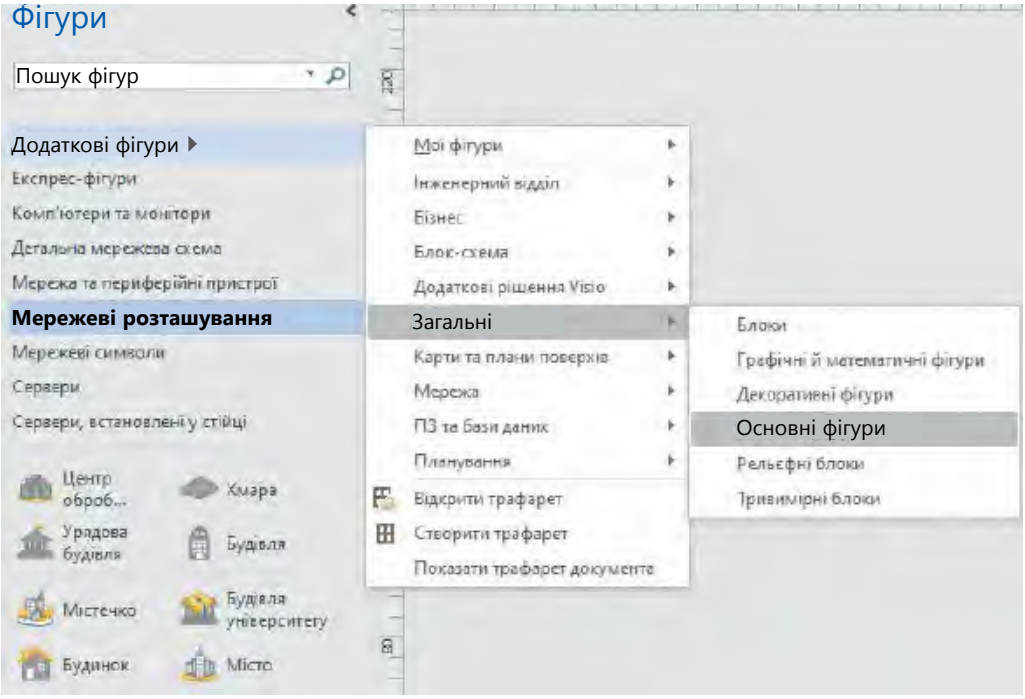
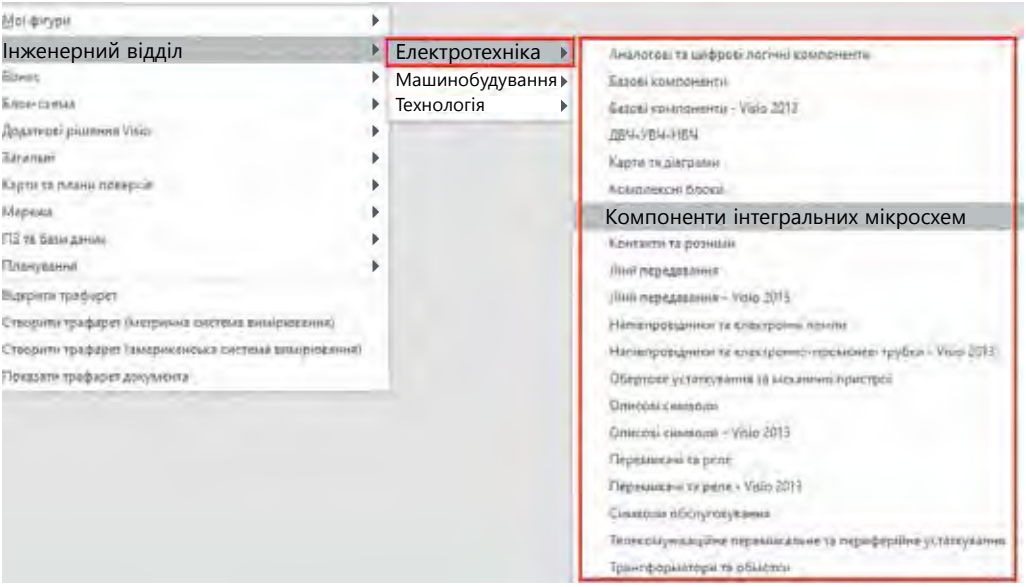


Рис. 12.19

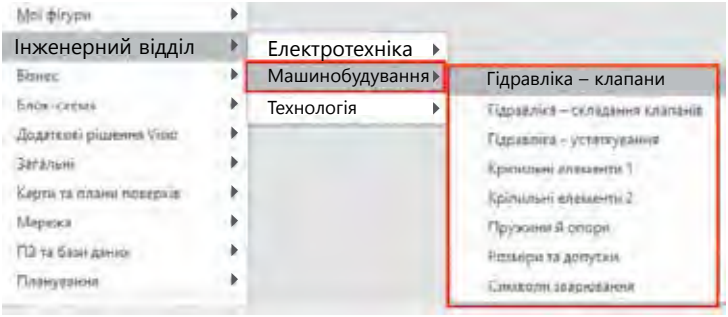
Найпростіший і найзручніший спосіб створити рисунок — скористатися відповідним трафаретом, тобто перенести майстер необхідної фігури з його панелі на сторінку. При клацанні на заголовку іншого трафарету панель поточного згортається в смугу заголовка, а вибраний трафарет розкривається на всю нижню частину області фігур.

У MS Visio є дуже великий вибір різноманітних за призначенням стандартних трафаретів, об'єднаних спільною темою. У цих трафаретах зібрано ті фігури, які часто використовують під час створення типових зображень. Наприклад, незамінною бібліотекою даних для розроблення електричних кіл буде категорія *Електротехніка* (рис. 12.20, *а*), а помічником у конструюванні й дизайні машин — категорія *Машинобудування* (рис. 12.20, *б*).

Також є набір специфічних трафаретів, які можна використовувати для розроблення будь-якого рисунка, — *Основні фігури*.



а



б

Рис. 12.20

12.3. ГРАФІЧНІ ОБ’ЄКТИ В MS Visio

Властивості графічних об’єктів

Поняття фігури в MS Visio. У MS Visio будь-який геометричний об’єкт називають *фігурою* (англ. shape). Це може бути елементарна фігура — лінія, ламана крива, дуга, сплайн (кілька об’єднаних дуг) або складна замкнута фігура, отримана в результаті групування декількох окремих об’єктів. Крім того, термін «*фігура*» може посилатися на об’єкт будь-якої зовнішньої програми. Можливі способи створення Visio-фігур:

- за допомогою інструментів редагування;
- конвертація метафайлів у фігури;
- імпорт графіки з інших програм;
- сканування зображень;
- адаптування наявних фігур для особистого користування.

Будь-яка фігура в MS Visio передбачає асортимент формул, які задають її атрибути та режими. Наприклад, фігури, призначені для проєктування дизайну офісу, можна наділити «числовою» інформацією і пов’язати її з іменами співробітників або фірмами — постачальниками обладнання.

Розглядаючи анатомію фігури, можна виділити низку її особливостей:

1. Фігура може бути *замкнутою* або *розімкнутою*, що впливає тільки на її заповнення.

2. Фігура може бути *одновимірною* або *двовимірною*, що впливає на її вигляд.

3. Перед створенням фігури потрібно вибрати *тип керування для фігури*, тобто вирішити, як елементи призначеного для користувача інтерфейсу будуть пов’язані з геометрією фігури і чи зможуть вони візуально представити їхній спосіб взаємодії з цією фігурою.

4. Перед створенням фігури необхідно вирішити, чи буде фігура самостійною, чи міститиметься в *групі*. Це рішення вплине на те, як користувач редагуватиме групу й кожну фігуру окремо.

Замкнуті і розімкнуті фігури. Фігуру створюють за допомогою ліній, дуг або сегментів кривої, які називають *шляхами*. Кожен із цих шляхів може бути замкнутим або розімкнутим, і фігури відповідно — замкнутими або розімкнутими. Фігури мають різні властивості. Наприклад, тільки замкнуту фігуру можна заповнити кольором або візерунком, а розімкнуту фігуру можна форматувати за допомогою вузлів тощо.

Щоб фігура мала замкнуті контури, початкова і кінцева вершини мають з’єднуватися в одній точці. Іноді замкнута і розімкнута фігури візуально не відрізняються, тож у таких випадках звертають увагу на те, чи заповнено фігуру кольором або візерунком. Розімкнуті шляхи фігури можна замкнути за допомогою інструменту *Олівець*.

Одновимірні і двовимірні фігури. У Visio будь-який об'єкт є фігурою, зокрема рисунки і текст. Усі фігури поділяють на два типи: одновимірні (1-D shapes) і двовимірні (2-D shapes).

В *одновимірній фігурі,* коли її виділено, є початкова і кінцева точки (рис. 12.21). За допомогою таких фігур з'єднують інші фігури.

Рис. 12.21

У *двовимірній фігурі* немає початкової і кінцевої точок, її неможливо використовувати для з'єднання інших фігур (рис. 12.22).

Рис. 12.22

Маркери та спеціальні точки фігури. *Маркери* — це спеціальні мітки, які призначені для візуальних змін параметрів фігури, її положення, зв'язків з іншими фігурами (рис. 12.23). Щоб відображалися маркери, фігуру потрібно виділити. У MS Visio є кілька типів маркерів.

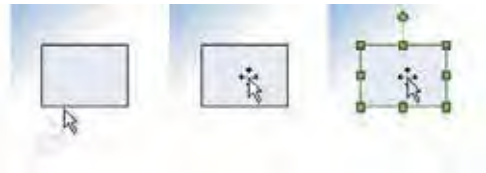
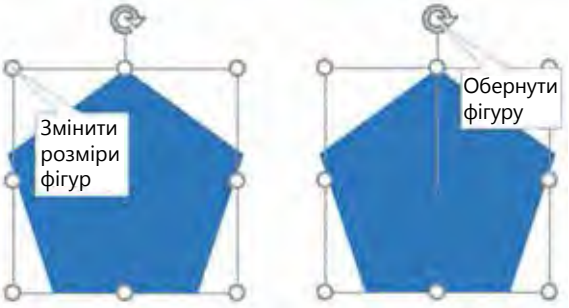


Рис. 12.23

Маркери виділення (інструмент *Покажчик*) розташовані в ключових точках прямокутника, в який вписано двовимірну фігуру. Кожна фігура має вісім маркерів виділення (рис. 12.24, *а*) для зміни горизонтальних і вертикальних розмірів фігури. Щоб пропорційно змінити розмір фігури, потрібно навести вказівник миші і, натискаючи кнопку миші, переміщувати маркер.

Маркер початку відрізняється від маркера кінця тільки візуально, а функціонально вони є ідентичними. За послідовного з'єднання кількох одновимірних фігур усі точки з'єднання стають початковими маркерами.

Маркер обертання має вигляд невеликого кола і призначений для повороту фігури на довільний кут щодо центру обертання. Можливість повороту фігури на довільний кут є однією з найбільш зручних і корисних можливостей MS Visio (рис. 12.24, *б*).



а *б*

Рис. 12.24

Для видалення фігури потрібно виділити її і натиснути клавішу *Delete*. Для переміщення фігури треба навести на неї вказівник миші (він матиме вигляд чотириспрямованої стрілки) і перетягнути фігуру в необхідну позицію. Розмір, положення і кут повороту фігур можна також змінювати за допомогою панелі *Розмір і розташування* (рис. 12.25).

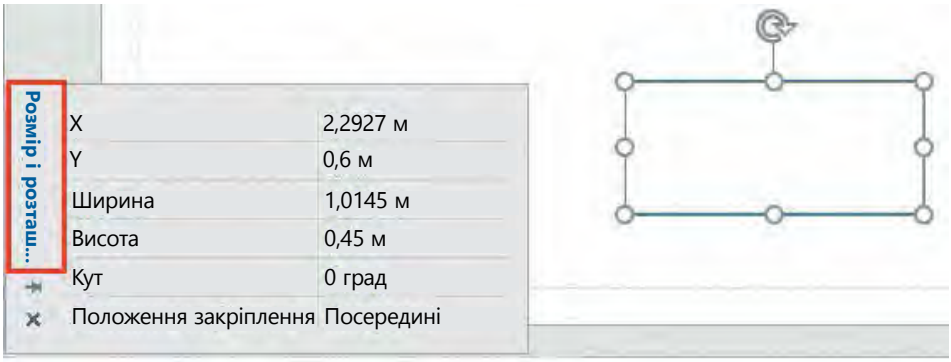


Рис. 12.25

Маркери редагування призначені для зміни положення вершин фігури і кривизни її сторін. Ці маркери бувають двох типів — маркери контролю і маркери вершин (рис. 12.26).

Маркери контролю — круглі маркери зеленого кольору, з'являються під час редагування фігури за допомогою інструменту *Олівець* і змінюють параметр кривизни вибраного відрізка.

Точки з'єднання (англ. connection pointer) позначено синіми хрестиками. Вони дають змогу спростити процес з'єднання різних фігур (або фігур і з'єднувачів). У MS Visio є три види точок з'єднання:

Рис. 12.26

- *внутрішня* — використовують у тих рисунках, де з'єднують одновимірні і двовимірні фігури (вузли та з'єднувачі в блок-схемах);
- *зовнішня* — для склеювання двовимірних фігур, дає змогу об'єднати фігури так, щоб вони зберігали свій порядок, кут нахилу одна щодо одної у разі переміщення або зміни однієї з фігур;
- *змішана* — об'єднує будь-які фігури, поєднуючи в собі можливості двох видів точок.

Усі точки з'єднання, які за замовчуванням має фігура, є внутрішніми і розташовуються у вузлових точках фігури: вершинах, центрах сторін, геометричному центрі, містах найбільш частого з'єднання з іншими фігурами.

Щоб захистити фігуру від випадкових змін, необхідно на вкладці *Розробник* у групі *Оформлення фігури* натиснути кнопку *Захист* (рис. 12.27).

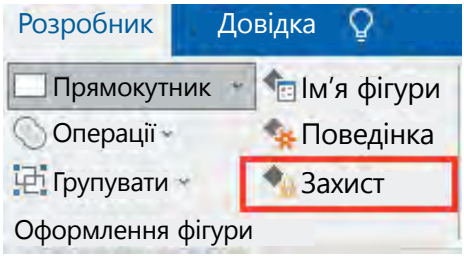


Рис. 12.27

Створення і редагування графічних об’єктів

MS Visio має велику колекцію готових фігур, розбитих на групи, — понад пів сотні трафаретів. Багато готових фігур можна знайти в Інтернеті. Крім того, можна змінити будь-яку наявну фігуру.

У процесі роботи з фігурою має бути ввімкненим режим *Вказівник* (рис. 12.28).

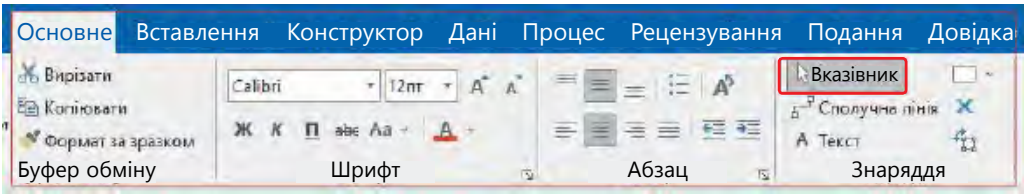


Рис. 12.28

Щоб додати на аркуш фігуру, потрібно на панелі трафаретів *Фігури* вибрати трафарет і елемент (*майстершейп*) та перетягнути лівою кнопкою миші на робоче поле (рис. 12.29). У створеній фігурі можна змінити розміри, властивості й параметри відображення на сторінці. Після натискання на *Олівець*

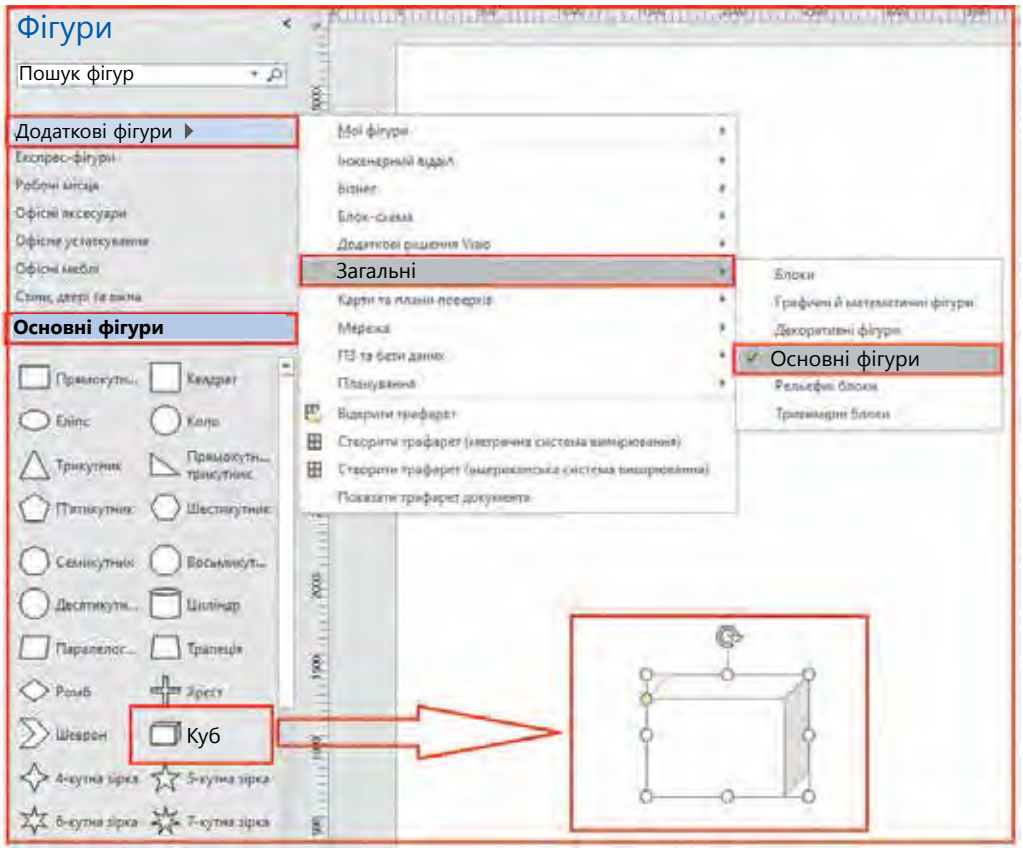


Рис. 12.29

маркери виділення, розташовані у вершинах прямокутника, перетворяться на маркери вершини, бічні маркери стануть точками контролю, а вказівник миші набере вигляду олівця. Переміщуючи маркери, можна змінювати фігуру. За допомогою інструменту *Олівець* додають нові або вилучають зайві вершини фігури. Щоб видалити вершину, потрібно виділити її і натиснути клавішу *Delete*. Після цього сусідні з видаленою вершини буде з'єднано відрізком. Щоб додати нову вершину, треба натиснути клавішу *Ctrl* і, тримаючи її, вказати інструментом *Олівець* точку на контурі фігури, куди потрібно вставити вершину. Разом із новою вершиною на половині відстані до найближчої наявної вершини фігури додасться нова точка контролю.

Інструмент *Лінія* дає змогу малювати прямі й ламані лінії (рис. 12.30). Під час використання цього інструменту вказівник миші має вигляд хрестика, підкресленого лінією, з усіх маркерів виділених фігур залишаються тільки маркери вершин, а сторінку поділено на вузли. Якщо натиснути і тримати клавішу *Shift*, то кут нахилу лінії завжди буде 45°.

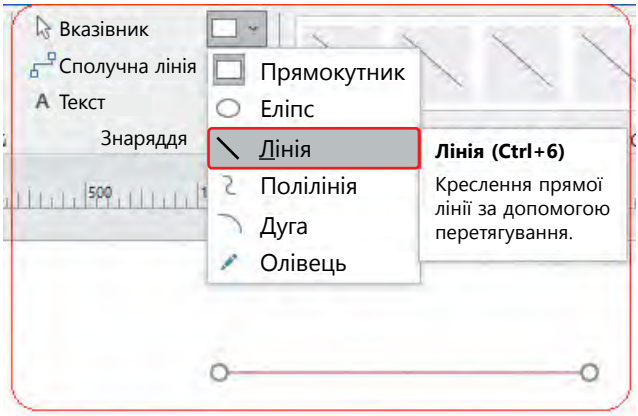


Рис. 12.30

Для малювання кривих у MS Visio використовують інструмент *Полілінія*. Під час роботи вказівник миші перетвориться на хрестик із зображенням ламаної лінії. В результаті виходить сплайн, що складається з безлічі точок контролю, з'єднаних дугами (рис. 12.31, *а*). Криву можна відредагувати, додавши або видаливши зайві точки контролю: виділити (крива буде бузкового кольору) (рис. 12.31, *б*) і натиснути клавішу *Delete* (рис. 12.31, *в*). Щоб додати точки контролю, треба натиснути клавішу *Ctrl* і, тримаючи її, клацнути лівою кнопкою миші. У точку вставлення буде додано маркер вершини.

а *б* *в*

Рис. 12.31

За допомогою інструменту *Дуга* малюють параболічні дуги (вказівник матиме вигляд хрестика із дугою). Інструмент *Еліпс* і *Прямокутник* дає змогу створювати прямокутники, еліпси, квадрати і кола. Щоб намалювати коло або

квадрат, потрібно вибрати відповідний інструмент, натиснути клавішу *Shift* і, тримаючи її, задати мишею розмір контуру. Вказівник набере вигляду, відповідно, хрестика з еліпсом або прямокутником, а в разі натиснутої клавіші *Shift* — хрестика з колом або квадратом.

Перетягування та з'єднання фігур

Для створення організаційної діаграми потрібно вибрати на вкладці *Файл — Створити* категорію *Бізнес — Організаційна діаграма,* а потім натиснути кнопку *Створити* (рис. 12.32).

Далі перетягнути фігуру з трафарету *Фігури — Стрічка фігури організаційної діаграми* (наприклад, Стрічка адміністратора) до сторінки документа та відпустити кнопку миші. Якщо підвести та тримати вказівник миші на фігурі, з'являться сині стрілки та мініпанель, які вказують місце, куди необхідно помістити фігуру (рис. 12.33).

Щоб додати інші фігури, необхідно вибрати їх на мініпанелі й перетягнути на синю стрілку. Нову фігуру буде з'єднано з першою (рис. 12.34).

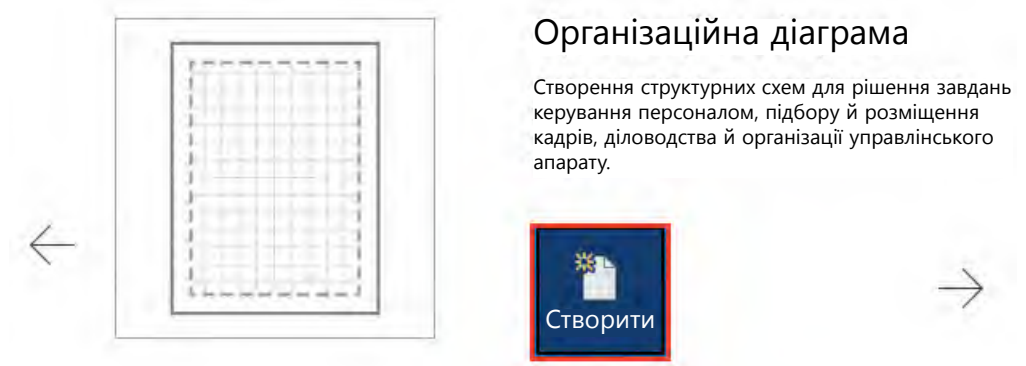


Рис. 12.32

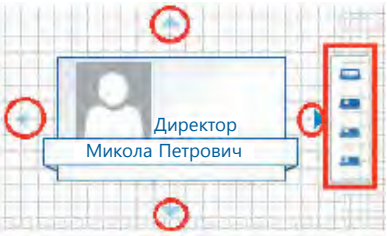


Рис. 12.33

Рис. 12.34

Форматування графічних об'єктів

Форматування лінії. Лінія є найпростішим самостійним елементом фігури, і її завжди можна додатково налаштувати. Для зміни параметрів лінії у Visio є набір спеціальних інструментів на вкладці *Основне* в групі *Стилі*. Основні інструменти фігури *Лінія* (рис. 12.35) такі:

- *Заливка* — вибирають колір лінії з наведеної палітри (типово колір лінії є чорним) (рис. 12.36);

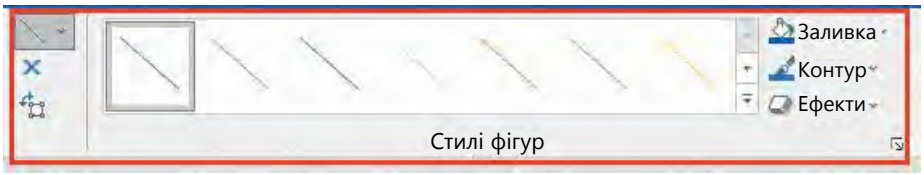


Рис. 12.35

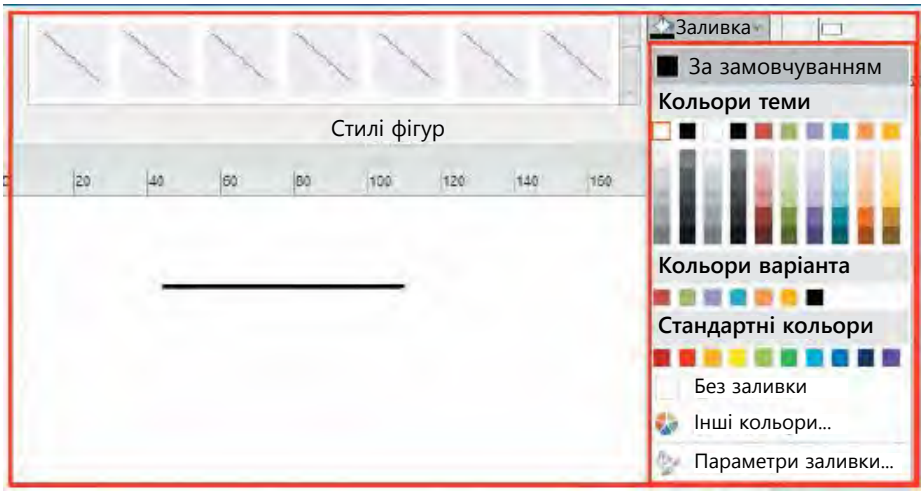


Рис. 12.36

- *Товщина* (стандартно дорівнює 0,72 pt);
- *Пунктир* — тип лінії (типово — суцільна);
- *Кінці лінії* — різні види закінчення лінії (можуть бути у вигляді стрілок різних форматів) (рис. 12.37);
- *Круглі кути* — згладжування кутів фігури (стандартне значення — 0 in).

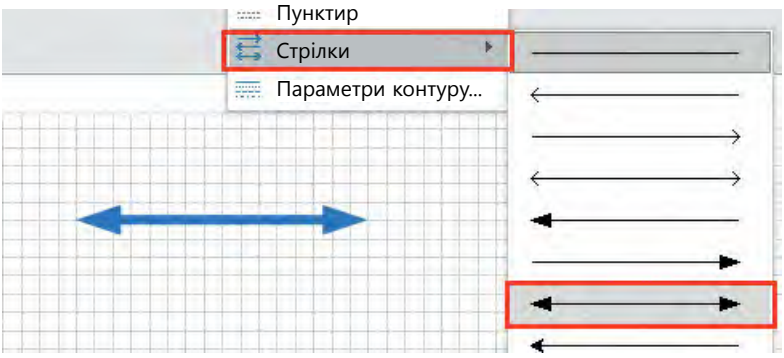


Рис. 12.37

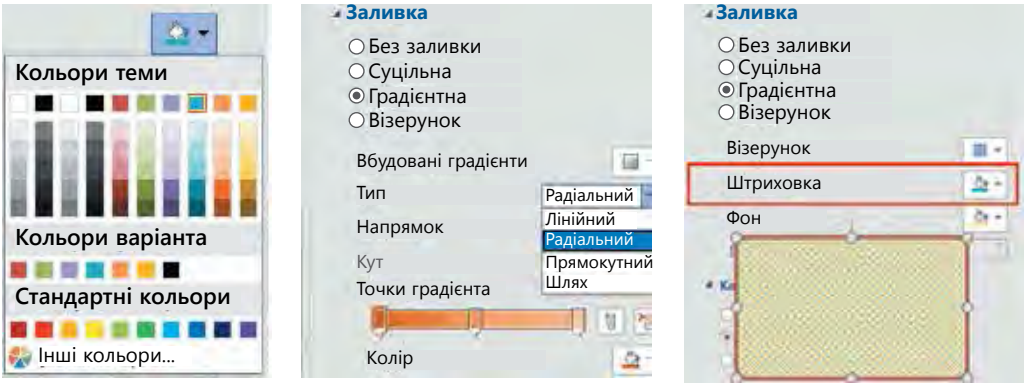
Керування кольором. Колір є одним із головних елементів формату фігури. Кольорове оформлення фігури можна розділити на дві частини: колір заповнення (заливка) і колір меж.

У MS Visio є можливість керувати параметрами кольорів усього документа. Схеми кольорів містять встановлені кольори ліній, заливки, фону й тексту.

Стандартно в MS Visio використовують схему кольорів *Black & White*. Щоб встановити іншу схему, потрібно вибрати колір на відповідній панелі (рис. 12.38, *а*).

Для керування кольором фігур у MS Visio є спеціальний набір інструментів: *Колір заливки, Колір лінії* і *Колір тексту.* Панелі інструментів *Колір лінії* і *Колір заливки* мають відповідно кнопки *Без лінії* і *Без заливки,* що дають змогу зробити відповідний елемент фігури невидимим.

Стандартні кольори, подані в палітрі, можна редагувати, зокрема додавати нові. У діалоговому вікні *Заливка* можна не тільки вказати колір заливки, а й вибрати візерунок, приміром *градієнтну заливку* (рис. 12.38, *б*) або *штрихування* (рис. 12.38, *в*).



а *б* *в*

Рис. 12.38

У цьому ж вікні можна встановити невидимість і властивості тіні фігури, вибрати колір тіні, її візерунок і колір візерунка, наприклад двомірним фігурам надати тривимірний вигляд.

Копіювання і переміщення фігур

Копіювання і переміщення — це операції, які користувач постійно виконує під час створення будь-яких зображень. Наприклад, створення стандартної фігури можна розглядати як копіювання майстра з панелі трафаретів на аркуші рисунка, а будь-яке пересування фігури аркушем — як операцію переміщення.

Розрізняють власне копіювання, коли скопійовані дані зберігаються в документі, й переміщення, коли вихідні дані видаляють, а скопійовані вставляють в іншому місці документа. Процес переміщення може полягати у простій зміні поточного розташування фігури, позиціонуванні або переміщенні в точно задане місце аркуша.

Операції копіювання і переміщення дають змогу значно розширити сферу дії програми Visio завдяки технології зв’язку програм *OLE*. Рисунок, зробле-

ний у Visio, можна використовувати в будь-якій іншій програмі Microsoft Office або програмі, яка підтримує технологію OLE.

Зміна координат фігури. Щоб отримати відомості про точне місце фігури або зазначити, де має розташовуватися фігура, використовують інструмент *Розмір і розташування*. У вікні (рис. 12.39) можна задати поточні координати геометричного центра фігури, а також її висоту, ширину і кут повороту. У полях X і Y вказують потрібні значення координат геометричного центру фігури. Після введення відповідного значення фігура автоматично переміститься в задане положення.

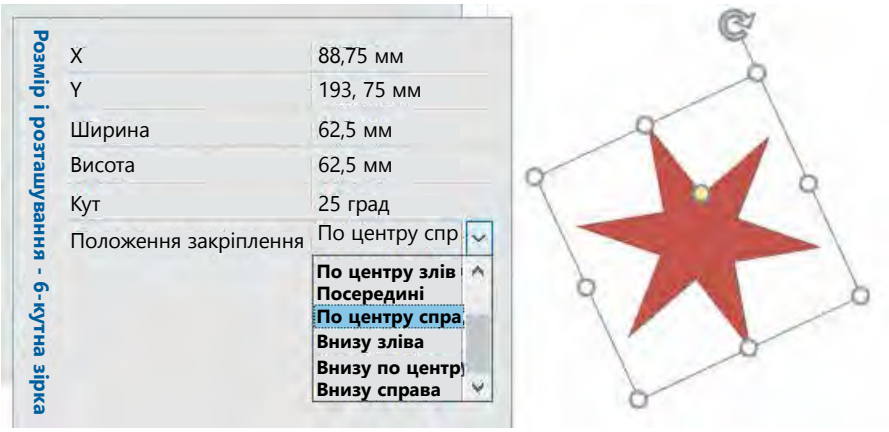


Рис. 12.39

12.4. СТВОРЕННЯ ТЕКСТУ

Текст як фігура. Текст у MS Visio є самостійним об'єктом, який можна форматувати, обертати, переміщувати, перевіряти, щоб знайти можливі помилки. Можливості форматування в MS Visio дещо менші, ніж у спеціалізованих текстових редакторах, однак їх цілком достатньо для створення підписів, пояснень і заголовків, які є основними об'єктами на сторінці документа.

Для створення текстових елементів у MS Visio використовують інструмент *Текст* на вкладці *Основне* у групі *Знаряддя* (рис. 12.40). За його допомогою

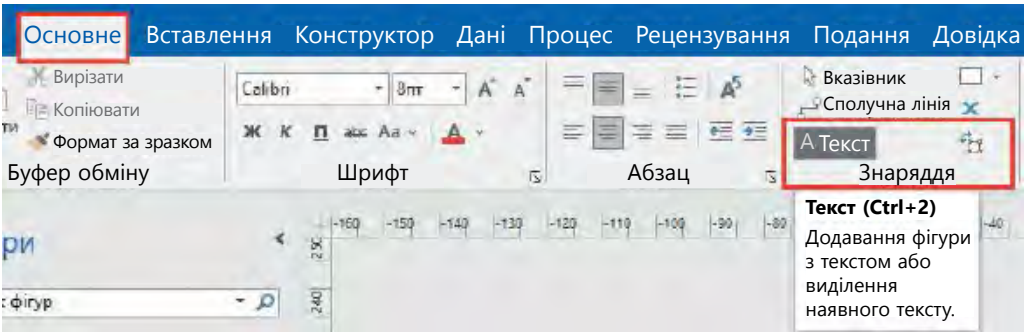


Рис. 12.40

можна вставити текст у будь-яке місце сторінки у вигляді текстового блоку (текстової фігури).

Текстовий блок має деякі властивості звичайної фігури, наприклад маркери виділення, які керують місцем тексту на сторінці. Бічними горизонтальними маркерами задають ширину текстового поля (рис. 12.41, *а*). Якщо ширина поля є більшою, ніж кількість уведених символів, поле складається з одного рядка, якщо меншою — текст автоматично ділиться на рядки (рис. 12.41, *б*).

Рис. 12.41

Текст як елемент фігури. У більшість фігур, зокрема в з'єднувачі, можна вставити текст. Такий текст є елементом фігури і автоматично розташовується в її центрі. Текст у фігуру вставляють за допомогою інструментів *Текст* і *Олівець*. У першому випадку необхідно виділити фігуру, вибрати на вкладці стрічки *Основне* кнопку A *Текст* (або просто двічі клацнути на фігурі мишею) і набрати текст (рис. 12.42). Або ж вибрати інструмент *Олівець* і двічі клацнути на фігурі лівою кнопкою миші. В обох випадках у фігурі відкриється текстове поле, розмір якого залежатиме від розмірів і форми відповідної фігури.

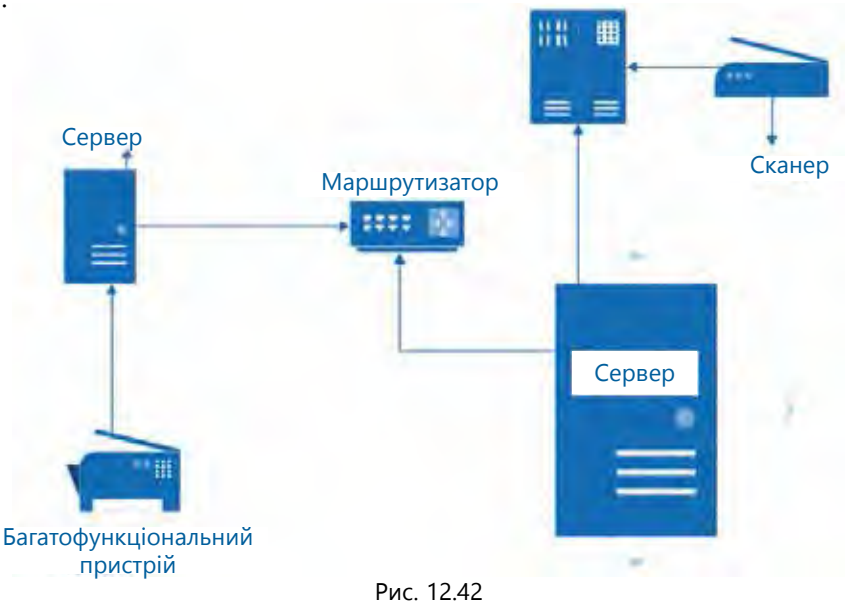


Рис. 12.42

Убудований у фігуру текст має деякі властивості звичайної фігури, наприклад його можна переміщувати та обертати.

Місце розташування текстового блока можна змінювати, перетягуючи його за ввімкненого інструменту *Блок тексту* на вкладці стрічки *Основне* у групі *Знаряддя*.

Шрифт тексту, розмір, вирівнювання, відступи, маркери, табуляцію тощо можна змінювати загальноприйнятими в MS Office засобами форматування:

- кнопки на вкладці стрічки *Основне*, групи *Шрифт*, *Абзац* (рис. 12.43, 12.45); мініпанель форматування (рис. 12.44);

Рис. 12.43

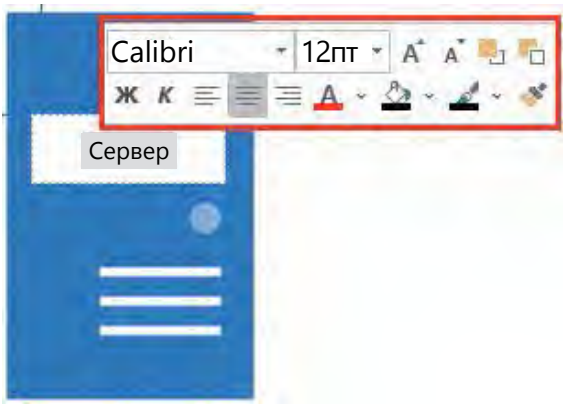


Рис. 12.44

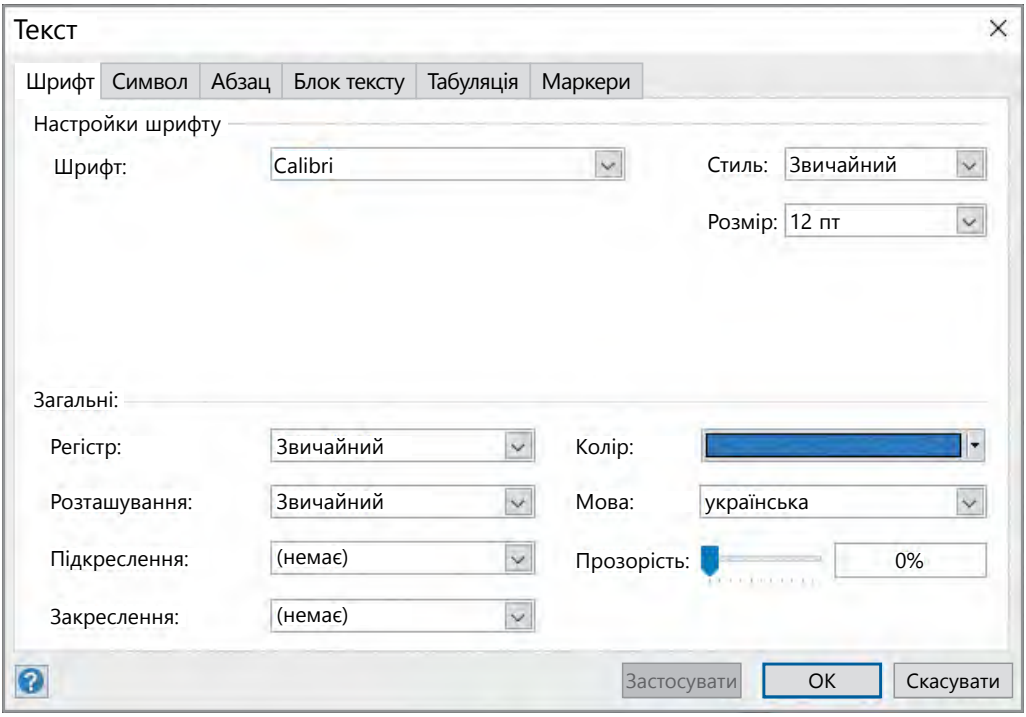
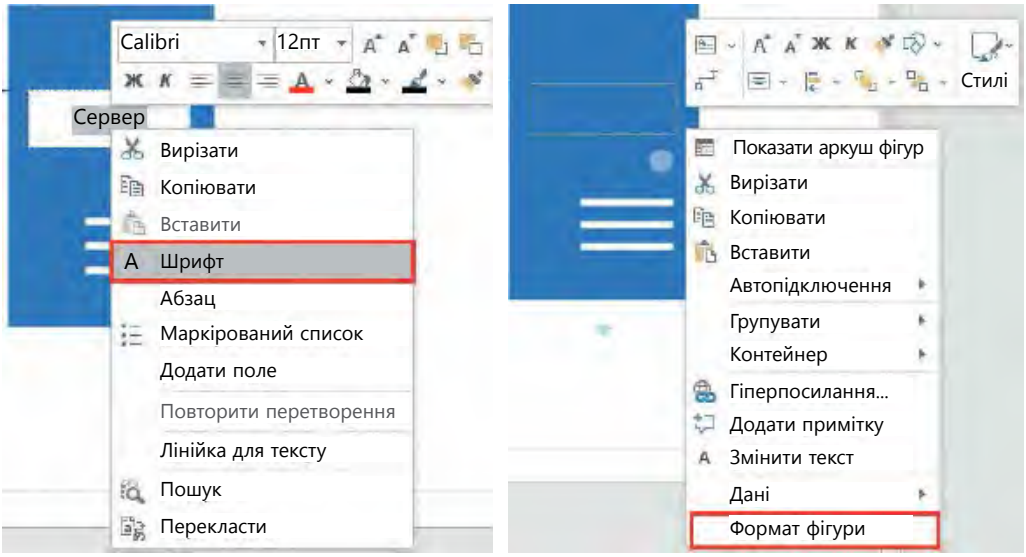


Рис. 12.45

- команди контекстного меню *Шрифт* (рис. 12.46, *а*);
- команда *Формат фігури* в контекстному меню фігури (рис. 12.46, *б*), яка дає змогу змінювати товщину, колір лінії, тип ліній, заливку, візерунок, фон, прозорість, штрихування, тип штрихування, тінь, округлення тощо (рис. 12.47).



а *б*

Рис. 12.46

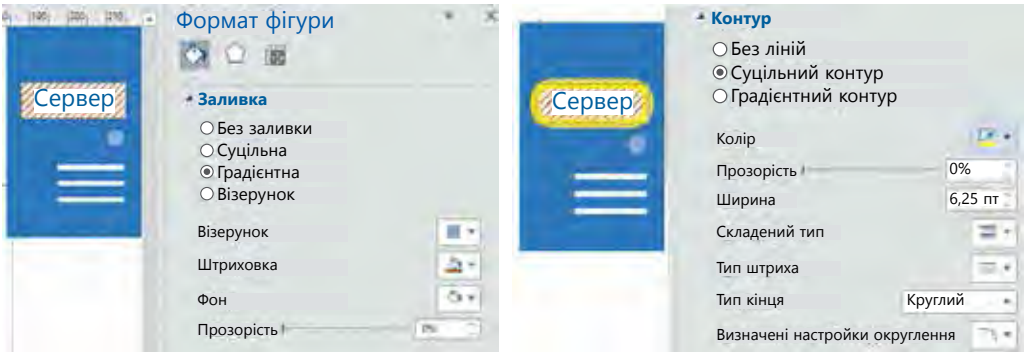


Рис. 12.47

Для нумерації створених елементів потрібно відкрити вкладку *Подання*, вибрати *Доповнення — Додаткові рішення Visio — Нумерація фігур* (рис. 12.48).

Для заощадження часу для однотипного оформлення фігур слугує кнопка *Формат за зразком*.

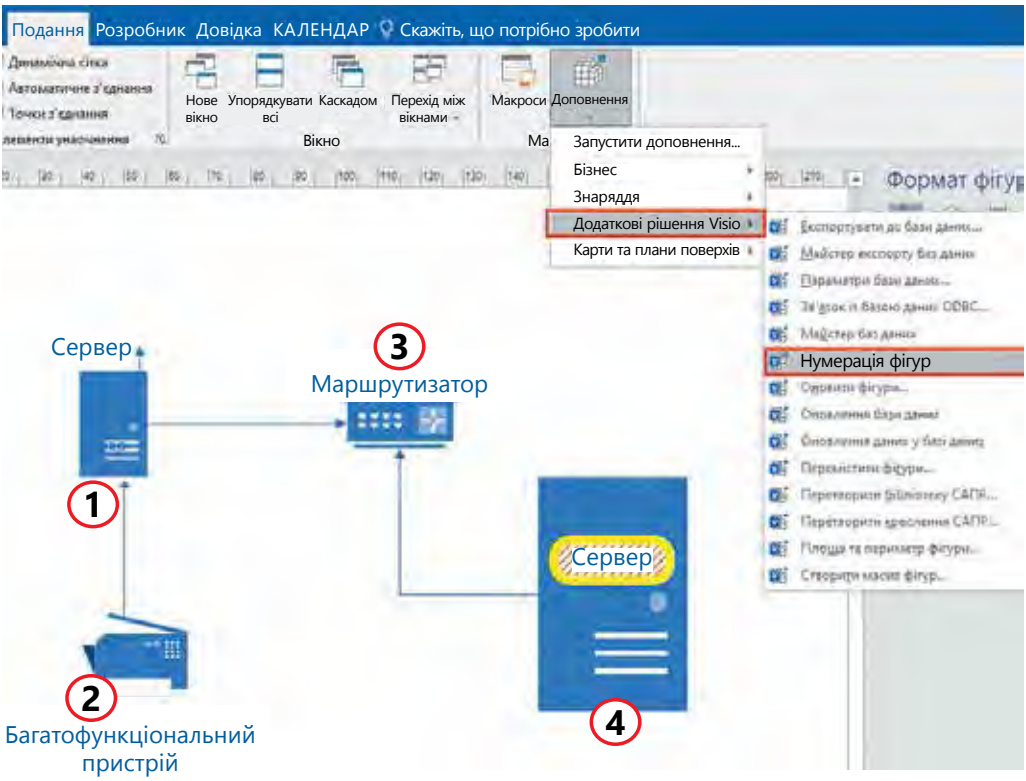


Рис. 12.48

Приклад 1. Створення проєкту «Робочий цикл бізнес-процесів». Найзручніший спосіб почати роботу з Visio — створити документ на основі шаблону.

1. Завантажити програму Microsoft Visio.

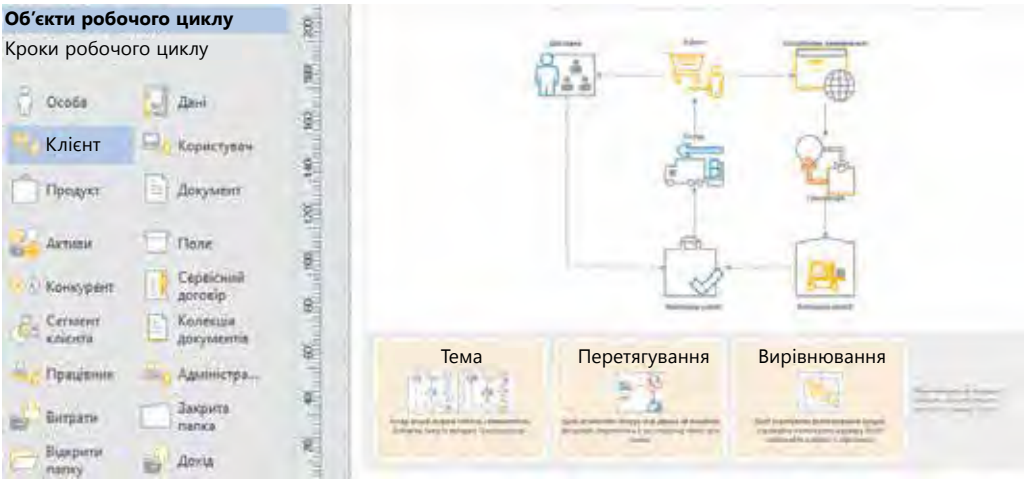


Рис. 12.49

2. Обрати шаблон для роботи категорії *Блок-схема — Схема робочого циклу — Робочий цикл підрозділу,* вибрати потрібний трафарет і натиснути на кнопку *Створити*. З'явиться вікно із шаблоном (рис. 12.49). Після завантаження шаблону на поле завдань *Фігури* завантажуються ті категорії графічних елементів, які можуть знадобитися у процесі створення діаграми, плану або карти вибраного типу.

3. Задати параметри сторінки:

- параметри друку: папір принтера А4, альбомна, масштаб 100 %;
- розмір сторінки: 297×210 мм;
- масштаб креслення: без масштабування (1:1);
- властивості сторінки: ім'я — Ілюстрація бізнес-процесів, фон — немає.

4. Перетягнути елементи з цього набору на робоче поле.

Приклад 2. Створення опису концептуальної моделі підприємства.

1. Вибрати у вікні *Фігури — Додаткові фігури — Блок-схема — Відділ 3-D*.

2. Для побудови в списку вибрати елементи, перетягнути їх на робоче поле і з'єднати один з одним.

3. Відрегулювати розмір або поворот фігури, натиснувши на неї два рази, доки не з'являться «стрілочки» розміру та повороту.

4. Тримати вказівник миші на фігурі, доки не з'являться сині стрілки.

5. Перемістити вказівник на верхню частину синьої стрілки, яка вказує місце, куди необхідно помістити другу фігуру. З'явиться мініпанель, яка містить фігури верхньої частини трафарету.

6. Для поєднання фігур потрібно поставити прапорець у вкладці *Подання — Автоматичне з'єднання*.

7. Далі натиснути на синю стрілочку, щоб задати новий зв'язок (рис. 12.50).

Рис. 12.50

8. Використовуючи інструмент *Текст*, додати підписи до створених елементів (рис. 12.51).

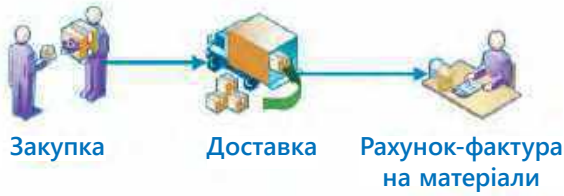


Рис. 12.51

9. Наступним кроком виділити лівою кнопкою миші всі елементи та вибрати команду *Групувати* у вкладці *Основне — Упорядкування — Групувати* (рис. 12.52).

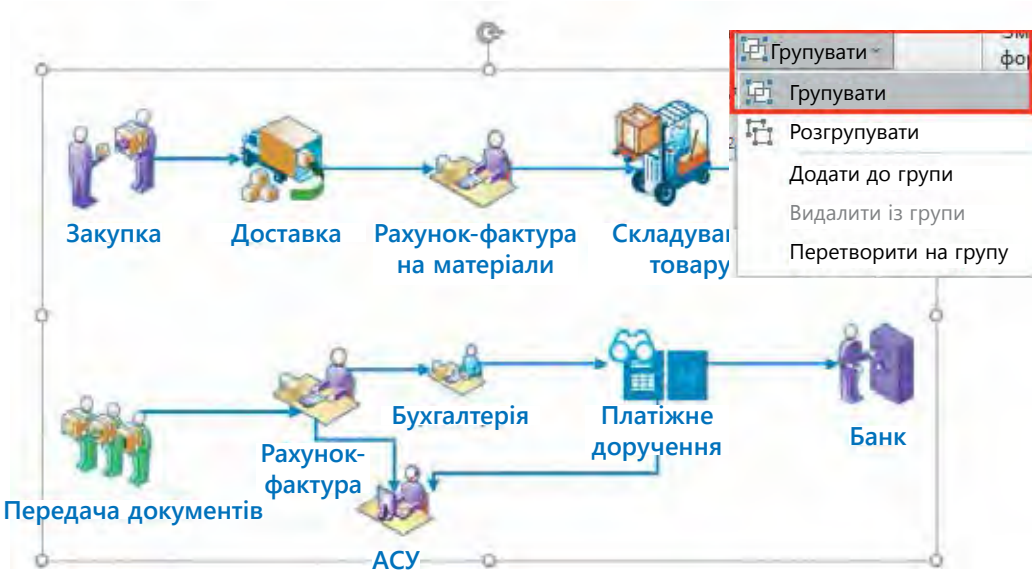


Рис. 12.52

10. За допомогою команди *Формат фігури* можна змінити колір лінії з синього на червоний (рис. 12.53).

11. Для побудованої діаграми потоків робіт можна сформувати звіт із переліком робіт.

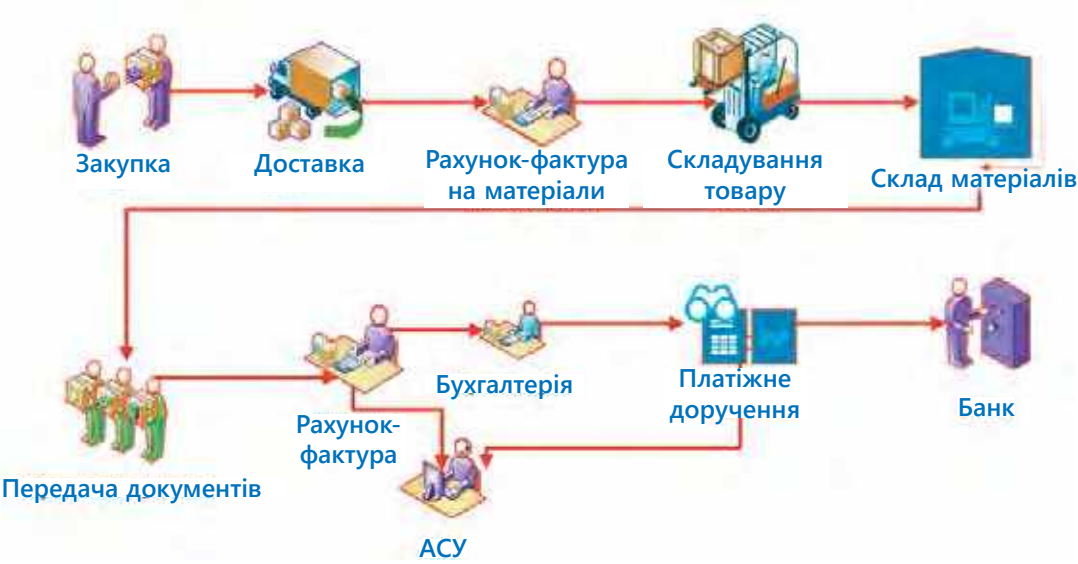


Рис. 12.53

12. У вкладці *Рецензування* натиснути на кнопку Звіти фігури, вибрати *Відомість* і *Створити*.

13. У вікні майстра, в якому вибрати один із варіантів (фігури на всіх сторінках; фігури на поточній сторінці; вибрані фігури; інше) й натиснути на кнопку *Далі* (рис. 12.54).

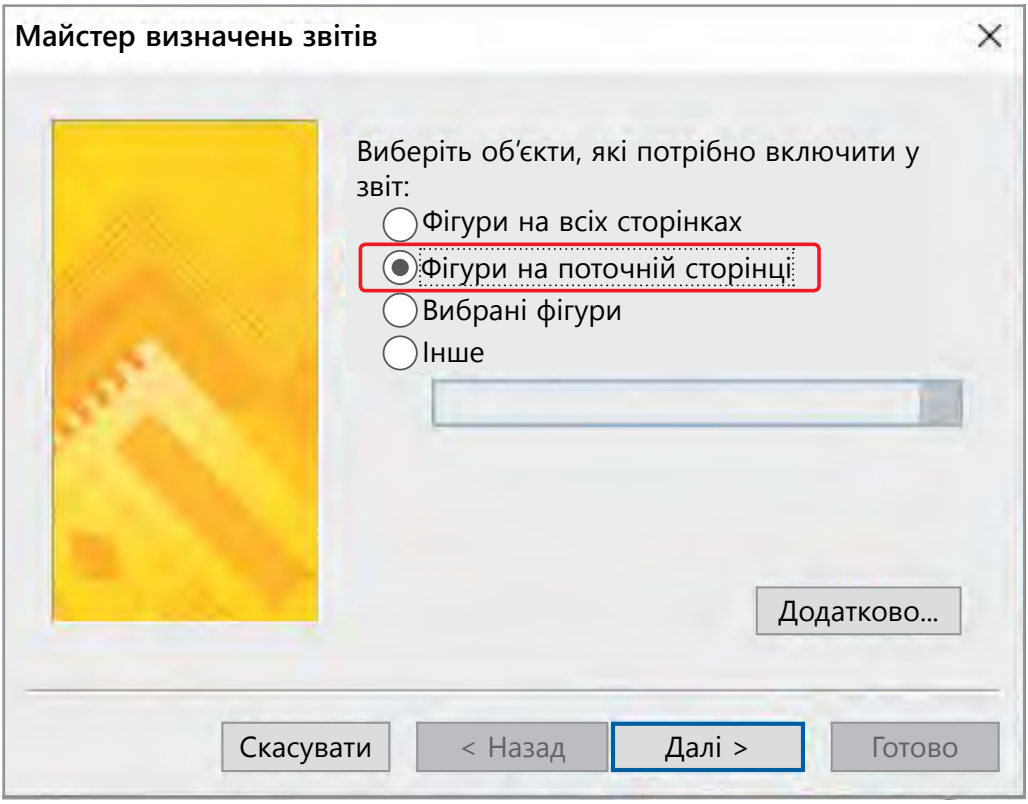


Рис. 12.54

14. На наступному кроці майстра звітів вибрати властивості, які відображатимуться як стовпці звіту (рис. 12.55).

15. Написати назву звіту й натиснути *Далі* (рис. 12.56).

16. Для запуску створеного звіту натиснути на кнопку *Виконати* й вибрати формат звіту, наприклад *Excel*.

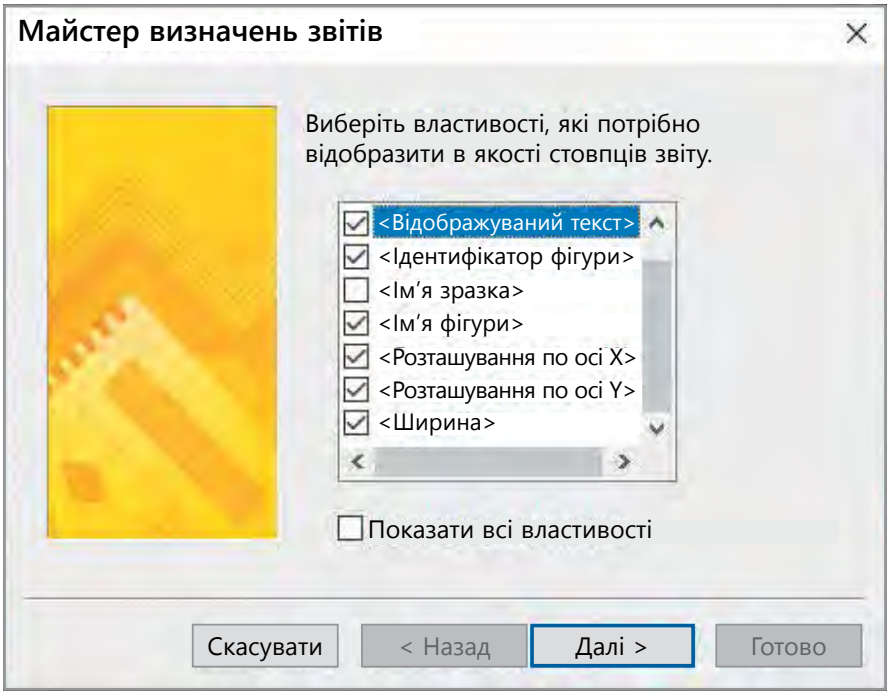


Рис. 12.55

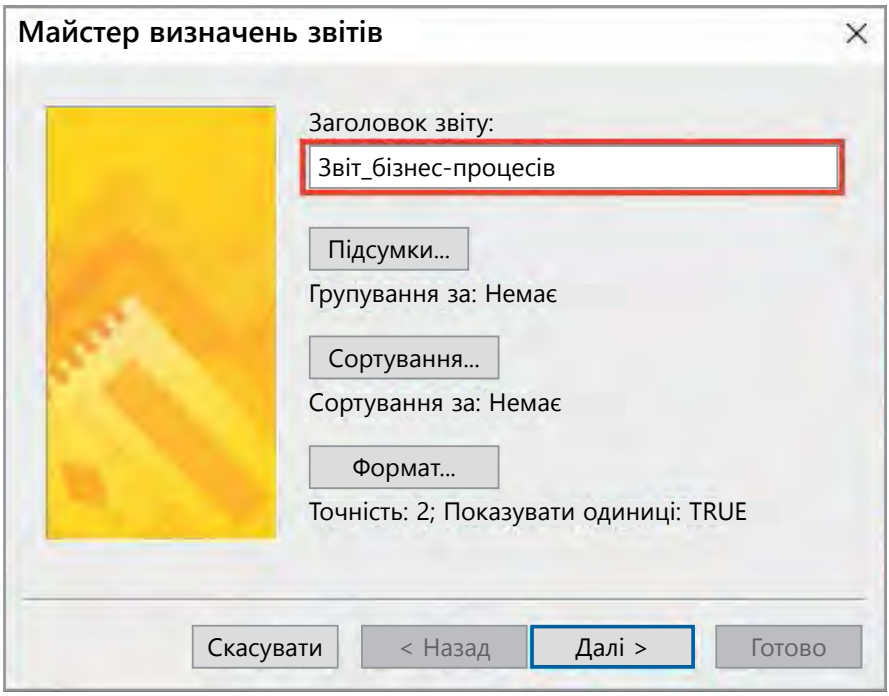


Рис. 12.56

ТЕСТОВІ ЗАВДАННЯ

ПРОГРАМА MS Visio ПРИЗНАЧЕНА ДЛЯ:

1. створення текстових документів
2. створення розрахункових таблиць
3. інформаційних технологій
4. технічних технологій

ЯКИЙ ІЗ ПЕРЕЛІЧЕНИХ ТИПІВ ТРАФАРЕТІВ МОЖНА ВІДРЕДАГУВАТИ?

1. Тільки створені користувачем
2. Будь-який
3. Завантажено лише із сайту підтримки корпорації Майкрософт
4. Постачається лише з пакетом Visio

У ЯКИХ КОМПОНЕНТАХ МОЖНА ЗМІНИТИ ВЛАСТИВОСТІ ЗА ДОПОМОГОЮ КОНТЕКСТНОГО МЕНЮ?

1. Карти й плани
2. Електротехніка
3. Технології
4. Блок-схеми

ЯКУ З КАТЕГОРІЙ МІНІАТЮР (ШАБЛОНІВ) ТРЕБА ВИБРАТИ ДЛЯ СТВОРЕННЯ ЛОГІЧНОЇ СХЕМИ МЕРЕЖІ З ВІДОБРАЖЕННЯМ ОСНОВНИХ ОБ'ЄКТІВ, ЩО ВХОДЯТЬ ДО ЛОКАЛЬНОЇ МЕРЕЖІ?

1. Блок-схема
2. Карта
3. Вебсхема
4. Мережа

Розділ 13
ОБРОБЛЕННЯ ІНФОРМАЦІЇ ЗА ДОПОМОГОЮ ЕЛЕКТРОННИХ ТАБЛИЦЬ. ПРОГРАМА Microsoft Excel

Інформацію в певних предметних галузях, зокрема в економічній, фінансовій, соціальній, доцільно подавати у вигляді таблиць. Для оброблення інформації таблиці у паперовому вигляді використовували задовго до комп'ютерів, як-от конторські книги, гросбухи, рахунки-фактури тощо.

Сьогодні електронні таблиці застосовують у таких сферах:

- економічні, зокрема бухгалтерські, розрахунки;
- інженерні задачі;
- статистичне оброблення даних;
- пошук оптимальних значень параметрів;
- побудова графічних залежностей і діаграм;
- оброблення результатів експерименту.

Для оброблення інформації в електронних таблицях створено комплекс програмних засобів — *табличний процесор*.

Першу електронну таблицю *VisiCalc* для персонального комп'ютера *Apple II* було розроблено в 1979 р. Відтоді з'явилося багато електронних таблиць різного рівня складності, зокрема *Lotus 1-2-3, SuperCalc, Abacus*. Прикладами вільного програмного забезпечення є електронна таблиця *LibreOffice Calc,* що входить до складу *LibreOffice,* і *OpenOffice.org Calc,* що належить до *OpenOffice.org*.

Електронну таблицю *Microsoft Excel,* що є складником *Microsoft Office,* широко застосовують у різних сферах, тому розглянемо докладніше на її прикладі будову, функціональні можливості й принципи роботи електронних таблиць.

13.1. ОСНОВНІ ЕЛЕМЕНТИ ЕЛЕКТРОННИХ ТАБЛИЦЬ

Вікно *Microsoft Excel* (рис. 13.1) містить такі основні елементи:

1. Панель швидкого доступу до функцій, які часто використовують, — *Зберегти, Скасувати, Повторити, Настроїти панель швидкого доступу.*
2. Рядок заголовка, на якому розташовані (зліва направо) назви поточного документа та програми.
3. Стандартні кнопки — *Згорнути, Поновити / Розгорнути, Закрити.*
4. Рядок з іменами вкладок, на якому стандартно відображаються **Файл, Основне, Вставлення, Макет сторінки, Формули, Дані, Рецензування, Подання, Довідка.**
5. Стрічка, що містить усі команди електронних таблиць (понад 1200).
6. Поле адреси поточної клітинки A1.

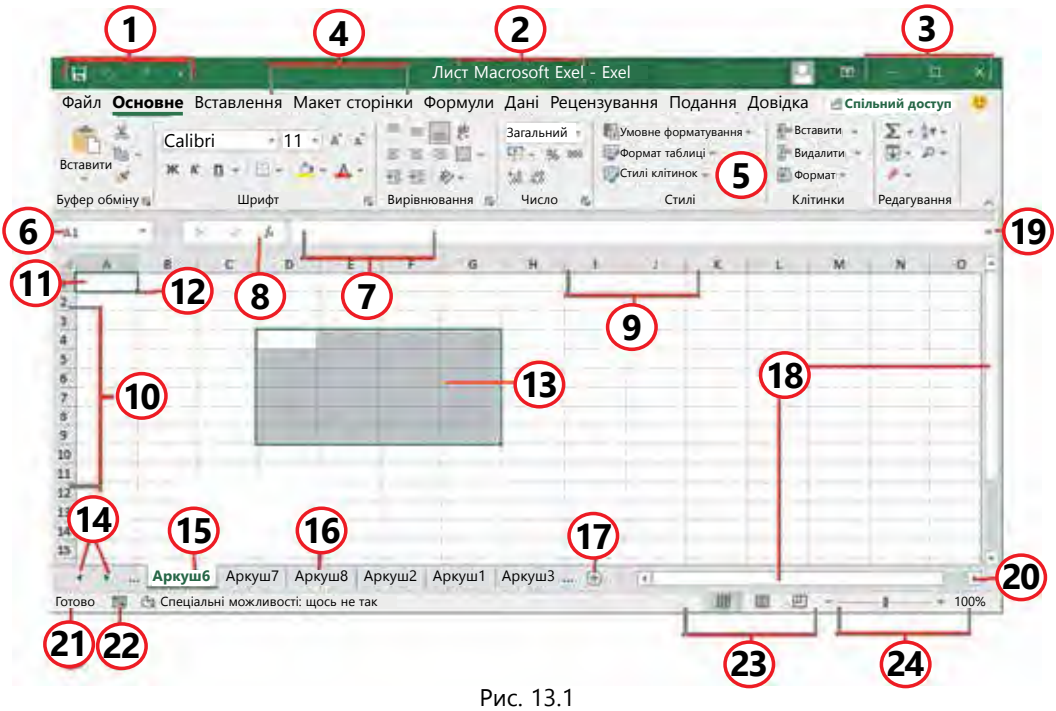


Рис. 13.1

7. Рядок формул.
8. Кнопка виклику *Майстра функцій*.
9. Заголовки стовпців.
10. Кнопки імен рядків — позначають порядкові номери рядків.
11. Поточна клітинка.
12. Маркер заповнення.
13. Виділений діапазон D4:G9.
14. Панель із кнопками для прокручування аркушів робочої книги.
15. Назва активного аркуша.
16. Назва неактивного аркуша.
17. Кнопка для додавання до книги нового аркуша.
18. Вертикальна та горизонтальна смуги прокручування.
19. Кнопка *Розгорнути рядок формул*.
20. Кнопка покрокового прокручування.
21. Режими роботи — *Готово, Ввід, Редагування*.
22. Кнопка для вмикання (вимикання) режиму запису макросу.
23. Панель із кнопками для змінювання вигляду таблиці на екрані.
24. Панель для зміни масштабу зображення.

Інформація в електронних таблицях, як і у звичайних, зберігається у *клітинках*, які утворюють *рядки* й *стовпці*. Рядки таблиці в Excel позначено цифрами натурального ряду 1, 2, ... , 1048576, а стовпці — літерами латинського алфавіту A, B, C, ... , Z, AA, AB, ... , AZ, BA, ... , BZ, ... XFD. Ці цифри і літери, розміщені в окремих клітинках, називають *заголовками* рядків і стовпців.

Отже, кожна клітинка електронної таблиці має свою адресу, яка складається з позначення стовпця і рядка, на перетині яких розташована ця клітинка, наприклад: адреса BA45 означає, що клітинка міститься на перетині стовпця BA і рядка 45.

Електронна таблиця в Excel — це *робочий аркуш,* що має ім'я, яке розміщено на ярличку внизу аркуша. Сукупність робочих аркушів становить *робочу книгу.* Кожна робоча книга зберігається як окремий файл зі своїм унікальним іменем. Стандартний робочий аркуш має 1 048 576 рядків і 16 384 стовпці, тому у вікні на екрані монітора видно лише його невелику частину. Щоб вивести на екран потрібну частину робочого аркуша, використовують горизонтальну і вертикальну смуги прокручування. Переміщуючи за допомогою миші повзунки на лінійках прокручування виводять на екран і роблять видимими потрібну частину робочого аркуша.

Виділення об'єктів. Потрібну клітинку виділяють, клацнувши на ній лівою кнопкою миші. *Активну* клітинку буде окреслено темною рамкою з *маркером заповнення* — квадратиком у нижньому правому кутку. Адреса активної клітинки виводиться у лівому полі рядка формул. Рядок і стовпець, у яких розташована активна клітинка, також є *активними*.

Виділити кілька суміжних клітинок, тобто їх *діапазон*, можна курсором, провівши по цих клітинках із натиснутою лівою кнопкою миші, або за допомогою комбінації клавіш *Shift* +↑, *Shift* +→. Щоб виділити стовпець або рядок, треба клацнути лівою кнопкою миші на його імені; весь робочий аркуш — клацнути мишею на кнопці *Виділити все* у верхньому правому куті робочого аркуша.

Зміна розмірів рядків і стовпців. Щоб змінити ширину стовпця, потрібно підвести курсор до його правої межі; курсор змінить свій вигляд. Тримаючи натиснутою ліву кнопку миші, перемістити межі стовпця на потрібну відстань. Таку саму операцію виконують із рядком таблиці.

Типи даних. За допомогою електронних таблиць обробляють різноманітну інформацію, як-от числа, тексти, формули.

В Excel доступні, зокрема, такі числові формати:

- загальний;
- числовий;
- фінансовий;
- грошовий;
- відсотковий;
- дробовий.

Загальний формат використовують як для чисел, так і для текстової інформації (рис. 13.2). Числа у форматі *Загальний* відображаються так, як їх вводять.

Для чисел у форматі *Числовий* зазначають кількість розрядів після десяткової коми. У форматах *Фінансовий* і *Грошовий* числа подають разом із символом національної грошової одиниці. У форматі *Дробовий* числа відображаються у вигляді простих, десяткових дробів тощо.

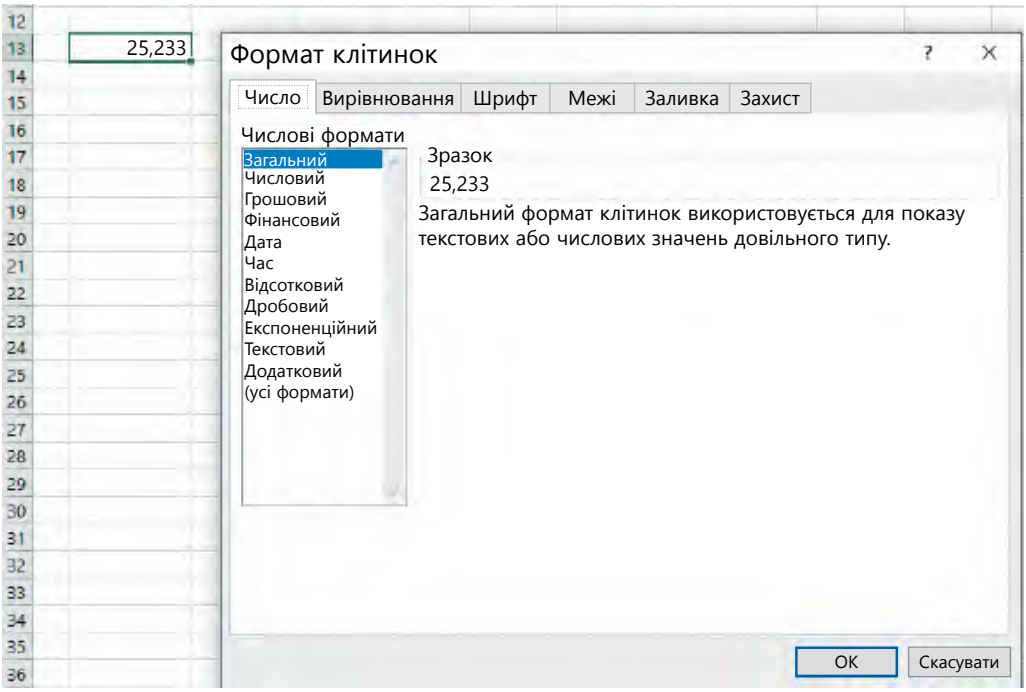


Рис. 13.2

Текстову інформацію, що складається з різноманітних символів — літер, цифр, розділових і спеціальних знаків тощо, вводять у форматі *Загальний*.

Формула — це математичний вираз, що починається зі знака = і містить знаки арифметичних операцій +, -,*,/ та назви функцій з аргументами. Формулу вводять у полі рядка формул.

Уведення інформації. Щоб ввести інформацію, потрібно виділити клітинку таблиці, тобто зробити її активною. Після введення першого символу форма курсора зміниться на тонку вертикальну риску, яка вказує на позицію, куди вводитиметься наступний символ. Щоб вилучити символ у позиції курсора, треба натиснути клавішу *Del*, а символ ліворуч від курсора — клавішу *Backspace*. Одразу після початку введення в рядку формул з'являються дві кнопки: ╳ — скасування і ✓ — введення. Щоб завершити введення, потрібно натиснути клавішу *Enter* або кнопку ✓, щоб скасувати введення — клавішу *Esc* або кнопку ╳. Після завершення введення інформації у клітинці автоматично активізується клітинка нижче. Напрямок зсуву активізації клітинок можна вибрати на вкладці *Файл* — команда *Параметри* — *Додатково* (рис. 13.3).

Щоб заповнити клітинки однаковою інформацією, ці дані вводять у першу клітинку діапазону. Курсор підводять до маркера заповнення — маленького темного квадратика у правому нижньому кутку виділеної клітинки. Курсор змінюється на маленький темний хрестик. Натиснувши і тримаючи ліву кнопку миші, маркер заповнення пересувають у потрібному напрямку й на потрібну кількість клітинок (рис. 13.4).

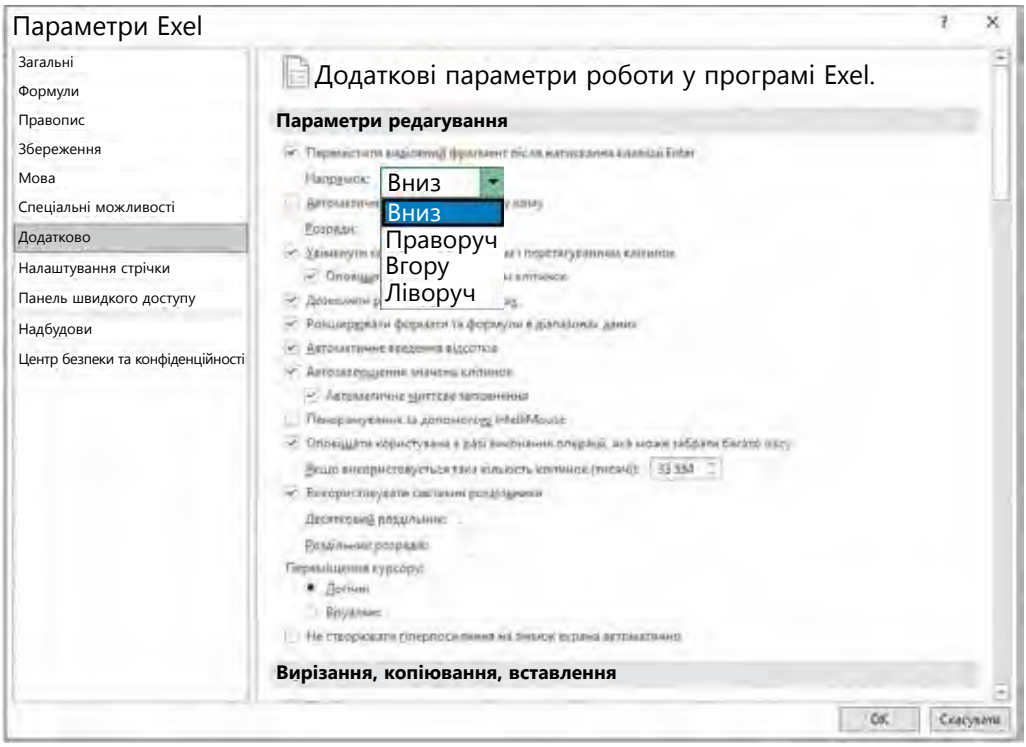


Рис. 13.3

Якщо потрібно заповнити діапазон клітинок членами арифметичної прогресії, вводять у дві сусідні клітини два перші члени цієї прогресії. Потім, виділивши мишею ці дві клітинки (рис. 13.5, *а*), захоплюють мишею маркер заповнення і переміщують рамку виділення на потрібний діапазон у потрібному напрямку (рис. 13.5, *б*).

	A	B	C	D	E
1	№ з/п	Назва товару	Ціна	Кількість	
2	1	Мікрофон	92,00 грн	15	
3	2	Мікрофон	92,00 грн	15	
4	3	Мікрофон	92,00 грн	15	
5	4	Мікрофон	92,00 грн	15	
6	5	Мікрофон	92,00 грн	15	
7	6	Мікрофон	92,00 грн	15	
8	7	Мікрофон	92,00 грн	15	
9					
10					

Рис. 13.4

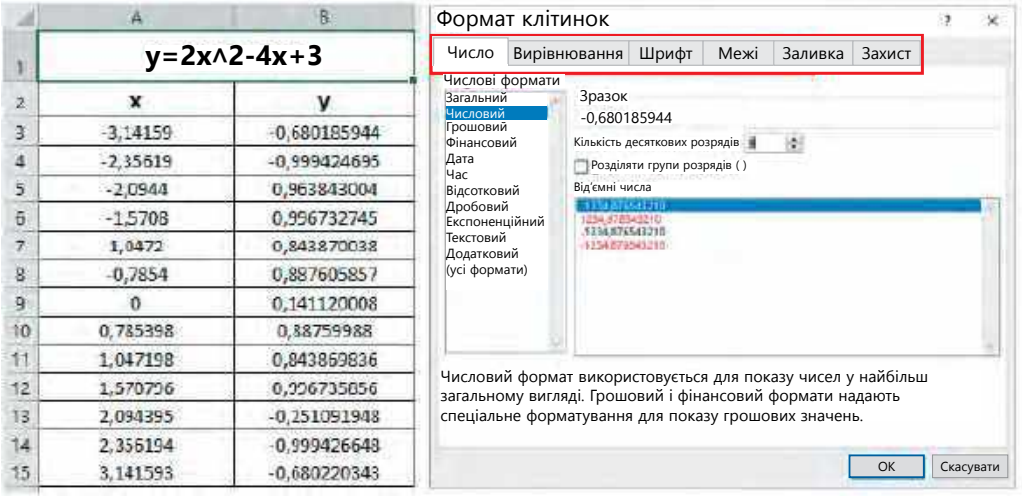
Форматування інформації в клітинках електронної таблиці здійснюють за допомогою команд на вкладці *Основне* у групах: *Шрифт, Вирівнювання, Стилі, Клітинки*. Команду *Формат клітинок* можна вибрати з контекстного меню, клацнувши правою кнопкою миші на активній (виділеній) клітинці або викликати комбінацією клавіш *Ctrl*+1. Діалогове вікно *Формат клітинок* цієї команди містить вкладки *Число, Вирівнювання, Шрифт, Межі, Заливка* і *Захист* (рис. 13.6).

На вкладці *Число* з випадного списку *Числові формати* можна вибрати потрібний формат. Зауважимо, що формат *Загальний* придатний для форматування як числової, так і текстової інформації.

а *б*

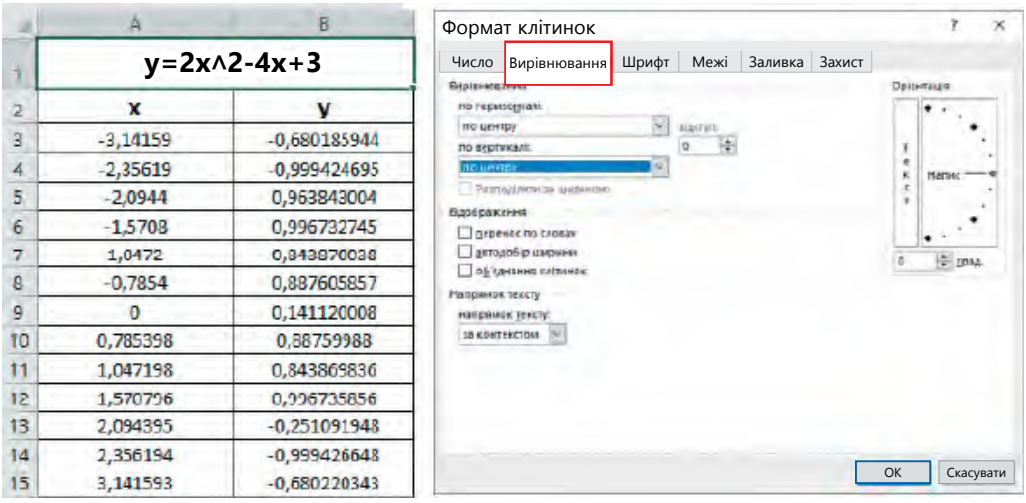
Рис. 13.5

На вкладці *Вирівнювання* в полях *По горизонталі* і *По вертикалі* задають вид вирівнювання і значення відступу. За допомогою транспаранта *Орієнтація* і регулятора *Градуси* — орієнтацію написів під довільним кутом до горизонталі (рис. 13.7).



	A	B
1	y=2x^2-4x+3	
2	x	y
3	-3,14159	-0,680185944
4	-2,35619	-0,999424695
5	-2,0944	0,963843004
6	-1,5708	0,996732745
7	1,0472	0,843870038
8	-0,7854	0,887605857
9	0	0,141120008
10	0,785398	0,88759988
11	1,047198	0,843869836
12	1,570796	0,996735856
13	2,094395	-0,251091948
14	2,356194	-0,999426648
15	3,141593	-0,680220343

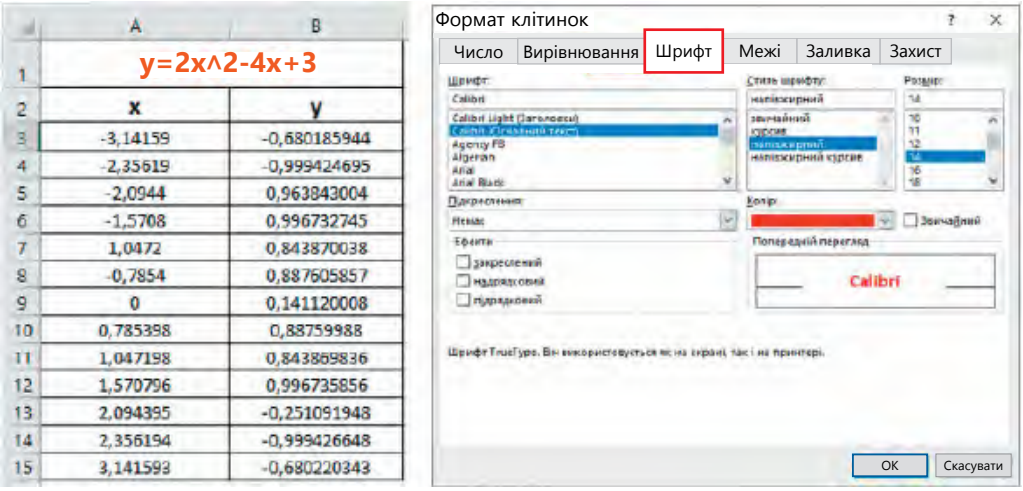
Рис. 13.6



	A	B
1	y=2x^2-4x+3	
2	x	y
3	-3,14159	-0,680185944
4	-2,35619	-0,999424695
5	-2,0944	0,963843004
6	-1,5708	0,996732745
7	1,0472	0,843870038
8	-0,7854	0,887605857
9	0	0,141120008
10	0,785398	0,88759988
11	1,047198	0,843869836
12	1,570796	0,996735856
13	2,094395	-0,251091948
14	2,356194	-0,999426648
15	3,141593	-0,680220343

Рис. 13.7

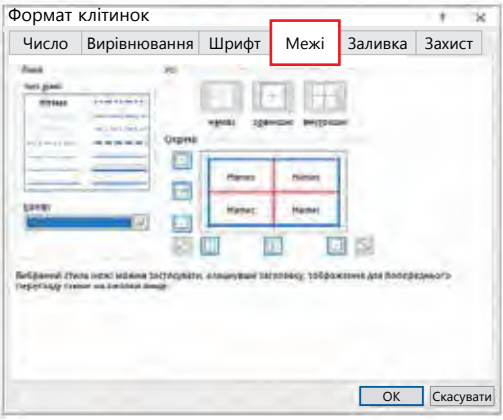
Параметри шрифту — гарнітуру, стиль, розмір, колір тощо — можна вибрати на вкладці *Шрифт* (рис. 13.8).



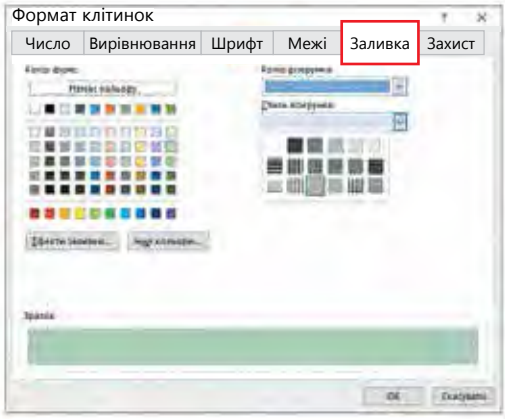
	A	B
1	y=2x^2-4x+3	
2	x	y
3	-3,14159	-0,680185944
4	-2,35619	-0,999424695
5	-2,0944	0,963843004
6	-1,5708	0,996732745
7	1,0472	0,843870038
8	-0,7854	0,887605857
9	0	0,141120008
10	0,785398	0,88759988
11	1,047198	0,843869836
12	1,570796	0,996735856
13	2,094395	-0,251091948
14	2,356194	-0,999426648
15	3,141593	-0,680220343

Рис. 13.8

До тексту або числових даних є змога додати рамки на вкладці *Межі* (рис. 13.9, *а*) і кольорове тло з візерунком, які можна вибрати на вкладці *Заливка* (рис. 13.9, *б*).



а



б

Рис. 13.9

Щоб убезпечити дані від несанкціонованих змін, треба установити прапорець *Захистити клітинку* на вкладці *Захист* (рис. 13.10).

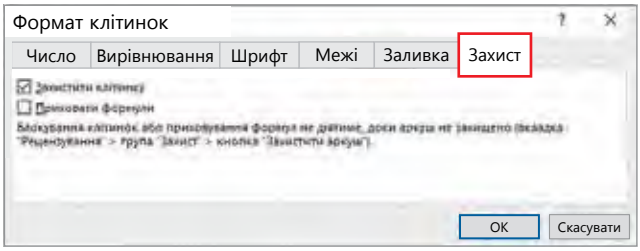


Рис. 13.10

13.2. ОБРОБЛЕННЯ ІНФОРМАЦІЇ В Excel

Для оброблення інформації в Excel є потужний механізм формул. *Формула* — це сукупність операндів, сполучених між собою знаками математичних операцій і дужками. Формальною ознакою формули є те, що першим символом, з якого вона починається, має бути символ =. Операндами формули можуть бути: числа, зокрема дата і час; текст; адреси клітинок таблиці, тобто посилання на клітинки; функції.

В електронних таблицях Excel є чотири види операторів: арифметичні, текстові (оператори об'єднання), оператори порівняння й оператори посилань (адресні).

Арифметичні оператори в Excel призначені для виконання арифметичних операцій із числами (табл. 13.1).

Таблиця 13.1

Арифметичні оператори

Символ оператора	Назва оператора	Приклад формули	Результат
+	Додавання	=1,6+3,2	4,8
–	Віднімання	=7–6,2	0,8
–	Заперечення	=–24	–24
/	Ділення	=3/5	0,6
*	Множення	=5*7	35
^	Піднесення до степеня	=2^2	4
%	Відсоток	=15%	0,15

Вводячи формули, потрібно враховувати порядок виконання арифметичних операторів. Якщо необхідно змінити звичайний порядок, застосовують круглі дужки. Наприклад, треба розв'язати рівняння $z=(x+y)^2–0,25$, якщо x=10, y=2. Для цього активізують клітинку A1 і вводять значення x, аналогічно до клітинки B1 — y, C1 — z, A2 — 10, B2 — 2, до клітинки C2 записують формулу =(A2+B2)^2-0,25 і натискають на клавішу *Enter*. У клітинці C2 з'являється розрахункове значення 143,75, а в рядку формул (рис. 13.11) — формула.

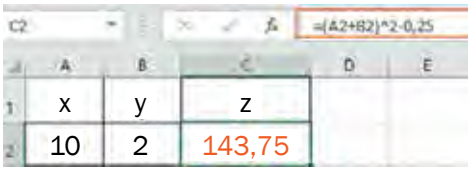


Рис. 13.11

Текстовий оператор «&» (англ. ampersand) використовують для об'єднання одного або кількох текстових рядків в один фрагмент тексту, тому його називають також *оператором об'єднання*.

Наприклад, якщо до клітинок A1 і A2 ввести слова *«Інформаційні тех»* і *«нології»* відповідно, а до клітинки A3 — формулу =A1&A2, то результатом у клітинці A3 буде словосполучення *«Інформаційні технології»* (рис. 13.12).

За допомогою **операторів порівняння** порівнюють значення. Якщо твердження правильне, то у клітинці, що містить формулу, буде значення TRUE; якщо твердження неправильне — значення FALSE (табл. 13.2).

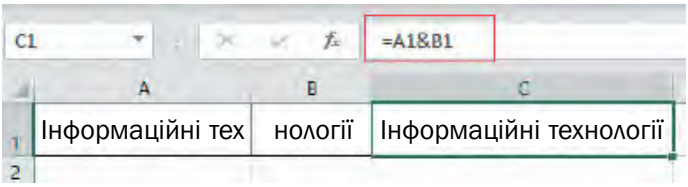


Рис. 13.12

Таблиця 13.2

Оператори порівняння

Символ оператора	Назва оператора	Приклад формули	Результат
=	Дорівнює	=4=4	TRUE
>	Більше	=4>5	FALSE
<	Менше	=3<4	TRUE
>=	Більше або дорівнює	=4>=5	FALSE
<=	Менше або дорівнює	=4<=5	TRUE
<>	Не дорівнює	=4<>5	TRUE

Оператори посилань (адресні) створюють посилання на клітинки таблиці. Прямокутні діапазони клітинок позначають за допомогою двокрапки, наприклад A2:F8. Якщо в посиланні потрібно об'єднати два діапазони клітинок, то діапазони записують через крапку з комою, наприклад A2:F8;B6:K14. Отже, вираз =СУММ(A2:F8;B6:K14) означає додавання клітинок прямокутних діапазонів A2:F8 і B6:K14.

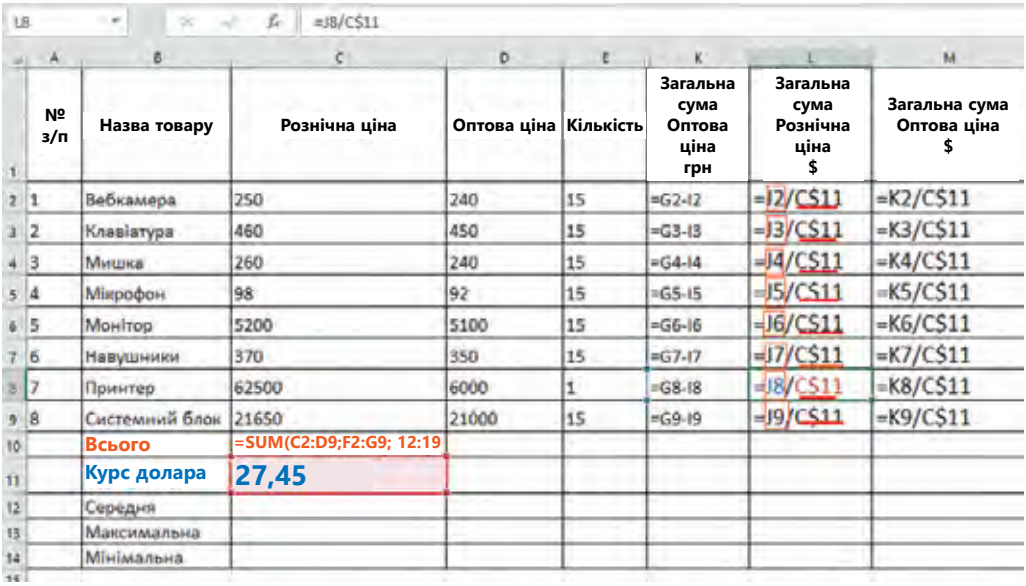
Посилання на клітинку — це адреса клітинки таблиці, що містить заголовок колонки (стовпця) та номер рядка і використовується у формулі як аргумент функції або операнд. Розрізняють відносні, абсолютні й змішані посилання.

Відносні посилання — це посилання на розташування клітинки. Зі зміною адреси клітинки (наприклад, під час додавання або вилучення рядка чи стовпця) автоматично змінюється і посилання у формулі, яка його використовує.

Щоб змінити посилання на *абсолютне*, додають знак $ перед заголовком рядка або колонки в адресі, наприклад C11. Якщо знаку $ в адресі клітинки немає — це відносне посилання. Абсолютні посилання за зміни адреси клітинки не змінюються.

У *змішаних* посиланнях абсолютною є адреса колонки, а відносною — адреса рядка $C11, або навпаки (наприклад, C$11 — це змішане (відносно-абсолютне) посилання, а I2, I3 ... I9 є відносними посиланнями) (рис. 13.13).

У формулі можуть бути посилання на діапазон клітинок, наприклад посилання AB345:AB355 вказує на десять сусідніх клітинок стовпця AB з 345 до 355 рядка. Щоб адресувати прямокутний діапазон клітинок, у посиланнях зазначають адресу лівої верхньої і правої нижньої клітинки діапазону, приміром



	A	B	C	D	E	K	L	M
1	№ з/п	Назва товару	Рознічна ціна	Оптова ціна	Кількість	Загальна сума Оптова ціна грн	Загальна сума Рознічна ціна $	Загальна сума Оптова ціна $
2	1	Вебкамера	250	240	15	=G2-I2	=I2/C$11	=K2/C$11
3	2	Клавіатура	460	450	15	=G3-I3	=I3/C$11	=K3/C$11
4	3	Мишка	260	240	15	=G4-I4	=I4/C$11	=K4/C$11
5	4	Мікрофон	98	92	15	=G5-I5	=I5/C$11	=K5/C$11
6	5	Монітор	5200	5100	15	=G6-I6	=I6/C$11	=K6/C$11
7	6	Навушники	370	350	15	=G7-I7	=I7/C$11	=K7/C$11
8	7	Принтер	62500	6000	1	=G8-I8	=I8/C$11	=K8/C$11
9	8	Системний блок	21650	21000	15	=G9-I9	=I9/C$11	=K9/C$11
10		Всього	=SUM(C2:D9;F2:G9; 12:19					
11		Курс долара	27,45					
12		Середня						
13		Максимальна						
14		Мінімальна						

Рис. 13.13

посилання AB45:AE51 адресує прямокутний діапазон клітинок завширшки чотири клітинки і заввишки шість клітинок.

За допомогою кнопки Σ *(Автосума)* автоматично розраховується сума значень у рядку або стовпці. Для цього потрібно активізувати вільну клітинку, встановити курсор під стовпцем (або праворуч від рядка) й натиснути на кнопку *Автосума* на вкладці *Основне*. В клітинці автоматично створюється функція *SUM* із зазначеним аргументом і відповідним діапазоном стовпця або рядка. Далі натиснути на кнопку, розрахунок суми в стовпці матиме такий вигляд: *=SUM(D2:D9)* (рис. 13.14).

Інколи під час обчислень у формулах виникають помилки. В табл. 13.3 подано перелік помилок та причини їх виникнення.

Таблиця 13.3

Види помилок

Помилка	Причина виникнення
#######	Результат обчислення формули не вміщується до клітинки
#DIV/0! (#ДІЛЕННЯ/0)	Спроба ділення на нуль
#N/A (#Н/Д)	Значення недоступне
#NAME? (#ІМ'Я?)	Неможливо розпізнати ім'я у формулі
#NUM! (#ЧИСЛО!)	Неправильні числові значення
#REF! (#ПОСИЛАННЯ)	Неприпустиме посилання на клітинку
#VALUE! (#ЗНАЧЕННЯ!)	Неприпустимий тип аргументу
#NULL	Використовується помилкове посилання на клітинку або діапазон

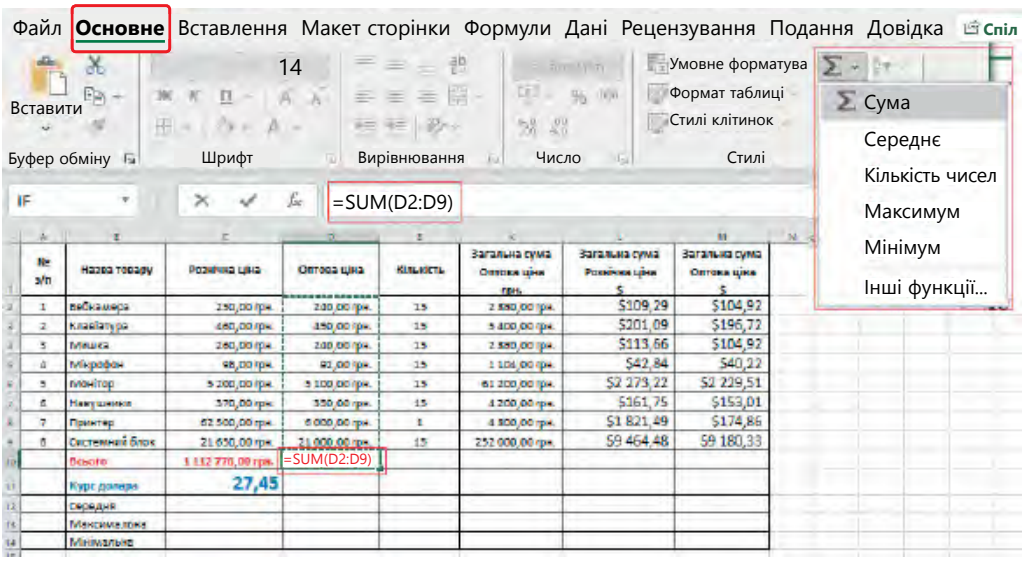


Рис. 13.14

Функції Excel призначені для виконання складних операцій оброблення інформації. Кожна функція має унікальне *ім'я*, після якого у круглих дужках наведено аргументи (або аргумент), із якими здійснюється відповідна операція. Пробіл між іменем функції і круглими дужками не ставлять. Значення, які повертаються функціями як відповіді, називають *результатами*.

В Excel є понад 400 вбудованих функцій, поділених на категорії, відповідно до галузей знань. Аргументами функцій можуть бути числові й текстові константи, посилання на клітинки і діапазони клітинок.

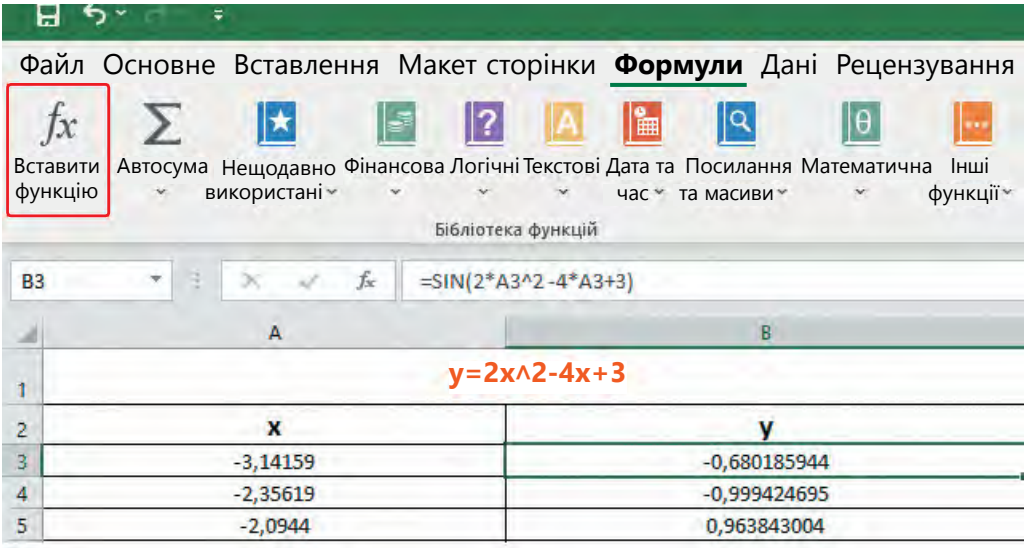


Рис. 13.15

Ввести функцію у формулу можна за допомогою вкладки **Формули** (рис. 13.15): вибрати опцію *Вставити функцію,* у вікні *Вставлення функції* (рис. 13.16) вибрати спочатку *Категорія* функції (наприклад, *Математичні*), а потім потрібну функцію (приміром, *SIN*). Або можна виділити клітинку й натиснути комбінацію клавіш *SHIFT+F3* — відкриється вікно *Майстра функцій.*

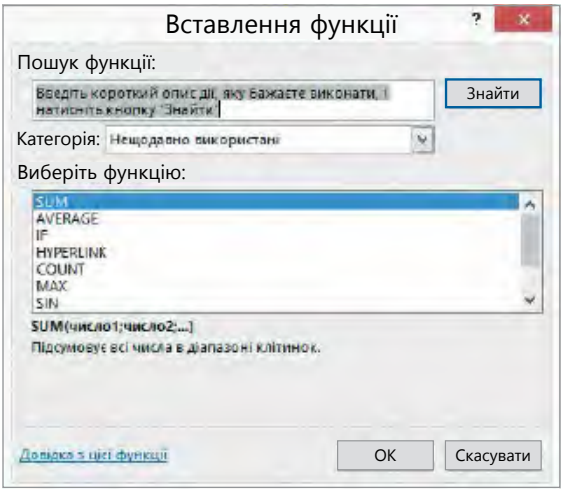


Рис. 13.16

Існують різні аргументи функцій: текст, число, формули, логічне значення тощо. Бувають функції без аргументів (наприклад, *TODAY(), PI*). Розглянемо деякі категорії функцій Excel.

Статистичні функції призначені для статистичного аналізу даних: *Сума, Середнє, Кількість чисел, Мінімум, Максимум, Інші функції* — група *Редагування* на вкладці **Основне**.

Функцію *COUNTIF* використовують для визначення кількості значень, що відповідають певній умові. Наприклад, у стовпці B наведено інформацію про назви товарів. Необхідно визначити кількість товарів із назвою *Вебкамери*. Для цього викликають функцію *COUNTIF* (рис. 13.17) і задають відповідні аргументи (рис. 13.18).

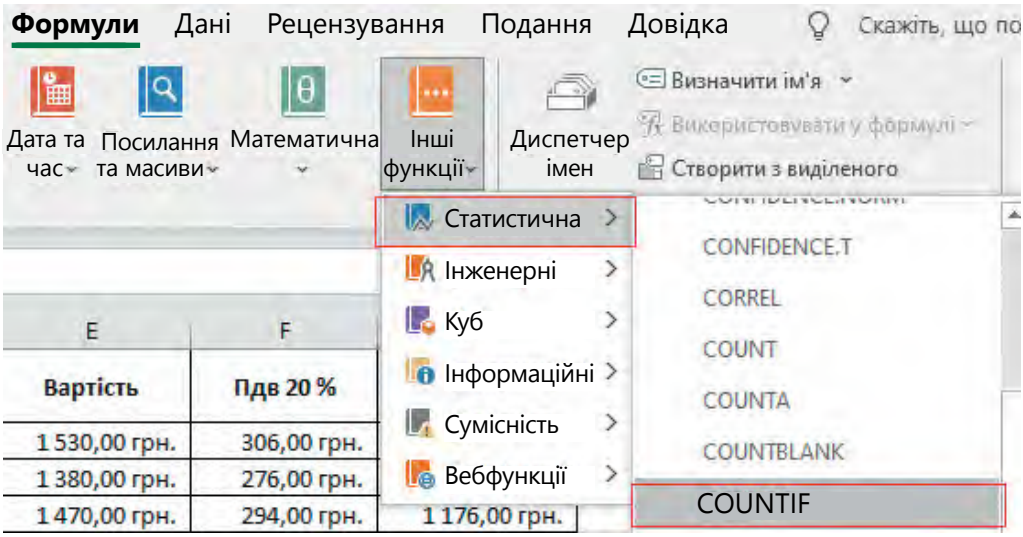
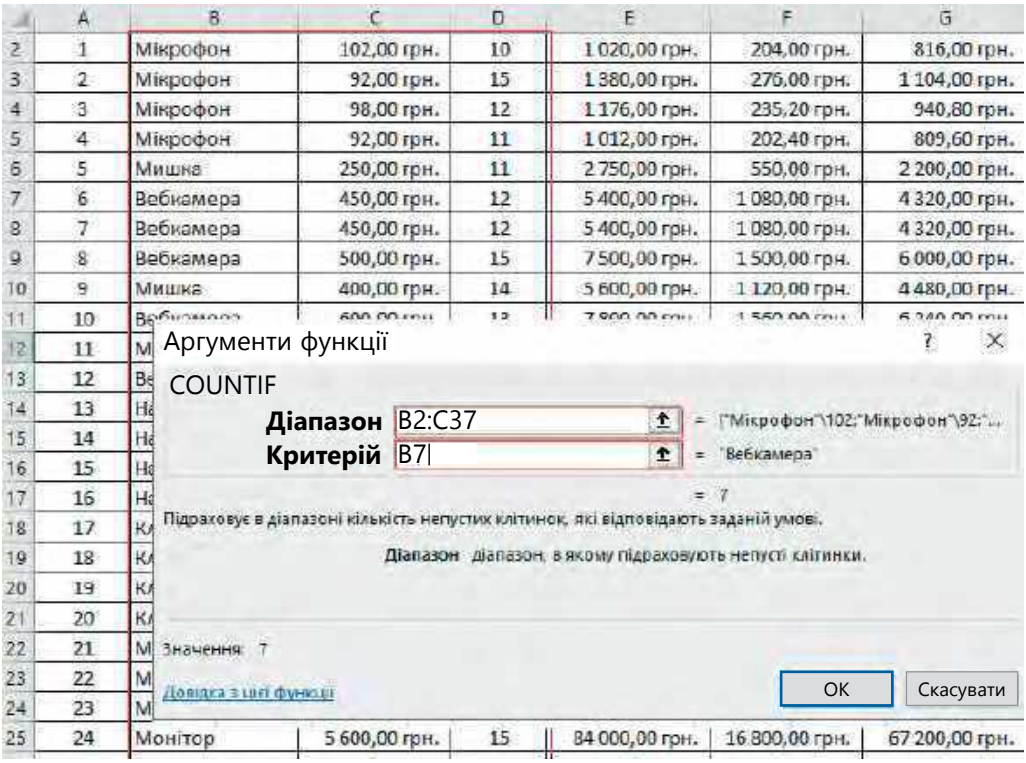


Рис. 13.17



	A	B	C	D	E	F	G
2	1	Мікрофон	102,00 грн.	10	1 020,00 грн.	204,00 грн.	816,00 грн.
3	2	Мікрофон	92,00 грн.	15	1 380,00 грн.	276,00 грн.	1 104,00 грн.
4	3	Мікрофон	98,00 грн.	12	1 176,00 грн.	235,20 грн.	940,80 грн.
5	4	Мікрофон	92,00 грн.	11	1 012,00 грн.	202,40 грн.	809,60 грн.
6	5	Мишка	250,00 грн.	11	2 750,00 грн.	550,00 грн.	2 200,00 грн.
7	6	Вебкамера	450,00 грн.	12	5 400,00 грн.	1 080,00 грн.	4 320,00 грн.
8	7	Вебкамера	450,00 грн.	12	5 400,00 грн.	1 080,00 грн.	4 320,00 грн.
9	8	Вебкамера	500,00 грн.	15	7 500,00 грн.	1 500,00 грн.	6 000,00 грн.
10	9	Мишка	400,00 грн.	14	5 600,00 грн.	1 120,00 грн.	4 480,00 грн.
11	10	Ве	600,00 грн.	13	7 800,00 грн.	1 560,00 грн.	6 240,00 грн.
12	11	М					
13	12	Ве					
14	13	На					
15	14	На					
16	15	На					
17	16	На					
18	17	Кл					
19	18	Кл					
20	19	Кл					
21	20	Кл					
22	21	М					
23	22	М					
24	23	М					
25	24	Монітор	5 600,00 грн.	15	84 000,00 грн.	16 800,00 грн.	67 200,00 грн.

Рис. 13.18

Математичні функції спрощують тригонометричні та арифметичні обчислення:

- визначають синус *SIN*, косинус *COS*, тангенс *TAN* кута тощо;
- працюють із матрицями, аргументами яких є масиви (матриці) і які повертають як результат матрицю (наприклад, множення матриць — *MMULT*). Для розміщення результату потрібно виділити діапазон перед введенням функції. Після завершення введення аргументів математичних функцій *не натискають кнопку OK*, а обов'язково застосовують комбінацію клавіш *CTRL+SHIFT+ENTER;*
- функція *ROMAN* перетворює число, записане арабськими цифрами, на число римськими цифрами як текст;
- функція *PI* повертає число пі;
- функція *ABS* повертає модуль числа;
- функція *SUMIF* підсумовує значення клітинок, яке відповідає заданому критерію;
- функція *SQRT* повертає додатне значення квадратного кореня.

Логічні функції призначені для перевірки виконання однієї або кількох умов:

- функція *IF* — для розв'язання задач, у яких потрібно перевірити, чи виконується умова;

- функція *FALSE* — повертає логічне значення FALSE (хибність);
- функція *TRUE* — повертає логічне значення TRUE (істина);
- функція *AND* — повертає логічне значення TRUE (істина), якщо всі аргументи мають значення TRUE (істина);
- функція *OR* — повертає логічне значення TRUE (істина), якщо хоча б один аргумент має значення TRUE;
- функція *NOT* — змінює логічне значення аргументу на протилежне: *TRUE* на *FALSE, FALSE* на *TRUE*.

Функції дати й часу визначають дату або час.

Фінансові функції призначені для обчислень, пов'язаних із фінансовим аналізом, виплатами за кредитом, ефективністю капіталовкладень, амортизацією устаткування, розрахунком відсотків тощо.

Функції посилань та масивів застосовують для розрахунків із таблицями даних.

За допомогою **текстових функцій** обробляють текстові дані.

Функції баз даних — це функції роботи зі списками та зовнішніми базами даних.

Приклад 1. *Множення матриці на число 3.* Потрібно створити матрицю А в діапазоні клітинок A2:C4, виділити поле вільних клітинок розміром, що відповідає заданій матриці. Далі в рядку формул ввести формулу: = A2:C4*3 і натиснути на клавіші *Ctrl+Shift+Enter* (рис. 13.19).

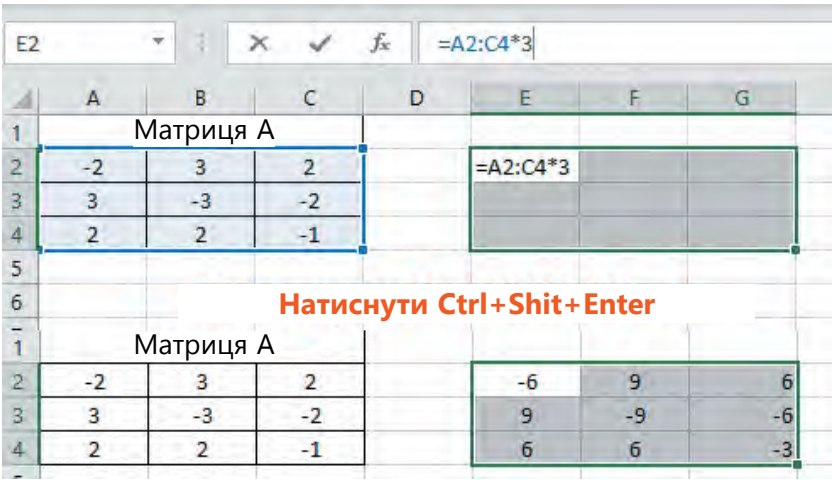


Рис. 13.19

Приклад 2. *Множення матриці на «вектор С».* Ввести дані таблиці *Матриця* та *Вектор* відповідно до зразка (рис. 13.20). Виділити діапазон для розміщення результату обчислення (наприклад, D6:F8) і викликати функцію *MMULT*, де Масив1 — діапазон A2:C4 (*Матриця*), Масив2 — діапазон G2:G4 (*Вектор*); це масиви, які перемножуються. Натиснути на комбінацію клавіш *Ctrl+Shift+Enter* (рис. 13.21).

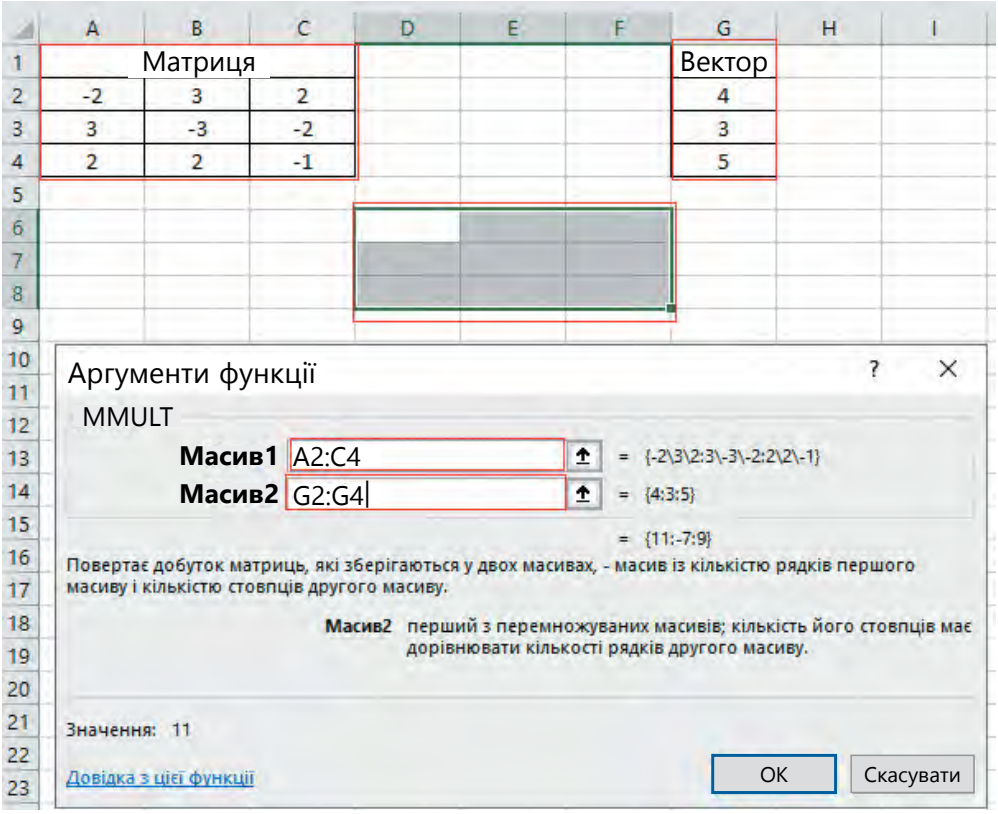


Рис. 13.20

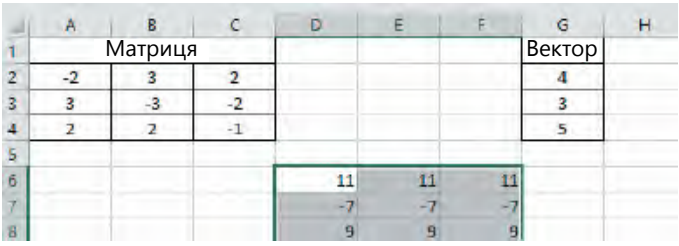


Рис. 13.21

13.3. ПОДАННЯ ІНФОРМАЦІЇ ЗА ДОПОМОГОЮ ДІАГРАМ І ГРАФІКІВ

Діаграми та графіки унаочнюють інформацію, роблять її зручнішою для сприйняття. За допомогою цих графічних зображень можна ілюструвати функціональну залежність однієї величини від іншої, виявляти тенденції зміни певного параметра в часі, порівнювати характеристики різних об'єктів, встановлювати взаємозв'язки різних величин і вплив певних чинників на перебіг досліджуваних процесів.

Для відображення таких складних інформаційних взаємозв'язків в Excel є понад 15 різноманітних типів діаграм, кожна з яких має кілька різновидів (рис. 13.22).

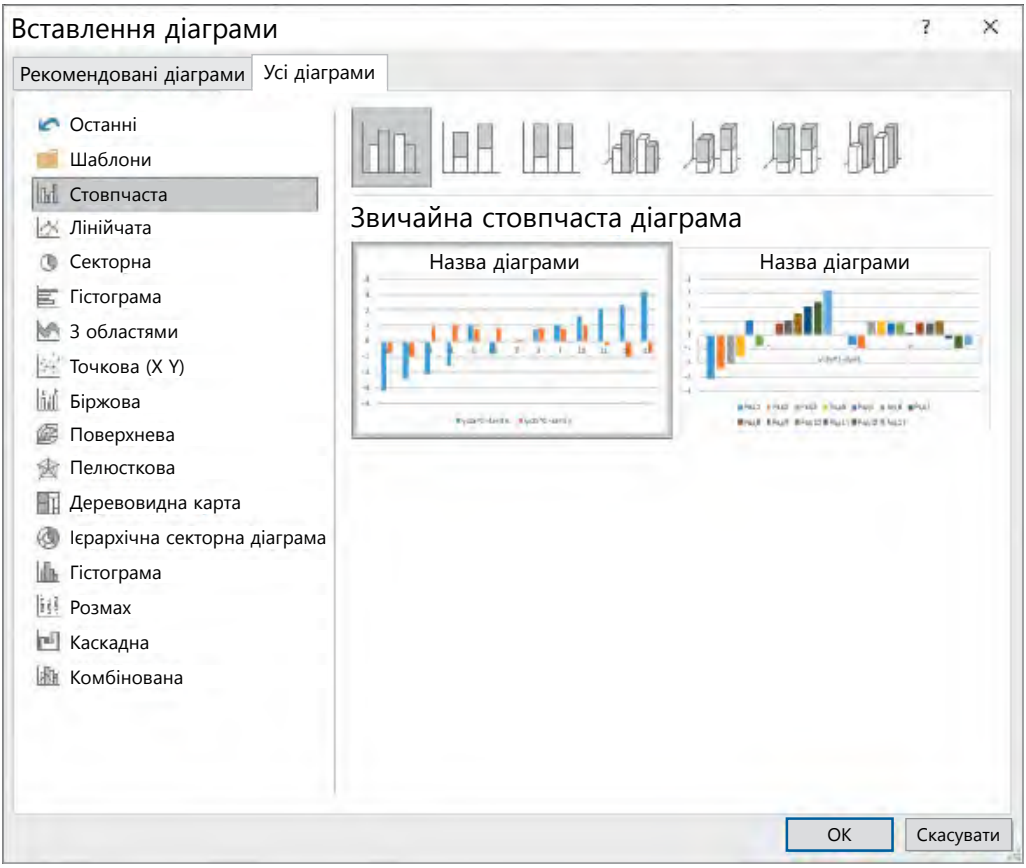


Рис. 13.22

Будь-який тип діаграми відображає залежність однієї чи кількох величин від іншої. Незалежні змінні (аргументи) — це *категорії*, а залежні — *значення*. Відповідно, горизонтальна вісь (вісь *x*) має назву *вісь категорій,* а вертикальна (вісь *y*) — *вісь значень*.

В Excel передбачено два варіанти відображення: за рядками і за стовпцями. У першому випадку заголовки рядків відображаються на осі категорій (осі *x*), а вміст клітинок певного стовпця — на осі значень (осі *y*), у другому — на осі категорій відображаються заголовки стовпців, а на осі значень — вміст клітинок певного рядка.

Для відображення залежностей в Excel можна будувати діаграми таких стандартних типів:

- стовпчаста — цифрові дані відображено у вигляді прямокутників або стовпчиків;

- лінійчаста — дані у вигляді окремих точок, які об'єднуються лініями різних типів;
- секторна — дані у вигляді секторів кола;
- з областями — як графік, але області розташовуються під лініями і виділяються різними кольорами;
- точкова (*x, y*) — окремі точки з позначенням координат *x, y*;
- біржова — відображено мінімальні й максимальні ціни, а також ціни на момент закриття торгів;
- поверхнева — як графік, але дані подано у вигляді тривимірної поверхні;
- пелюсткова — дані відображено щодо центральної точки, а не щодо осей *x, y*;
- деревовидна карта — ієрархічне подання даних, є змога легко порівняти різні рівні категоризації;
- ієрархічна секторна — відображено ієрархічні дані, наприклад діаграма *Сонячне проміння* — нагадує секторну, однак у центрі має отвір;
- гістограма — аналогічна стовпчастій, але зображення розташовано горизонтально;
- розмах — подано розподіл даних за квартилями й виділено середні та сторонні значення;
- каскадна — проміжний підсумок фінансових даних у процесі додавання або віднімання значень;
- комбінована — поєднано різні типи діаграм.

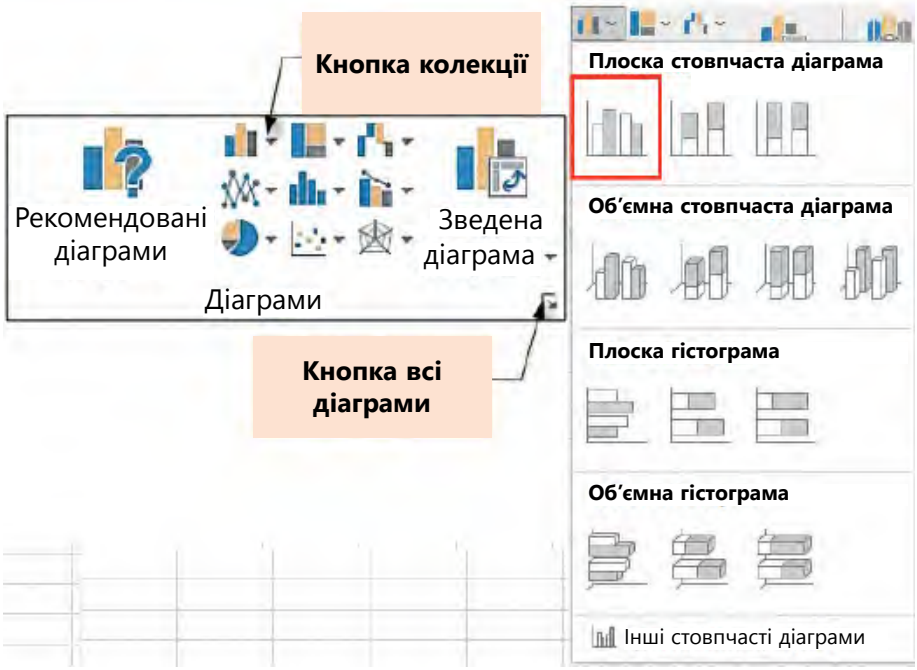


Рис. 13.23

Щоб почати побудову діаграми, треба вибрати тип діаграми й натиснути трикутник праворуч від мініатюри в групі *Діаграми* на вкладці **Вставлення**. Відкриється колекція діаграм. На рис. 13.23 праворуч наведено колекцію для вибору стовпчастої діаграми.

Відтак діаграму, побудовану відповідно до виділеного діапазону даних у таблиці, буде автоматично вставлено на робочий аркуш (рис. 13.24).

x	y
-3,14	-0,68
-2,36	-1,00
-2,09	0,96
-1,57	1,00
1,05	0,84
-0,79	0,89
0,00	0,14
0,79	0,89
1,05	0,84
1,57	1,00
2,09	-0,25
2,36	-1,00
3,14	-0,68

Рис. 13.24

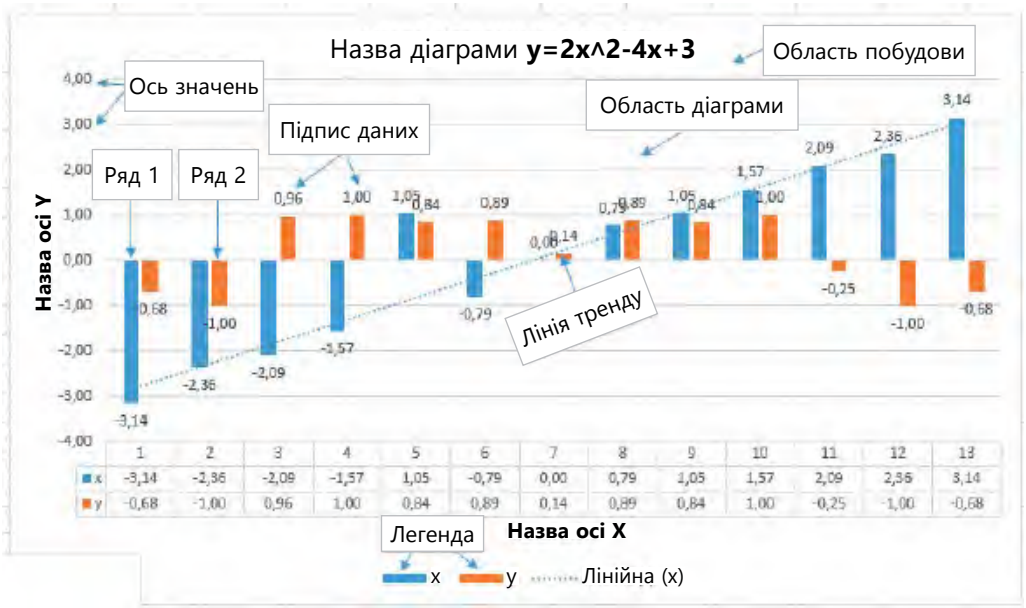


Рис. 13.25

Основні **елементи діаграми** (рис. 13.25):

1. Область побудови, в якій розміщуються всі елементи діаграми.
2. Область діаграми, обмежена осями, яка містить ряди даних.

3. Ряди даних — набір пов'язаних між собою числових даних, що відображаються на вертикальній осі діаграми у вигляді стовпців, секторів тощо. У вихідній таблиці зазвичай розташовані по рядках.

4. Категорії — зазвичай назви даних, що відображаються під горизонтальною віссю діаграми. У вихідній таблиці зазвичай розташовані по стовпцях.

5. Осі — лінії, що обмежують область діаграми і мають поділки зі значеннями вибраних одиниць виміру. Горизонтальна вісь (*x*) відповідає категоріям. Вертикальна вісь (*y*) відповідає значенням даних у категоріях. У тривимірних діаграм є третя вісь (*z*), яка, зазвичай, відповідає часу.

6. Легенда — графічні зображення та підписи, які відповідають категоріям і полегшують читання діаграми.

7. Назви — текст, який відображає назву діаграми або осей.

8. Підписи даних — значення рядів даних у певних категоріях, що полегшують читання діаграми.

9. Лінія тренду — графік певної функції (лінійної, логарифмічної, статистичної тощо), отриманий у результаті оброблення даних ряду методом найменших квадратів; дає змогу наочно уявити тенденцію зміни даних.

Форматування діаграми проводять за допомогою команд, розташованих на контекстних вкладках *Конструктор* і *Формат*, які автоматично стають доступними відразу після виділення діаграми.

Для форматування окремих елементів діаграми можна скористатися колекцією оформлення з групи *Стилі діаграм* (рис. 13.26).

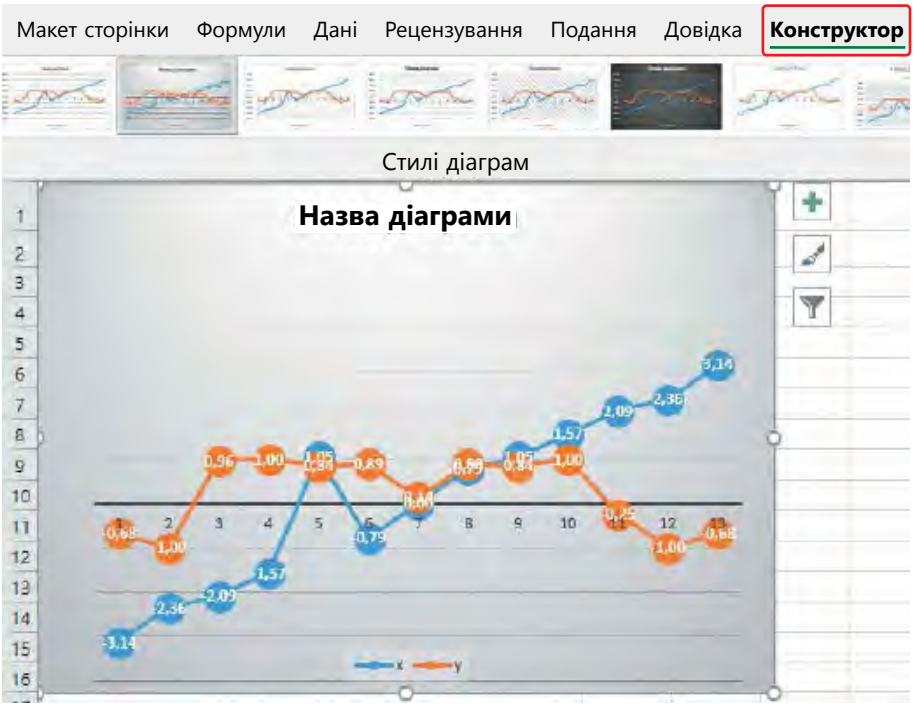


Рис. 13.26

ТЕСТОВІ ЗАВДАННЯ

ЕЛЕКТРОННА ТАБЛИЦЯ — ЦЕ:

1. прикладна програма, призначена для оброблення структурованих даних у вигляді таблиці
2. прикладна програма для оброблення кодових таблиць
3. пристрій ПК, що керує його ресурсами в процесі оброблення даних у табличній формі
4. системна програма, що керує ресурсами ПК при обробленні таблиць

ВИБЕРІТЬ ПРАВИЛЬНО ЗАПИСАНУ АДРЕСУ КОМІРКИ Excel:

1. 13A
2. A13
3. A:13
4. 13:A

ВИБЕРІТЬ ВАРІАНТ ВІДПОВІДІ, ЯКИЙ ВІДПОВІДАЄ ПОЗНАЧЕННЮ ПОМИЛКИ В Excel – #####:

1. ширина комірки не дає змоги відобразити число в заданому форматі
2. у формулі є спроба поділити на нуль
3. порушено правила задавання операторів, прийняті в математиці
4. Microsoft Excel не зміг розпізнати ім'я, використане у формулі

ОБЕРІТЬ ТИП ДІАГРАМ, ЯКИХ НЕМАЄ В Excel:

1. гістограма
2. точкова
3. кругова
4. ступенева

СУКУПНІСТЬ АРКУШІВ, РОЗМІЩЕНИХ В ОДНОМУ ФАЙЛІ В Microsoft Excel, НАЗИВАЮТЬ:

1. робочими аркушами
2. робочою книгою
3. електронною таблицею
4. електронним листом

ЩОБ ЗМІНИТИ ВИД АДРЕСАЦІЇ ДО КОМІРКИ, НЕОБХІДНО ВСТАНОВИТИ КУРСОР ПОРЯД З АДРЕСОЮ У ФОРМУЛІ ТА НАТИСНУТИ КЛАВІШУ:

1. *F*
2. *Shift*
3. *F4*
4. *Alt*

ЯКІ ІЗ НАВЕДЕНИХ ВАРІАНТІВ Є ПРАВИЛЬНИМИ ЗАПИСАМИ ФОРМУЛ В ЕЛЕКТРОННИХ ТАБЛИЦЯХ?

1. =(R18+L21)/G8-87,16*E12
2. =(R18+L21)/(G8-87,16E12)
3. =(R18+L21)(G8-87,16*E12)
4. =(R18+L21)/(G8-87,16*E12)

Розділ 14
БАЗИ ДАНИХ І СИСТЕМИ КЕРУВАННЯ БАЗАМИ ДАНИХ. ПРОГРАМА Microsoft Access

14.1. ОСНОВНІ ПОНЯТТЯ БАЗ ДАНИХ

У сучасному суспільстві для діяльності у будь-якій сфері (промисловість, наука, культура) потрібен потужний інформаційний супровід. Наприклад, сучасне промислове виробництво неможливе без розроблення креслень виробу з усіма його найменшими деталями, карток технологічних процесів та іншої технологічної документації. Для зберігання таких великих інформаційних потоків створюють відповідні засоби.

Для інформаційного забезпечення діяльності у кожній предметній галузі функціонують *інформаційні системи,* призначені для автоматизованого збирання, оброблення і надання інформації. До складу таких систем входять технічні засоби оброблення даних, програмне забезпечення і обслуговуючий персонал. Основним складником інформаційних систем є бази даних, що забезпечують зберігання, поповнення і оновлення інформації, пошук і вибір даних за запитами користувачів, оброблення даних і подання результатів.

База даних (англ. data base) — це сукупність взаємопов'язаних даних про властивості об'єктів предметної галузі, їхні взаємозв'язки і взаємний вплив. Сучасні бази даних містять інформацію про десятки, сотні тисяч і навіть мільйони об'єктів в електронній формі. Прототипом електронних баз даних є алфавітні і систематичні каталоги бібліотек, картотеки працівників у відділі кадрів підприємства, які зберігали, а подеколи і нині зберігають інформацію на паперовому носії.

Існують такі види баз даних: реляційна, ієрархічна, мережна. Найбільшого поширення у наш час набули реляційні бази даних. Розглянемо їх детальніше.

Реляційні бази даних

Реляційна база даних (від англ. relation — «відношення, залежність, сполучення, зв'язок») містить структуровану інформацію про об'єкти певної предметної галузі. Для подання реальних об'єктів у базі даних з усього різноманіття характеристик відбирають тільки ті, що з достатнім ступенем точності описують об'єкт у конкретній предметній галузі. Такі узагальнені об'єкти баз даних називають *сутностями* (англ. entity). Кожна сутність має набір характеристик — *атрибутів*. Набір атрибутів сутності — це *запис*, або *кортеж* (англ. tuple). Запис складається з *полів*, у яких записані значення атрибутів. Записи, об'єднані за певною ознакою, утворюють *відношення*. *Ключем*, або *первинним ключем,* називають атрибут або сукупність атрибутів, які однозначно ідентифікують запис.

Відношення подають у вигляді *таблиць* (англ. table). Кожен рядок таблиці містить запис, а в кожному стовпчику таблиці згруповано поля, у яких розмі-

щено однорідні дані про атрибути сутностей. Дані в таблицях мають відповідати таким вимогам:

- у кожному полі, тобто у кожній клітинці таблиці міститься значення одного атрибута сутності;
- дані в стовпчику мають однаковий тип (числа, текст тощо);
- кожен стовпчик таблиці має унікальне ім'я;
- кожен запис у таблиці унікальний, тобто в таблиці немає записів, у яких усі поля містять однакові значення;
- послідовність розміщення рядків і стовпчиків у таблиці є довільною.

Для адекватного відображення взаємозв'язків між об'єктами предметної галузі в реляційних базах даних встановлено *взаємні зв'язки* між таблицями, що є дуже важливим аспектом. Найпоширеніший тип зв'язку — *«один-до-багатьох»,* коли одна сутність з однієї таблиці пов'язана з кількома сутностями інших таблиць. Таблиця, у якій пов'язується тільки одна сутність, є *головною (первинною);* таблиця, у якій пов'язані кілька сутностей, — *підлеглою (вторинною)*, коли кожному запису з однієї таблиці відповідає один запис в іншій таблиці. У вторинній таблиці створюється поле, у якому розміщується первинний ключ із головної таблиці — *зовнішній ключ* (англ. foreign key). За типу зв'язку *«багато-до-багатьох»* кілька об'єктів однієї таблиці пов'язані зі кількома об'єктами іншої таблиці.

Для правильного функціонування бази даних важливо усунути *надмірність* (англ. redundancy) даних, яка може призвести до помилкової вибірки або зміни даних. Процес перетворення бази даних відповідно до комплексу вимог для усунення надмірності називають *нормалізацією бази даних.*

Важливим показником якості бази даних є її *цілісність* (англ. database integrity) — відповідність предметній галузі. Щодо бази даних передбачено *обмеження цілісності* (англ. integrity constraints).

Цілісність сутностей (англ. entity integrity) забезпечується, якщо кожна таблиця має первинний ключ. У процесі додавання записів до таблиць перевіряють унікальність їхніх первинних ключів. Заборонено змінювати значення атрибутів, що входять до первинного ключа.

Цілісність посилань (англ. referential integrity) забезпечується, якщо для кожного значення зовнішнього ключа у вторинній таблиці є запис у первинній таблиці з таким самим значенням первинного ключа.

Основні елементи реляційної моделі подано в табл. 14.1.

Таблиця 14.1

Основні елементи реляційної моделі

Елемент реляційної моделі	Форма подання
Атрибут	Заголовок стовпця таблиці
Відношення	Таблиця
Домен	Стовпець таблиці
Кортеж	Рядок таблиці
Первинний ключ	Один або кілька атрибутів
Сутність	Опис властивостей об'єкта
Схема відношення	Рядок заголовків таблиці
Тип даних	Тип значень елементів таблиці

Відношення є двовимірною таблицею, що містить деякі дані. *Сутність* — це об'єкт будь-якої природи. *Атрибут* — це поіменована характеристика сутності. *Домен* — це множина атомарних значень одного типу. *Кортеж* — це значення всіх атрибутів одного екземпляра сутності у відношенні. *Ключ (первинний ключ)* — це мінімальний набір атрибутів, за значеннями яких можна однозначно знайти кожний екземпляр сутності. *Нормалізація відношень* — це розбиття всієї інформації на дві або більше таблиць, що мають кращі властивості щодо додавання, зміни та вилучення даних.

Система керування базою даних

Для підтримки функціонування баз даних використовують комплекс програм — *систему керування базою даних (СКБД)* (англ. database management system, DBMS).

Крім забезпечення взаємодії користувача з базою даних, системи керування базами даних виконують такі функції:

- створення баз даних і їхніх основних елементів;
- введення, коригування і вилучення даних;
- упорядкування даних за певними критеріями;
- вибір сукупності даних, що відповідають певним вимогам;
- оформлення і подання вихідних даних тощо.

До складу СКБД входять дві мови програмування: мова опису даних (англ. data definition language) і мова маніпулювання даними (англ. data manipulation language). У сучасних реляційних базах даних широко застосовують алгоритмічну мову програмування SQL (structured query languare — мова структурованих запитів), яка має засоби і для опису даних, і для маніпулювання даними. SQL належить до декларативних мов програмування, у яких, на відміну від процедурних мов, визначається не послідовність операцій, а умови виконання запиту.

14.2. СИСТЕМА Microsoft Access

Система керування базами даних Microsoft Access призначена для реляційних баз даних. Інформацію про характеристики певного класу реальних об'єктів у реляційних базах даних заносять до спеціальних таблиць. Кількість характеристик (атрибутів), що визначають сутність об'єктів певного класу, відповідає кількості стовпців таблиці. Кожному об'єкту одного класу відповідає один рядок таблиці. Стовпці таблиці реляційних баз даних називають *полями*, а рядки — *записами*. Таблиці є основою реляційної бази даних і мають такі властивості:

- кожний елемент таблиці (клітинка) є елементом даних;
- кожний стовпець таблиці має унікальне ім'я;
- кожен стовпець у таблиці однорідний, тобто складається з однотипних елементів. Це можуть бути числа, текст, дати, логічні значення, графічні об'єкти тощо;

- у таблиці не може бути однакових рядків;
- рядки мають бути з однаковою кількістю елементів (полів), які є різнорідними і взаємопов'язаними;
- послідовність розміщення рядків і стовпців є довільною.

Основними об'єктами Microsoft Access є таблиці, форми, запити, звіти, макроси, модулі.

Таблиці баз даних призначені для збереження даних про характеристики об'єктів предметної галузі.

Форми створюють для введення, перегляду, коригування взаємопов'язаних даних у базі. За зовнішнім виглядом форми відповідають звичайному документові.

Запити слугують для вибору з бази даних інформації за певним критерієм або системою критеріїв. За допомогою запиту можна також додати, вилучити або відновити дані в таблиці, а також на основі вже наявних таблиць створити нові.

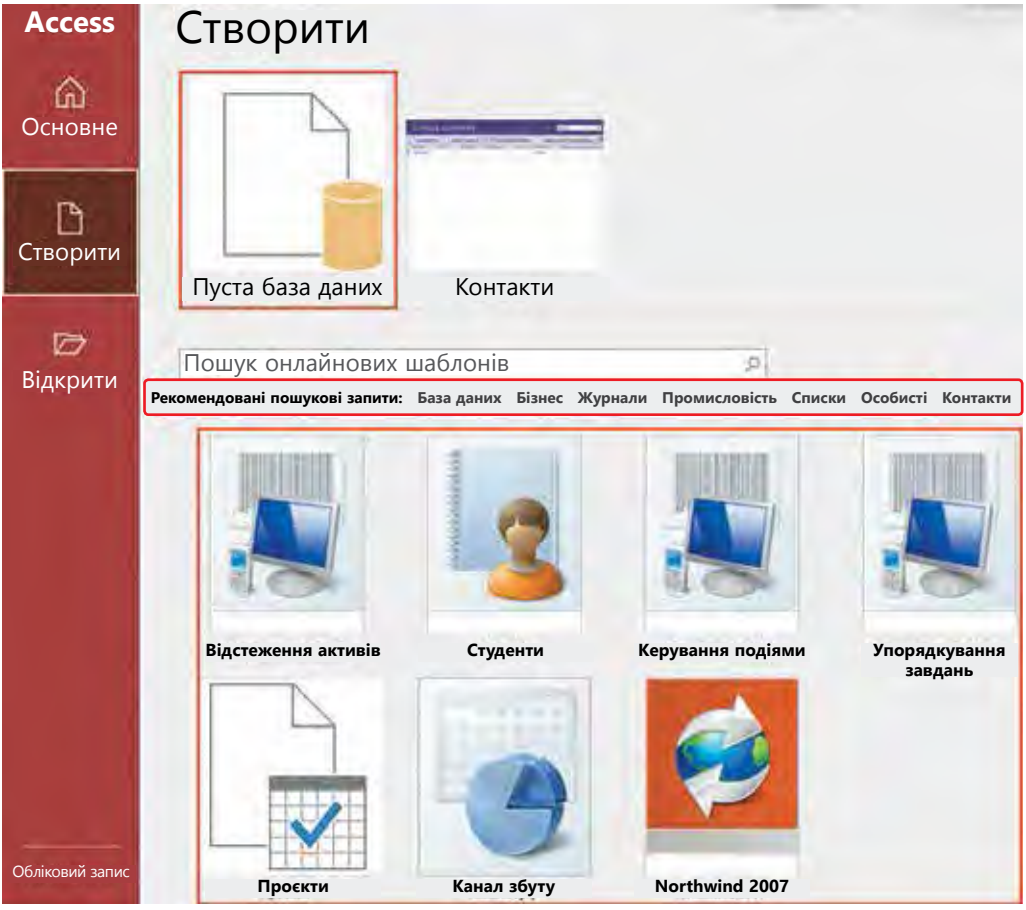


Рис. 14.1

Звіти призначені для формування вихідного документа, який виводять, як правило, на друк.

Макроси використовують для об'єднання певної послідовності дій під час виконання багатоетапної процедури оброблення інформації.

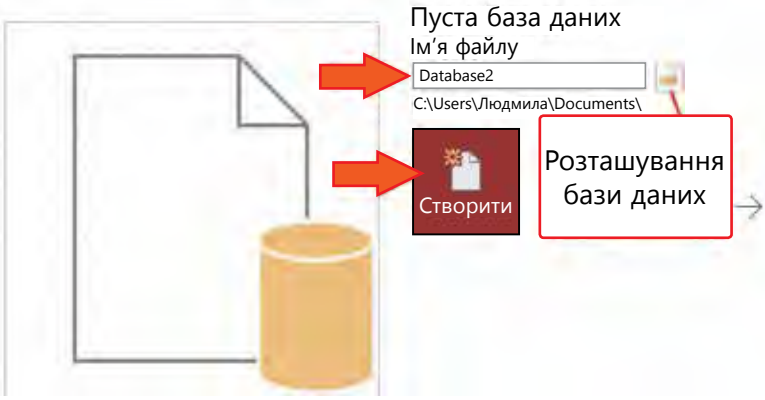
Модулі містять програми, написані однією з алгоритмічних мов, для реалізації нестандартних процедур оброблення інформації.

14.3. ОСНОВНІ ЕЛЕМЕНТИ СКБД MS Access

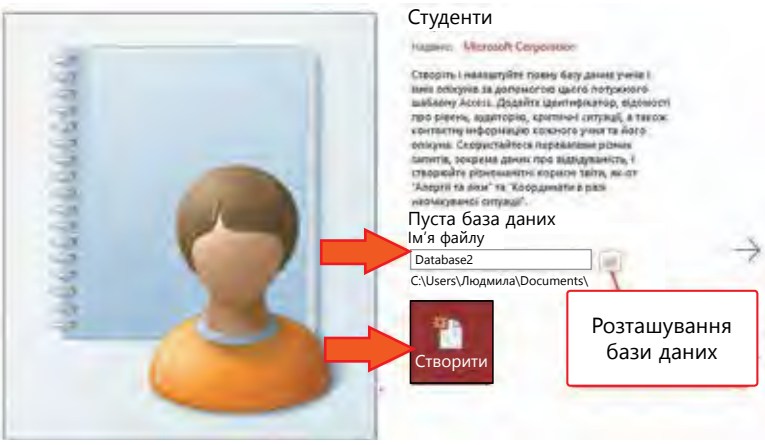
Систему Access запускають із меню *Пуск* або за допомогою ярлика

. У стартовому вікні Access можна вибрати один із варіантів:

- створення нової порожньої бази даних — *Пуста база* (рис. 14.2, *а*);
- створення нової бази даних на основі шаблону, у нижній частині якого відображаються мініатюри шаблонів та список категорій доступних шаблонів (рис. 14.2, *б*).



а



б

Рис. 14.2

Далі зазначають ім'я та місце розташування нової бази даних і натискають кнопку *Створити*. Нова база даних, наприклад *Замовлення*, містить одну порожню таблицю (рис. 14.3). Файли бази даних Access 2016 мають розширення **.accdb.**

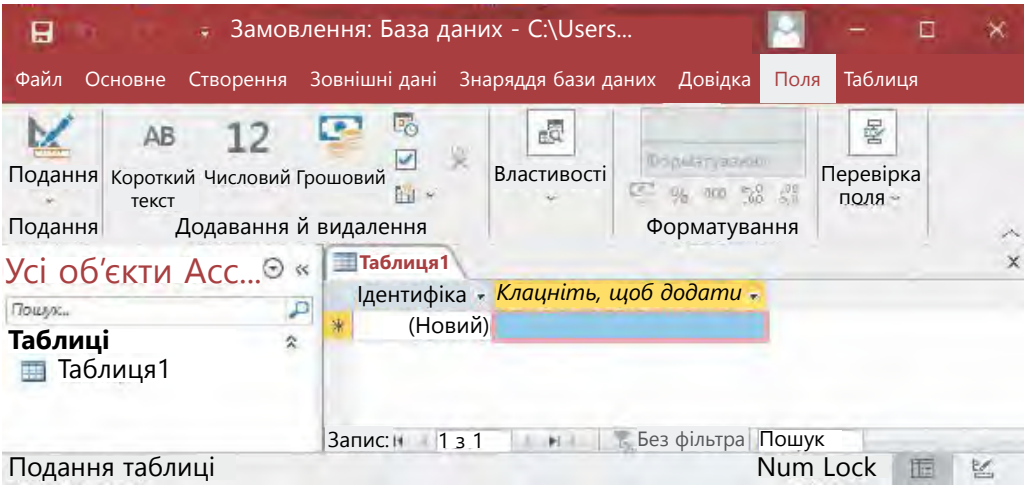


Рис. 14.3

Основні елементи вікна Access (рис. 14.4):

1. Рядок заголовка — містить ім'я відкритої бази даних.
2. Кнопки керування вікном — *Згорнути, Розгорнути, Згорнути у вікно, Закрити.*
3. Кнопки команд — значки із зображеннями.
4. Стрічка містить кнопки команд.
5. Вкладки — назви стрічок, які об'єднують командні кнопки за функціональним принципом: **Файл, Подання, Створення, Зовнішні дані, Знаряддя бази даних, Довідка.**
6. Кнопка *Згорнути стрічку.*
7. Панель швидкого доступу — швидкий доступ до команд із різних вкладок.
8. Контекстна вкладка — відповідає конкретному об'єкту та його поточному стану. Наприклад, якщо в базі даних відкрити таблицю, то автоматично з'явиться вкладка *Робота з таблицями.*
9. Розділи, призначені для прискорення доступу команд, які містять інструменти на стрічках (наприклад, у розділі *Форматування тексту* вкладки *Основне* містяться основні інструменти: накреслення і розмір шрифту, вирівнювання тексту, його колір тощо). Всі розділи мають назви.
10. Вкладки відкритих об'єктів.
11. Область об'єктів (область переходів).
12. Назви всіх об'єктів, які відображаються в області об'єктів «.
13. Рядок стану — основні елементи керування, за допомогою яких можна швидко переходити між поданнями активного вікна, повідомлення про стан, пошук та масштабування.

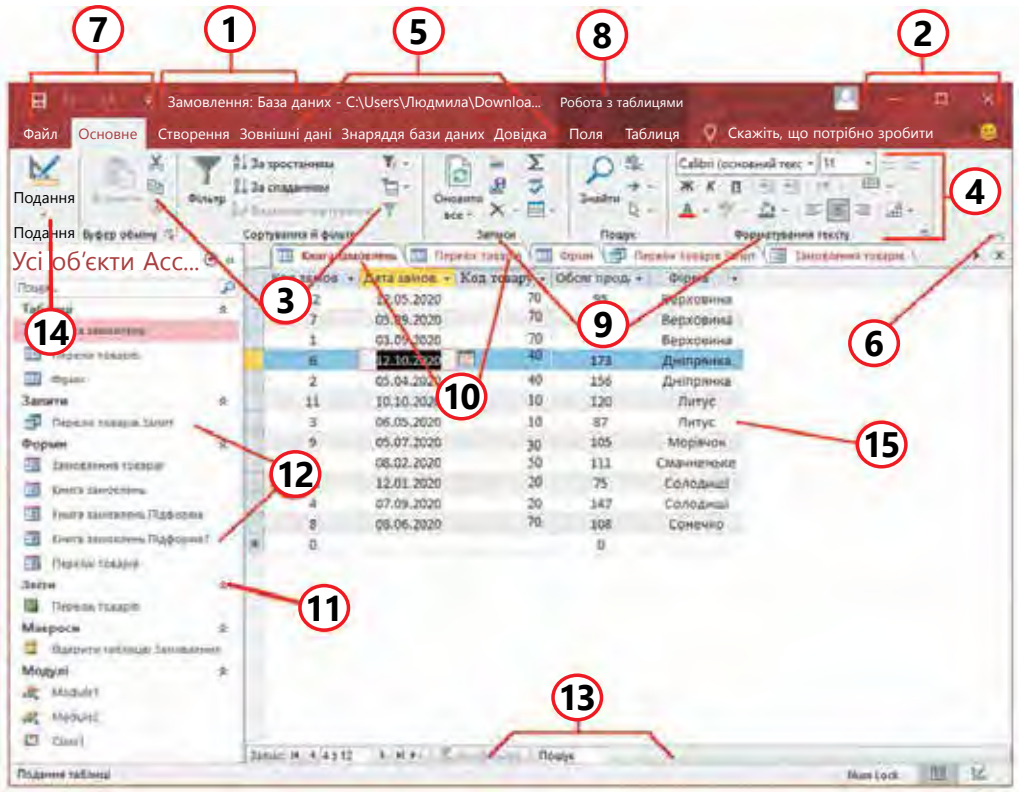


Рис. 14.4

14. Кнопка подання. Для кожного з об'єктів бази даних доступні різні подання (наприклад, таблиці можна відкрити в *Подання таблиці* або *Конструктор*).

15. Робоча область, де здійснюється робота зі всіма об'єктами бази даних.

14.4. ТАБЛИЦІ Й РОБОТА З НИМИ

Основні операції оброблення інформації у Microsoft Access виконують за допомогою двовимірних прямокутних таблиць. У таблицях розміщено інформацію про найважливіші характеристики певного класу об'єктів. Кожному об'єкту відповідає один *рядок* таблиці. Дані, що містяться в одному рядку, — це запис (рис. 14.5, *1*). Отже, кожен об'єкт із певного класу об'єктів характеризується одним записом. Кожен рядок складається з однакової кількості клітинок — *полів* (рис. 14.5, *2*). До полів заносять інформацію (дані) про певну характеристику *(атрибут)* об'єкта (рис. 14.5, *3*). Однотипні поля, розміщені одне під одним, становлять *стовпець* таблиці, якому надають *ім'я*. Тож кожен стовпець (колонка) таблиці описує характеристику (атрибут) об'єктів певного класу.

Створюючи структуру таблиці (рис. 14.5), обов'язково зазначають імена й типи полів. Оскільки у стовпці таблиці розміщені поля одного типу й імені, то ім'я і тип поля є іменем і типом стовпця. Тип поля визначає тип даних, що зберігаються в цьому полі.

Типи даних, які використовують в Access, відображено в табл. 14.2.

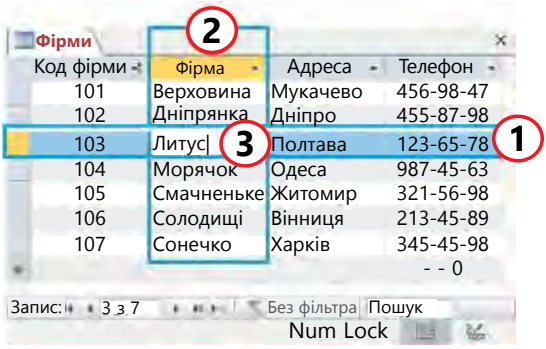


Рис. 14.5

Таблиця 14.2

Доступні типи даних

Тип даних	Використання
Короткий текст	Для зберігання алфавітно-цифрових символів завдовжки від 0 до 255, таких даних, як імена та адреси, а також для чисел, які не потребують обчислень
Довгий текст (Поле Мемо)	Для зберігання абзаців тексту: резюме, коротких описів, до 65 536 алфавітно-цифрових символів
Число	Для зберігання числових даних. Числові значення (цілі або дробові). 1, 2, 4 або 8 байтів
Велике число	Числові значення. 8 байтів
Дата й час	Зберігаються дата й час різних форматів. 8 байтів
Грошова одиниця	Для проведення розрахунків із грошовими значеннями або для обчислень із фіксованою комою, у яких потрібна висока точність. Грошові значення (грн, $ тощо). 8 байтів
Автонумерація	Унікальне числове значення, яке у Access автоматично вставляється в разі додавання запису. 4 байти
Так / Ні	Логічні значення встановлюють для полів, які можуть містити одне з двох можливих значень, наприклад: «Так / Ні» або «True / False». 1 біт
Об'єкт OLE	Об'єкти OLE — для збереження графічних об'єктів (наприклад, фотографій) з інших програм, які підтримують цю технологію. Не більше ніж 1 Гбайт
Гіперпосилання	Встановлюють для збереження гіперпосилань, які надають безпосередній доступ до вебсторінок і створення зв'язків з об'єктами Access, які зберігаються в базі даних
Вкладення	Для вкладення файлів. Файли у форматі MDB не підтримують тип даних «Вкладення». До 2 Гбайтів
Обчислюваний	Для створення виразу, який використовує дані з одного або кількох полів. Файли у форматі MDB не підтримують тип даних «Обчислюваний»
Майстер підстановок	Фактично не є типом даних, використовують для більш ефективного і коректного введення даних

В Access для створення таблиць передбачено режими *Таблиця* і *Конструктор*, також можна здійснювати імпорт таблиць із зовнішніх джерел.

Для створення таблиці в режимі *Таблиця* потрібно вибрати на вкладці *Створення* в групі *Таблиці* команду *Таблиця*. Першим полем автоматично створиться *Ідентифікатор* із типом даних *Автонумерація* (рис. 14.6).

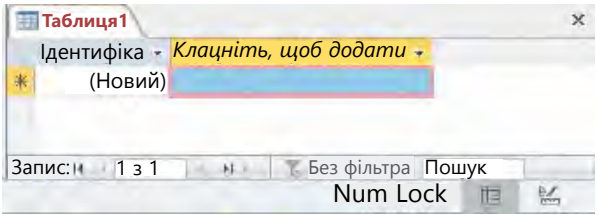


Рис. 14.6

Щоб перейменувати поле *Ідентифікатор*, потрібно двічі клацнути лівою кнопкою миші на полі й ввести назву *Код замовлення* або за допомогою команди контекстного меню *Перейменувати поле*. Далі створити друге поле, натиснувши на кнопку *Клацніть, щоб додати* (тип поля — *Число*) і ввести його назву *Код товару*. Аналогічно можна додати до таблиці такі поля: *Обсяг продажу* (тип поля — *Число*), *Дата продажу* (тип поля — *Дата і час*) і *Код фірми* (тип поля — *Число*). З контекстного меню ярлика *Таблиця1* обрати команду *Зберегти*, у вікні діалогу, що відкрилося (рис. 14.7), увести ім'я таблиці *Книга замовлень* і натиснути кнопку *ОК*.

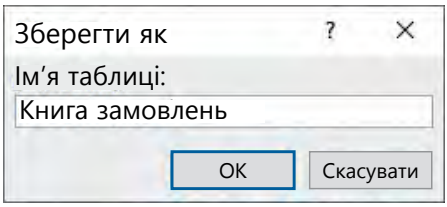
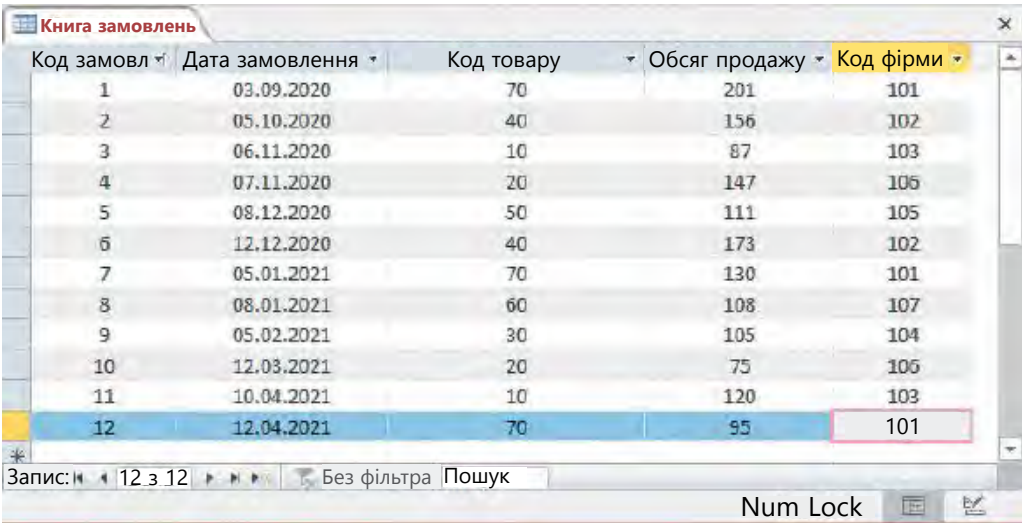


Рис. 14.7

Заповнена створена таблиця з вихідними даними може мати, наприклад, такий вигляд (рис. 14.8).

Конструктор є найефективнішим засобом побудови таблиць із широкими можливостями вибору параметрів. Щоб відкрити таблицю в режимі кон-



Книга замовлень

Код замовл	Дата замовлення	Код товару	Обсяг продажу	Код фірми
1	03.09.2020	70	201	101
2	05.10.2020	40	156	102
3	06.11.2020	10	87	103
4	07.11.2020	20	147	106
5	08.12.2020	50	111	105
6	12.12.2020	40	173	102
7	05.01.2021	70	130	101
8	08.01.2021	60	108	107
9	05.02.2021	30	105	104
10	12.03.2021	20	75	106
11	10.04.2021	10	120	103
12	12.04.2021	70	95	101

Запис: 12 з 12 Без фільтра Пошук
Num Lock

Рис. 14.8

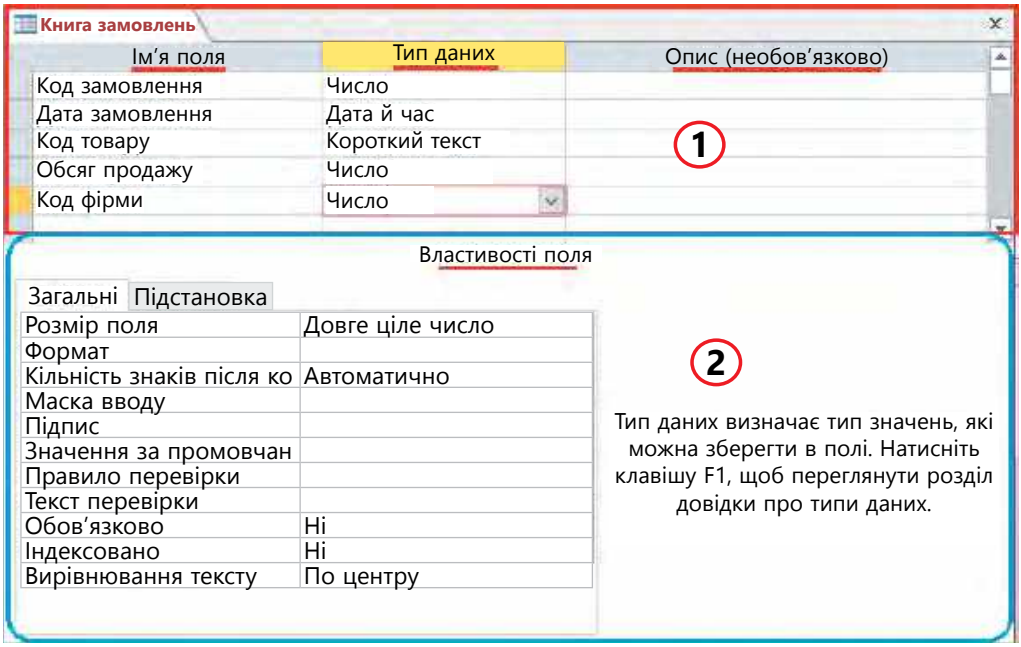


Рис. 14.9

структор *Книга замовлення* потрібно вибрати на вкладці *Основне — Подання — Конструктор*.

Для створення таблиці в режимі *Конструктор* треба перейти на вкладку *Створення* і в групі *Таблиці* вибрати команду *Конструктор таблиць* (Конструктор таблиць). З'явиться вікно *Конструктора*, що складається з двох частин. У верхній частині (рис. 14.9, *1*) відображаються список полів таблиці, тип даних для кожного поля та опис поля (описувати поля не обов'язково), а в нижній (рис. 14.9, *2*) — властивості вибраного поля (наприклад, розмір і формат поля, маска введення даних у поле, умови на значення тощо).

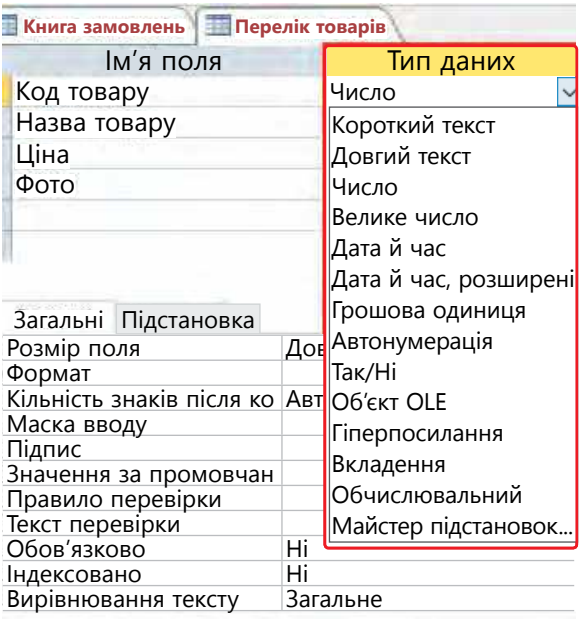


Рис. 14.10

У розділі *Ім'я поля* вводять імена створюваних полів, а в розділі *Тип даних* для кожного поля вибирають тип даних (рис. 14.10). У розділі *Опис* можна задати відповідний коментар.

Вміст вкладки *Властивості поля* змінюється залежно від типу поля (табл. 14.3).

Таблиця 14.3

Властивості текстового поля

Властивість поля	Використання
Розмір поля	Короткий текст (до 255 символів)
Формат	Налаштування вигляду даних у полі під час відображення або друку
Маска вводу	Відображення символів редагування для керування введенням даних
Підпис	Введення тексту, який стандартно відображається в підписах форм, звітів і запитів
Стандартне значення	Автоматичне визначення для поля стандартного значення в разі додавання нових записів
Правило перевірки	Налаштування виразу, який має бути істинним у разі додавання або змінення значення в цьому полі
Текст перевірки	Введення тексту, який відображається в разі порушення виразу з поля властивості *Правило перевірки*
Обов'язково	Потрібно ввести дані в поле
Дозволити нульову довжину	Дає змогу ввести рядок із нульовою довжиною («») в текстовому полі або в полі приміток (якщо має значення Так)
Індексовано	Прискорює доступ до даних у цьому полі за допомогою створення та використання індексу
Стискання Юнікод	Стискання тексту, який зберігається в цьому полі, якщо зберігається великий текстовий фрагмент (>4096 символів)
Режим редактора IME	Визначає перетворення символів в азійській версії Windows
Вирівнювання тексту	Визначення вирівнювання тексту

Після створення всіх полів і визначення їхніх властивостей у контекстному меню треба вибрати команду *Зберегти*. З'явиться діалогове вікно, в якому, замість імені Таблиця1, треба ввести ім'я нової таблиці, наприклад *Перелік товарів* і натиснути кнопку ОК. Приклади даних для заповнення таблиць *Перелік товарів* і *Фірма* наведено відповідно в таблицях 14.4 і 14.5.

Таблиця 14.4

Вихідні дані таблиці *Перелік товарів*

Код фірми	Фірма	Адреса	Телефон
101	Верховина	Мукачеве	456-98-47
102	Дніпрянка	Дніпро	455-87-98
103	Литус	Полтава	123-65-78
104	Морячок	Одеса	987-45-63
105	Смачненьке	Житомир	321-56-98
106	Солодощі	Вінниця	213-45-89
107	Сонечко	Харків	345-45-98

Таблиця 14.5

Вихідні дані таблиці *Фірма*

Код товару	Назва товару	Ціни
10	Ананаси	75,00 грн
20	Банани	29,00 грн
30	Виноград	58,00 грн
40	Ківі	45,00 грн
50	Кавун	25,00 грн
60	Диня	63,00 грн
70	Хурма	57,00 грн

Створення структури таблиці, як правило, завершується визначенням *первинного ключа*, який однозначно визначає кожен запис таблиці. *Первинний ключ* — це одне або кілька полів, зміст яких унікальний (не повторюється) для кожного запису. Дотримання цієї умови забезпечує цілісність даних. Первинний ключ вводять за командою *Ключове поле* контекстного меню вибраного поля або за допомогою кнопки

на вкладці *Конструктор*. Якщо ключ складено, тобто створено з кількох полів, то спочатку необхідно виділити імена цих полів лівою кнопкою миші й, тримаючи натиснутою клавішу *CTRL*, натиснути кнопку *Ключове поле*.

Якщо користувач не визначив ключове поле, то система автоматично створює ключове поле з ім'ям *Ідентифікатор* і типом даних *Автонумерація*.

У режимі *Конструктор* можна створювати і змінювати тільки структуру таблиці. Заповнення таблиці даними виконується у режимі *Подання таблиці*.

Усі записи таблиці бази даних можна імпортувати з інших баз даних, а також із файлів Excel або текстових файлів. Для цього необхідно на вкладці *Зовнішні дані* в групі *Імпорт і зв'язування* вибрати один із варіантів (рис. 14.11). Відкриється майстер імпорту та зв'язування *Отримати зовнішні*

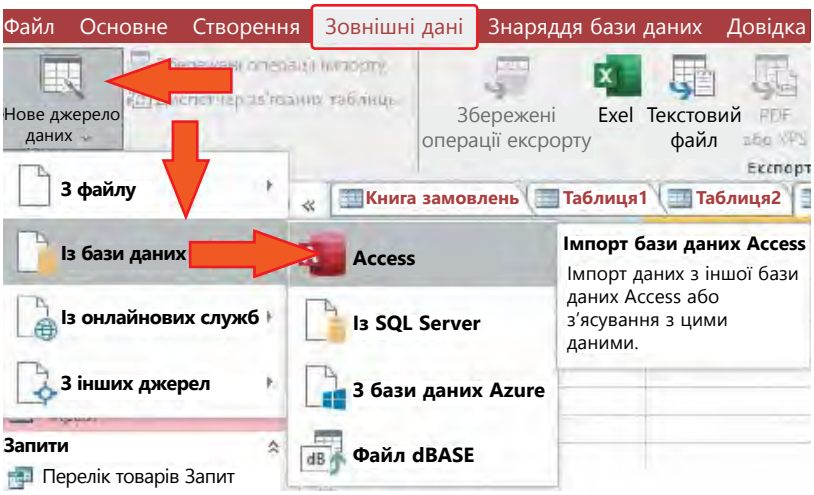


Рис. 14.11

дані — База даних Access. У текстове поле *Ім'я файлу* ввести ім'я вихідної бази даних. Далі вибрати команду *Імпортувати таблиці* і натиснути кнопку *OK*. Відкриється діалогове вікно *Імпортувати об'єкти*. На вкладці *Таблиці* вибрати таблиці, які потрібно імпортувати, й натиснути кнопку *OK*.

Створені таблиці можна редагувати в разі необхідності в режимі *Конструктор* або *Подання таблиці*.

Щоб виправити помилки в таблиці, використовують вікно *Конструктор*, що містить усі поля та параметри редагованої таблиці. Засоби цього вікна дають можливість:

- змінювати імена полів, їхні тип і параметри;
- вилучати поля з таблиці й додавати нові;
- змінювати послідовність полів;
- змінювати або задавати нові ключові поля тощо.

Ім'я й тип поля можна змінювати за допомогою клавіш *Backspase* або *Delete*. Ім'я редагують або вводять заново, а новий тип поля вибирають зі списку стовпця *Тип даних*. Для кожного типу даних за допомогою елементів у нижній частині вікна *Конструктора* зазначають властивості.

Щоб змінити послідовність розміщення полів, виділяють потрібний рядок за допомогою миші — з'являється стрілка ►. Треба клацнути мишею на стрілці, захопити, тримаючи натиснутою ліву кнопку, маленький штриховий прямокутник під стрілкою і перемістити виділений рядок у потрібне місце.

Щоб вилучити поле з таблиці, потрібно спочатку його виділити, а потім натиснути на клавішу *Delete*. Можна вилучати також групи суміжних і несуміжних полів. Групу суміжних полів виділяють так: спочатку виділяють перше поле з діапазону, а далі за натиснутої кнопки *Shift* — останнє поле діапазону. Поля другої несуміжної групи виділяють за натиснутої кнопки *Ctrl*.

Новий рядок додають за допомогою команди *Вставити рядки* у контекстній вкладці *Робота з таблицями — Конструктор* над виділеним рядком із позначкою ► або команди *Додати рядок* із контекстного меню виділеного рядка.

14.5. СТВОРЕННЯ ЗВ'ЯЗКІВ МІЖ ТАБЛИЦЯМИ

Таблиці містять інформацію про об'єкти предметної галузі, які пов'язані між собою у процесі їх функціонування, тому для адекватного відображення реальних процесів таблиці теж мають бути взаємопов'язані. Зв'язки між таблицями встановлюють відповідно до інформаційно-логічної моделі предметної галузі.

Програма автоматично визначає за вибраним полем тип зв'язку між таблицями. Якщо пов'язані поля головної і підпорядкованої таблиць є унікальними ключами, то встановлюється зв'язок 1:1 («один-до-одного»). Якщо поле зв'язку в головній таблиці є унікальним ключем, а в підпорядкованій — ні, то встановлюється зв'язок 1:M («один-до-багатьох»). Для обох типів зв'язку є можливість задати параметр забезпечення цілісності даних, що означає вико-

нання таких умов: не можна вилучати записи з головної таблиці, якщо не вилучено пов'язані з ними записи в підпорядкованій таблиці; у підпорядковану таблицю не можна вводити запис із значенням ключа, якого немає в головній таблиці; зміну значень ключа зв'язку головної таблиці слід здійснювати відповідно до змін підпорядкованої таблиці.

Первинні ключі використовують для зв'язку таблиць. Поля в іншій таблиці, які пов'язують із полем первинного ключа, називають *зовнішнім ключем*.

Щоб встановити зв'язки між таблицями, потрібно перейти на вкладку *Знаряддя бази даних — Зв'язки* та в групі *Зв'язки* натиснути на кнопку *Зв'язки*. У вікні *Відображення таблиці* вибрати всі таблиці й натиснути кнопку *Додати*, наприклад *Книга замовлень, Перелік товарів* і *Фірма* (рис. 14.12). Таблиці відобразяться у вікні *Зв'язки*. Лівою кнопкою миші натиснути ключове поле *Код товару* таблиці *Перелік товарів* і перенести його на поле *Код товару* таблиці *Книга замовлень*. У результаті на екрані з'явиться вікно *Редагування зв'язків* (рис. 14.13), у якому потрібно встановити перемикачі:

- Забезпечення цілісності даних;
- Каскадне оновлення пов'язаних полів;
- Каскадне видалення пов'язаних полів.

Також у цьому вікні відображається тип зв'язку «один-до-багатьох».

Для створення зв'язку між таблицями *Книга замовлень* і *Фірма* потрібно виконати такі самі дії. Після закриття вікна *Зв'язки* Access запропонує зберегти зміни, внесені до розмічування «*Зв'язки*», для чого необхідно натиснути на кнопку *Так* (рис. 14.14).

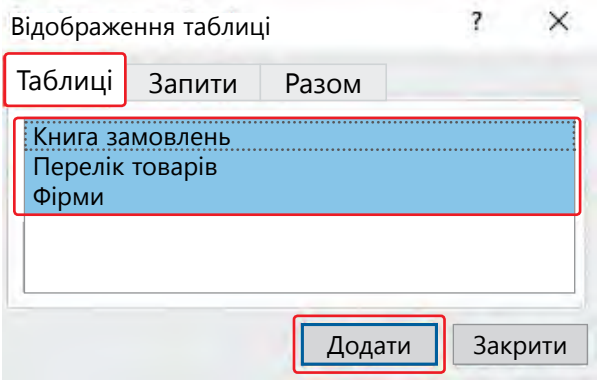


Рис. 14.12

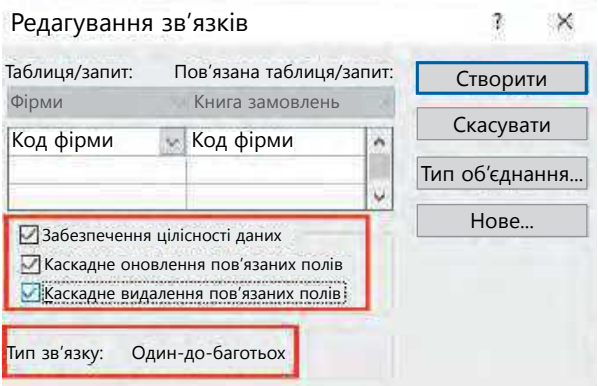


Рис. 14.13

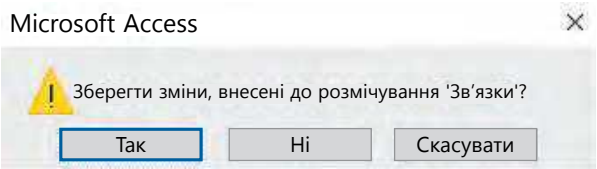


Рис. 14.14

Результат вікна *Зв'язки* для бази даних *Замовлення* наведено на рис. 14.15.

Зв'язки між таблицями також створюють за допомогою *Майстра підстановок*.

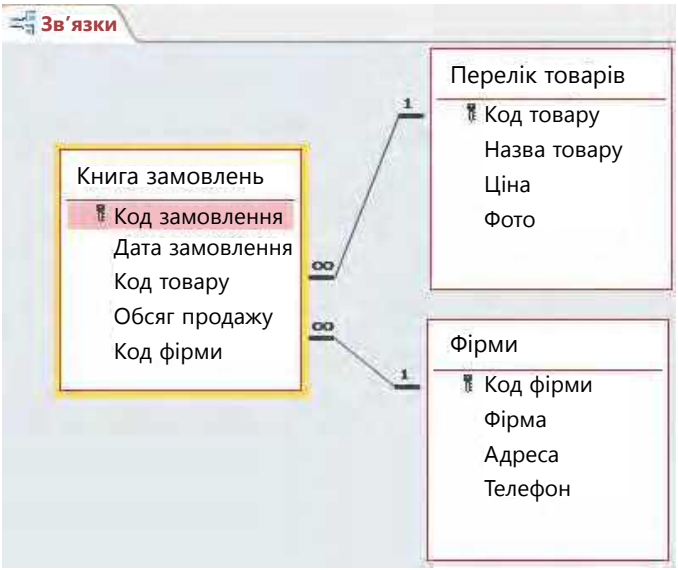


Рис. 14.15

14.6. ЗАСТОСУВАННЯ ЗАПИТІВ ДЛЯ ОБРОБЛЕННЯ І ПОДАВАННЯ ІНФОРМАЦІЇ

Інформація, що зберігається в базах даних, відображає стан і функціонування реальних об'єктів. Для оброблення інформації в таблицях, застосовують потужний і гнучкий інструмент — *запити*.

За допомогою запитів можна виконувати такі операції: вибрати записи, що задовольняють певні вимоги, із кількох таблиць; додати до підсумкової таблиці поля й у разі необхідності виконати обчислення для них; згрупувати запити з певними ознаками у визначеному полі; користуючись наявними таблицями, створити нову таблицю; вилучити з пов'язаних таблиць записи, що відповідають певним умовам тощо.

У Microsoft Access є такі види запитів:

- запит на вибірку, в результаті виконання якого вибираються дані з пов'язаних таблиць, а також таблиць, побудованих під час здійснення інших запитів;
- запит на створення нової таблиці;
- запит на оновлення, який дає змогу вносити зміни до групи записів;
- запит на додавання, за допомогою якого додають записи до таблиць бази даних;
- запит на вилучення даних;

- запит із параметрами, під час виконання якого користувач може ввести у діалоговому вікні конкретне значення й потім одержати потрібний результат;
- запит із полем, що обчислюється;
- перехресний запит, який дає змогу групувати за кількома ознаками.

В Access можна створювати запити:

- за допомогою *Майстра запитів;*
- у режимі *Конструктора (Макета запиту);*
- мовою SQL.

Для створення простого запиту за допомогою **Майстра запитів** на вкладці **Створення** в групі *Запити* треба виконати команду *Майстер запитів*. У діалоговому вікні *Новий запит* вибрати тип *Майстер простих запитів* і натиснути кнопку *OK*. На екрані з'явиться вікно *Майстер простих запитів*. Імена потрібних для запиту полів передають у вікно *Вибрані поля* за допомогою кнопок: додати одне поле [>], всі поля [>>] (для вилучення полів — прибрати одне поле [<], всі поля [<<]). Вибрати можна, наприклад, таблиці:

- *Перелік товарів* і перенести у праве вікно *Вибрані поля: Назва товару* та *Ціна*;
- *Книга замовлень* і перенести вправо поля: *Обсяг продажу* та *Дата замовлення (*рис. 14.16).

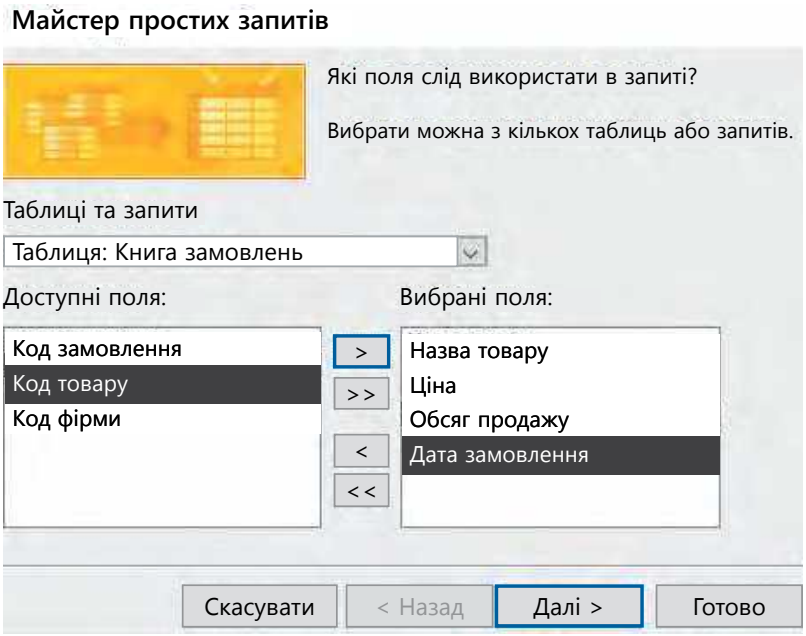


Рис. 14.16

Натиснути на кнопку *Далі*. На наступному кроці потрібно встановити опцію *Зведення*, вибрати *Параметри зведення,* поставити позначки напроти функції *Sum* у полі *Обсяг продажу* та натиснути кнопку *OK* (рис. 14.17).

Майстер простих запитів

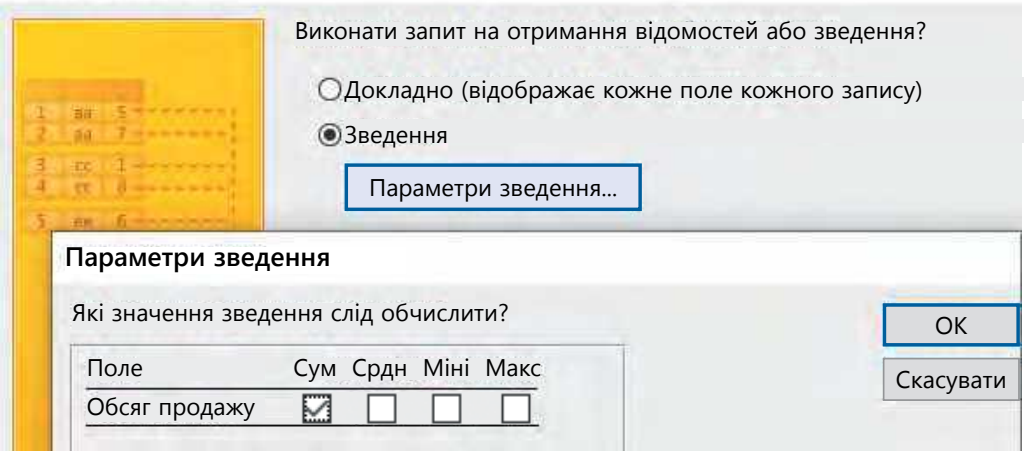


Рис. 14.17

Далі необхідно задати ім'я запиту — *Сумарні обсяги продажу* та натиснути кнопку *Готово*. Результат виконання простого запиту, створеного за допомогою *Майстра запитів*, наведено на рис. 14.18.

Сумарні обсяги продажу

Назва товару	Ціна	Сума З Обсяг продажу	Перший З Дата замовлення
Ананаси	75,00 грн	207	10.04.2021
Банани	29,00 грн	222	12.03.2021
Виноград	58,00 грн	105	05.02.2021
Ківі	45,00 грн	329	12.12.2020
Кавун	25,00 грн	111	08.12.2020
Диня	63,00 грн	108	08.01.2021
Хурма	57,00 грн	426	12.04.2021

Рис. 14.18

Конструктор запитів є простим засобом створення запитів. Потрібно вибрати команду *Макет запиту* у групі *Запити* на вкладці **Створення**. З'явиться вікно, яке складається з двох частин. У верхній (рис. 14.19, *1*) відображається структура вибраних таблиць, яким адресований запит, а нижня частина (рис. 14.19, *2*) є бланком запиту за зразком, яка розбита на стовпці. Кожний стовпець бланка стосується одного поля. Щоб заповнити бланк запиту, необхідно:

- до рядка *Поле* внести імена полів;
- вибрати поля у рядку *Таблиця*;
- у рядку *Сортування* зазначити порядок сортування відібраних записів, наприклад назви товарів в алфавітному порядку або в порядку зменшення ціни;
- у рядку *Відображення* поставити позначку ☑;

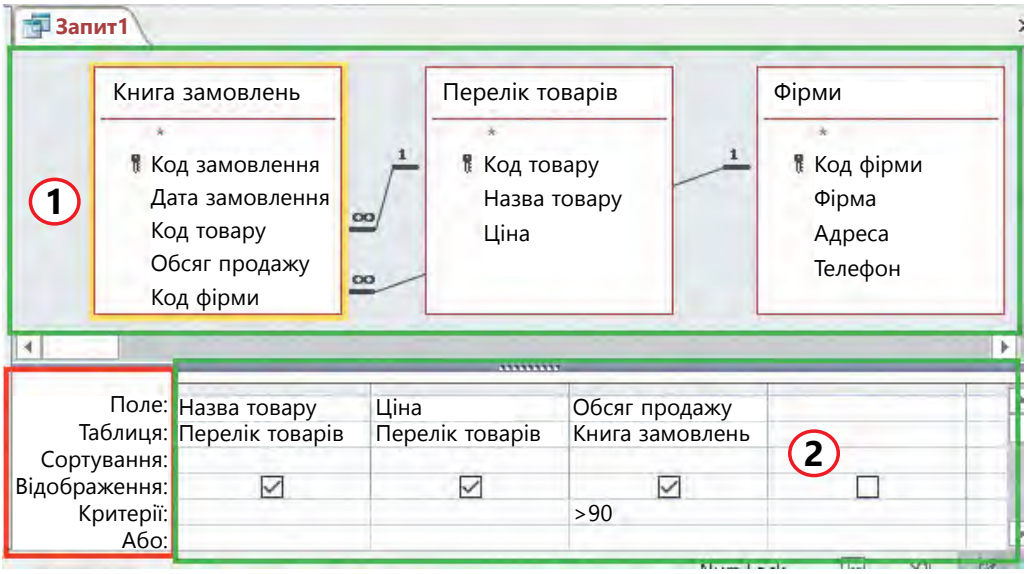
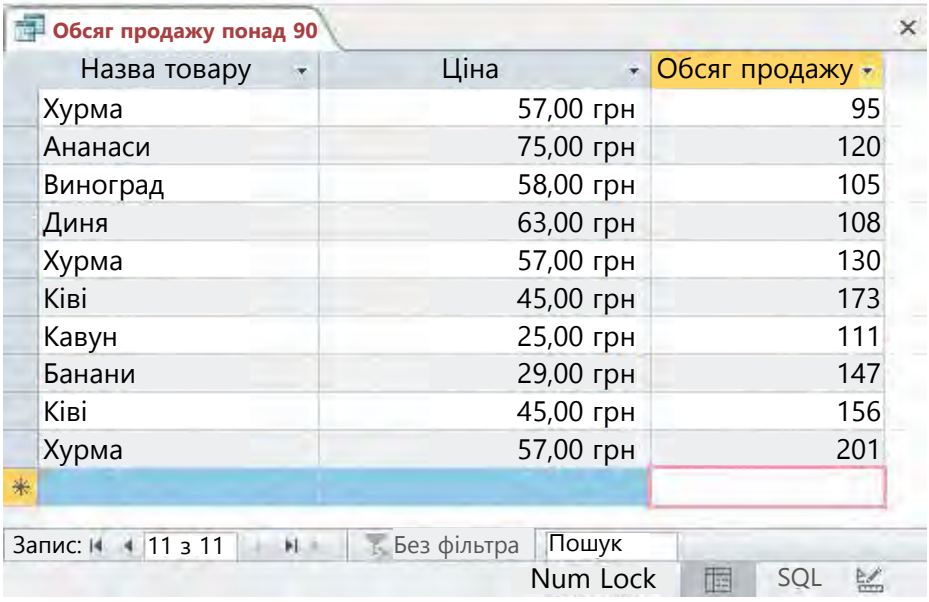


Рис. 14.19

- у рядку *Критерії* сформувати критерії вибірки записів, які можуть містити символи («*», «#», «!», «?» та ін.), а також логічні функції. Наприклад, для відбору даних *Обсягу продажу* понад 90 треба задати критерій >90.

Запустити запит на виконання можна командою *Запуск* !, яка міститься на вкладці *Конструктор* у групі *Результати*. Результат виконання запиту в режимі *Таблиця* наведено на рис. 14.20.



Обсяг продажу понад 90

Назва товару	Ціна	Обсяг продажу
Хурма	57,00 грн	95
Ананаси	75,00 грн	120
Виноград	58,00 грн	105
Диня	63,00 грн	108
Хурма	57,00 грн	130
Ківі	45,00 грн	173
Кавун	25,00 грн	111
Банани	29,00 грн	147
Ківі	45,00 грн	156
Хурма	57,00 грн	201

Запис: 11 з 11 Без фільтра Пошук
Num Lock SQL

Рис. 14.20

Передбачено різні способи відбору. Наприклад, відібрати дані з текстового поля *Назва товару* "ананаси" або "банани" можна за допомогою таких критеріїв:

- "ананаси" OR "банани";
- IN ("ананаси"; "банани");
- LIKE ("*ананаси");
- параметр *Або* у вікні *Конструктора запитів*.

Критерії відбору можуть бути складними. Наприклад, відібрати товар, який було замовлено в межах одного року, можна за допомогою критерію: BETWEEN #01.01.2020# AND #31.12.2020#.

Для вилучення поля потрібно його виділити й натиснути клавішу *Delete*.

Вартість товару підраховують за допомогою вікна *Побудовника виразів* — команда *Побудувати* в контекстному меню (рис. 14.21).

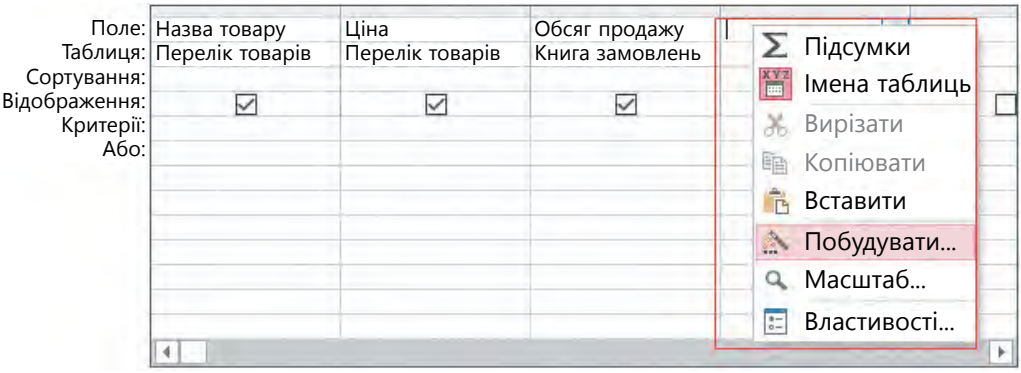


Рис. 14.21

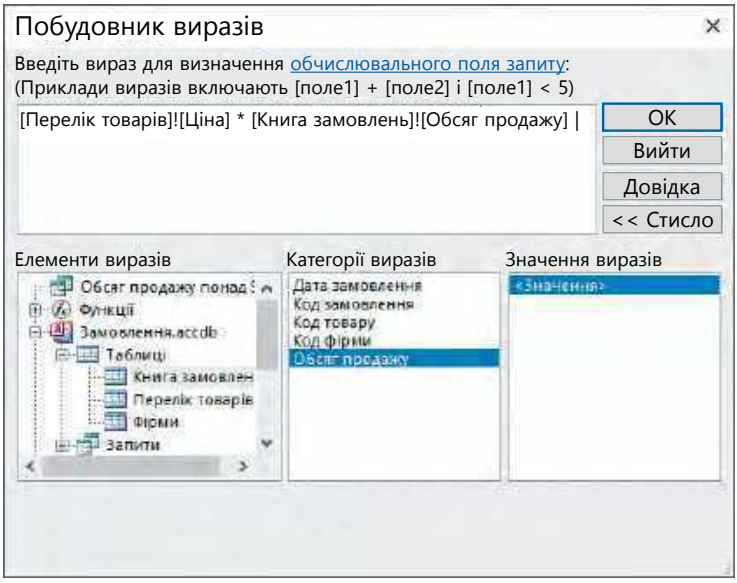


Рис. 14.22

У вікні ліворуч потрібно розкрити список об'єктів бази даних *Замовлення* і обрати таблицю *Перелік товарів*. У полі *Категорії виразів* подвійним клацанням на полі *Ціна* помістити його у верхнє поле вікна та додати з клавіатури знак множення (див. рис. 14.22). Аналогічно знайти та ввести друге поле — *Обсяг продажу* з таблиці *Книга замовлень*. Вікно *Побудовник виразів* із отриманим результатом наведено на рис. 14.22.

Натиснути на кнопку *ОК*. Вираз передається в бланк запиту (див. рис. 14.22), де він розміщується після системного імені поля та відокремлюється від нього двома крапками: **Вираз1:[Перелік товарів]![Ціна] * [Книга замовлень]![Обсяг продажу]**. В полі Вираз1 змінити на Вартість.

Створений запит запускають за допомогою кнопки *Запуск* у групі *Результати* на вкладці *Конструктор* (рис. 14.23).

Хурма	57,00 грн	95	5 415,00 грн
Ананаси	75,00 грн	120	9 000,00 грн
Банани	29,00 грн	75	2 175,00 грн
Виноград	58,00 грн	105	6 090,00 грн
Диня	63,00 грн	108	6 804,00 грн
Хурма	57,00 грн	130	7 410,00 грн
Ківі	45,00 грн	173	7 785,00 грн
Кавун	25,00 грн	111	2 775,00 грн
Банани	29,00 грн	147	4 263,00 грн
Ананаси	75,00 грн	87	6 525,00 грн
Ківі	45,00 грн	156	7 020,00 грн

Рис. 14.23

14.7. ЗАСТОСУВАННЯ ФОРМ ДЛЯ РОБОТИ З БАЗОЮ ДАНИХ

Дані, що зберігаються в базах даних, потрібно змінювати залежно від того, як змінюються об'єкти. Для занесення оновлених даних до пов'язаних таблиць бази даних, коригування, оброблення і вилучення використовують такий інструмент, як *форми*. Конструкцію форми визначає користувач, зазначаючи, дані яких таблиць і яких полів оброблятиме ця форма, які графічні елементи мають бути до неї додані. У програмі є такі інструменти керування: поле, напис, група, перемикач, позначка, вимикач, поле зі списком тощо. Елементи керування форми можуть бути прив'язаними, вільними або такими, що визначаються.

Прив'язаний елемент керування приєднаний до поля таблиці або запиту і застосовується для відображення, введення або оновлення значень у полі таблиці. *Вільний елемент керування* не прив'язаний до певного поля таблиці і слугує для виведення на екран цифрової, текстової або графічної інформації. *Елемент керування, що визначається,* використовують для здійснення розрахунків.

Форма в базі даних — це спеціальне структуроване вікно для введення й відображення інформації. Форми створюють на основі таблиць або запитів із набору окремих елементів керування: текстових полів для введення і редагування даних, підписів полів, кнопок, списків, перемикачів, а також рамок об'єктів для відображення графіки та об'єктів OLE.

В Access можна створити форми таких видів:

- форму в стовпець або повноекранну форму використовують для введення й редагування даних;
- стрічкова форма слугує для відображення полів групи записів;
- таблична форма відображає дані в режимі таблиці;
- форма головна / підлегла являє собою сукупність форми в стовпець і табличної;
- зведена таблиця виконується Майстром створення зведених таблиць Excel на основі таблиць і запитів Access;
- форма-діаграма будує діаграму, створену Microsoft Graph. Graph є впроваджуваним OLE-додатком і може бути запущений з Access.

Для створення форми у вікні бази даних треба вибрати вкладку **Створення** і в групі *Форми* активізувати одну із команд (рис. 14.24). Щоб швидко створити форму — вибрати таблицю, наприклад *Перелік товарів*, і натиснути на кнопку *Форма* (рис. 14.25).

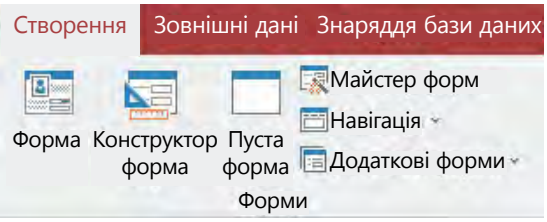


Рис. 14.24

Для самостійного створення форми використовують *Конструктор форм* або *Пусту форму* (див. рис. 14.24).

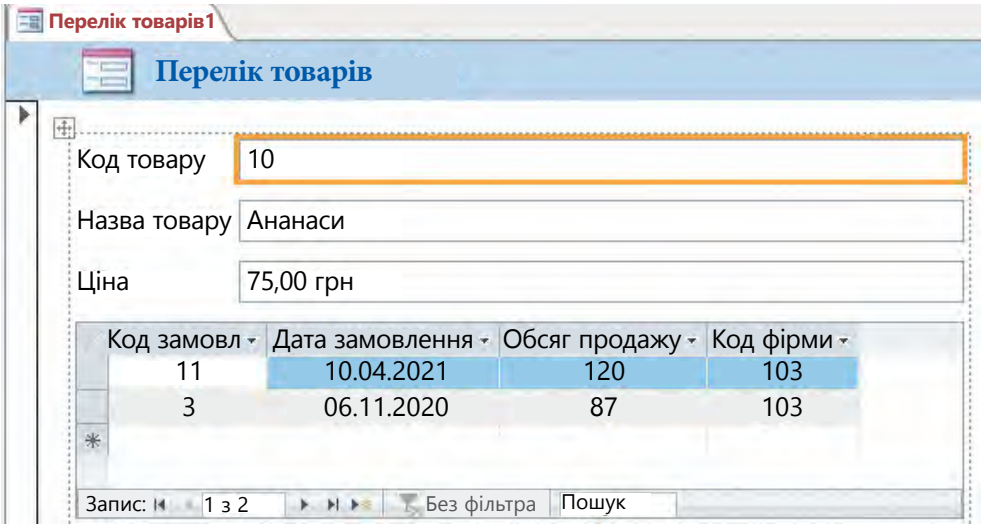


Рис. 14.25

Щоб створити форму за допомогою *Майстра форм,* на вкладці *Створення* у групі *Форми* потрібно натиснути кнопку Майстер форм. У вікні вибрати в *Таблиці та запити* таблицю *Фірми*, а в списку *Доступні поля* — поля для форми. За допомогою кнопки з подвійною стрілкою перенести всі поля до списку *Вибрані поля* і натиснути кнопку *Далі*. В наступному вікні вибрати один із варіантів макета: стовпці, таблиця, таблиця даних, за шириною та встановити перемикач у положення *Стовпці* й натиснути кнопку *Далі*. В останньому вікні майстра ввести заголовок форми *Фірми* й натиснути кнопку *Готово* (рис. 14.26).

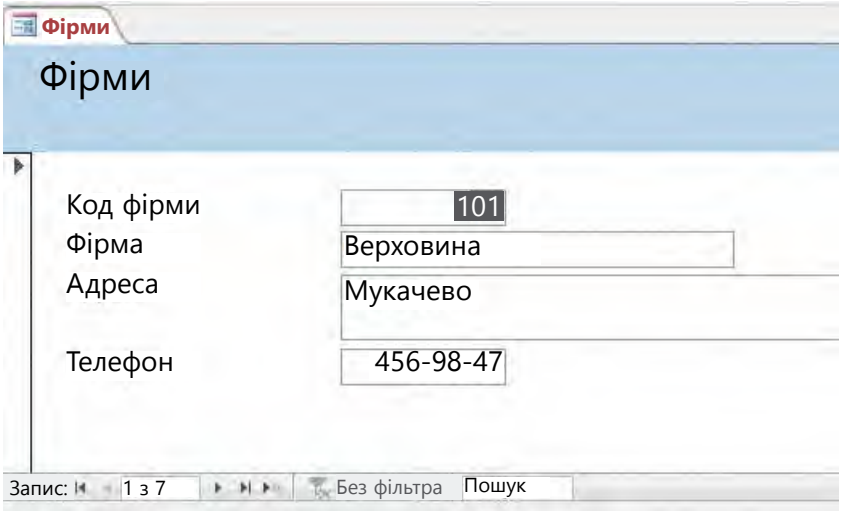


Рис. 14.26

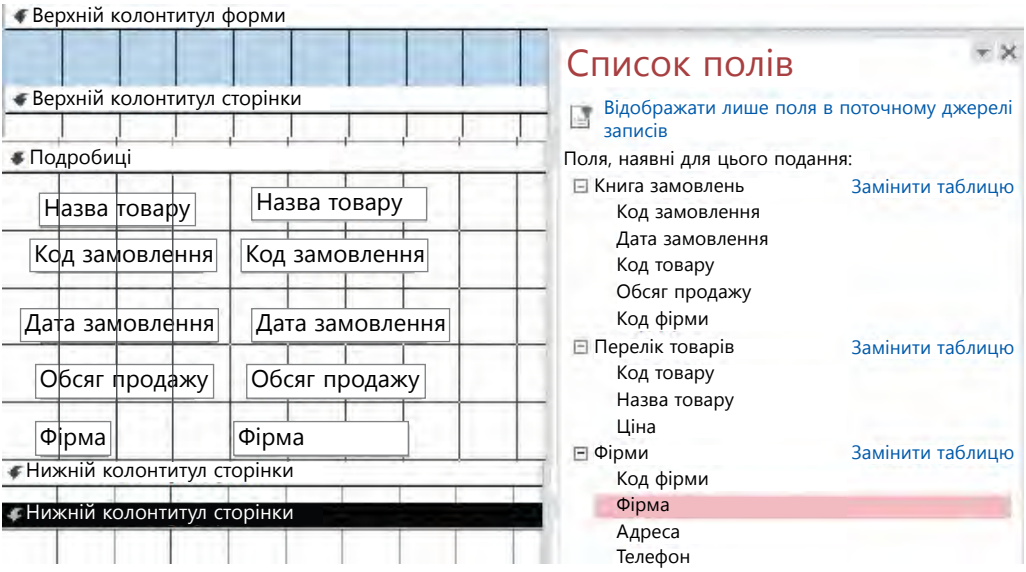


Рис. 14.27

Режим *Конструктор форм* використовують для удосконалення зовнішнього вигляду форми, створеної за допомогою *Майстра форм* або ж для створення форми «з нуля». Треба скористатися командою *Конструктор форм* на вкладці **Створення** у групі *Форми*. Робочим полем *Конструктора* є форма, яка складається з таких областей: *Верхній колонтитул форми* (у деяких версіях *Заголовок форми*), *Верхній колонтитул сторінки, Подробиці* (у деяких версіях *Область даних*), *Нижній колонтитул сторінки, Нижній колонтитул форми* (у деяких версіях *Примітка форми*) (рис. 14.27).

Щоб додати поля до форми, потрібно виконати команду *Додавання наявних полів* у групі *Знаряддя* на вкладці *Знаряддя конструктора форм*. Ліворуч від форми з'явиться вікно *Список полів,* у якому відображаються створені таблиці з наявними полями. Далі вибрати поле лівою кнопкою миші на імені поля в *Списку полів* і перетягти його на форму. Наприклад, для створення форми *Відомості про замовлений товар* треба перенести на форму поля таблиці *Обсяг продажу, Перелік товарів, Фірма.*

На контекстній вкладці *Знаряддя конструктора форм* містяться інструменти форматування тексту — вкладка *Формат*, зміна теми, додавання елементів керування, емблеми, назви — вкладка *Конструктор,* переміщення та розташування — вкладка *Упорядкування*. Після усіх налаштувань треба назвати форму, наприклад *Відомості про замовлений товар,* і закрити (рис. 14.28).

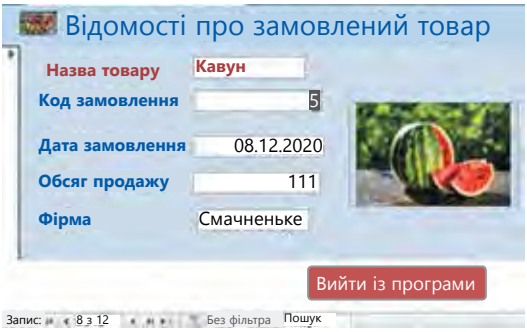


Рис. 14.28

14.8. СТВОРЕННЯ ЗВІТІВ

Звіт зазвичай складають за певний період діяльності в певній предметній галузі. Звіт, створений за допомогою системи Microsoft Access, забезпечує подання даних у різних форматах, із різним ступенем деталізації. Для більшої наочності до звіту додають графічні об'єкти (рамки, рисунки, графіки, діаграми тощо).

Звіт — це об'єкт бази даних, який використовують для відображення, узагальнення даних. Звіт являє собою спеціальну форму, здебільшого призначену для виведення на принтер.

Для створення простого табличного звіту потрібно виділити, наприклад, таблицю *Перелік товарів* і вибрати команду *Звіт* у групі *Звіти* на вкладці *Створення* (рис. 14.29).

Команда *Пустий звіт* відкриває порожній звіт у режимі розмітки і список полів. У *Список полів* треба натиснути на знак плюс поруч із таблицею або таблицями з полями, які потрібно відобразити у звіті, й перетягнути поля

до звіту. За допомогою засобів із групи *Елементи керування* на вкладці **Формат** можна додати до звіту емблему, назву, номери сторінок, дату й час.

Конструктор звітів відкриває порожній звіт у режимі конструктора і дає змогу додати до нього потрібні поля й елементи керування.

Етикетки — майстер, у якому можна вибрати стандартний розмір підписів, а також зазначити, які поля потрібно відображати та як їх слід сортувати. Відтак майстер створить звіт із підписами на базі вибраних параметрів.

Щоб створити звіт у *Майстрі звітів,* у групі *Звіти* треба перейти на вкладку *Створення* і натиснути кнопку *Майстер звітів*. У вікні, що відкрилося, зі списку джерел *Таблиці і запит,* вибрати *Запит: Вартість товару* та перенести всі поля обраного запиту до списку *Вибрані поля* за допомогою кнопки переносу полів >>. Натиснути кнопку *Далі* й у вікні додати рівні групування, наприклад, *Назва товару*. Натиснути кнопку *Далі*. У наступному вікні визначити спосіб сортування даних у звіті. Задати сортування за полем *Вартість*. Натиснути кнопку *Далі*. У цьому вікні вибрати вид макета (східчастий, блок, структура) та орієнтацію (книжкову або альбомну). Натиснути кнопку *Далі*. В наступному вікні майстра потрібно ввести заголовок звіту (наприклад, *Вартість товару*) і натиснути кнопку *Готово*. В режимі *Попередній перегляд* звіт відображається в тому вигляді, у якому його буде виведено на друк (рис. 14.30).

Перелік товарів

Код товару	Назва товару	Ціна	Фото
10	Ананаси	75,00 грн	
20	Банани	29,00 грн	
30	Виноград	58,00 грн	
60	Диня	63,00 грн	
70	Хурма	57,00 грн	
50	Кавун	25,00 грн	
40	Ківі	45,00 грн	
		325,00 грн	

Сторінка 1 з 1

Рис. 14.29

Вартість товару

Назва товару	Вартість	Ціна	Обсяг продажу
Ананаси	6 525,00 грн	75,00 грн	87
Банани	9 000,00 грн	75,00 грн	120
	2 175,00 грн	29,00 грн	75
	4 263,00 грн	29,00 грн	147
Виноград	6 090,00 грн	58,00 грн	105
Диня			
Кавун	6 804,00 грн	63,00 грн	108
	2 775,00 грн	25,00 грн	111
Ківі			
	7 020,00 грн	45,00 грн	156
	7 785,00 грн	45,00 грн	173
Хурма	5 415,00 грн	57,00 грн	95
	7 410,00 грн	57,00 грн	130
	11 457,00 грн	57,00 грн	201

Сторінка 1 з 1

Рис. 14.30

ТЕСТОВІ ЗАВДАННЯ

ЩО ТАКЕ БАЗА ДАНИХ?

1. Особливі файли, використання яких разом зі спеціальними програмними засобами дає змогу переглядати необхідну інформацію
2. Набір інформації, організованої тим чи тим способом
3. Вся наявна інформація
4. Особливі файли, використання яких разом зі спеціальними програмними засобами дає змогу переглядати необхідну інформацію, а також маніпулювати нею

ЯКИЙ РЕЖИМ БАЗИ ДАНИХ Access ПРИЗНАЧЕНИЙ ДЛЯ СТВОРЕННЯ ТА ВНЕСЕННЯ ЗМІН ДО СТРУКТУРИ БАЗИ ДАНИХ, РОЗРОБЛЕННЯ ФОРМ, ЗАПИТІВ, ЗВІТІВ?

1. Режим конструктора
2. Режим перегляду
3. Початковий режим
4. Режим виконання

ЯКІ ОБ'ЄКТИ БАЗИ ДАНИХ Access Є КІНЦЕВОЮ МЕТОЮ БУДЬ-ЯКОГО ОБЛІКУ ДАНИХ?

1. Звіти
2. Таблиці
3. Запити
4. Форми

У ЧОМУ ПОЛЯГАЄ ОСОБЛИВІСТЬ ПОЛЯ «ЛІЧИЛЬНИК»?

1. Слугує для введення числових даних
2. Слугує для введення дійсних чисел
3. Має обмежений розмір
4. Має властивість автоматичного нарощування

У ЯКОМУ ДІАЛОГОВОМУ ВІКНІ СТВОРЮЮТЬ ЗВ'ЯЗКИ МІЖ ПОЛЯМИ ТАБЛИЦЬ БАЗИ ДАНИХ?

1. Таблиця зв'язків
2. Схема зв'язків
3. Зв'язки
4. Таблиця даних

ЩО ТАКЕ ЗАПИТИ В MS Access?

1. Вікно діалогу або документ в електронному варіанті
2. Засіб відбору даних з однієї або декількох таблиць за допомогою визначеної користувачем умови
3. Засіб створення складних документів і виведення їх на друк
4. Пошук потрібного файлу

ПОШУК ПОТРІБНОЇ ІНФОРМАЦІЇ У БАЗАХ ДАНИХ ЗДІЙСНЮЄТЬСЯ НА ОСНОВІ:

1. звітів
2. запитів
3. форм
4. макросів

Розділ 15
СТВОРЕННЯ ПРЕЗЕНТАЦІЙ. ПРОГРАМА Microsoft PowerPoint

15.1. ОСНОВНІ ПОНЯТТЯ ПРЕЗЕНТАЦІЇ

Презентації використовують для подання (демонстрації) нових проєктів, товарів, послуг, звіту роботи підприємства, доповідей, рекламних повідомлень тощо.

Презентація (англ. presentation — «показ») — це демонстрування певного матеріалу.

Комп'ютерна презентація — це електронний документ (файл) рекламного або інформаційного характеру. Для демонстрації комп'ютерних презентацій використовують комп'ютери, мультимедійні проєктори, аудіообладнання, екрани, мультимедійні та інтерактивні дошки тощо.

Слайд — це окрема екранна сторінка, на якій розміщують текстову, графічну інформацію, відео, анімацію, звукові об'єкти та гіперпосилання.

Мультимедіа — це поєднання різних форм подання інформації: текстової, графічної, звукової, відео тощо.

Залежно від способу реалізації на комп'ютері розрізняють такі види презентацій:

- зі сценарієм;
- інтерактивні;
- автоматичні.

Презентація зі сценарієм — це традиційна презентація зі слайдами, доповнена засобами показу кольорової графіки й анімації. Під час показу в ній є можливість вносити зміни в процес демонстрації.

Інтерактивна презентація — це діалог користувача з комп'ютером. Користувач вирішує, який матеріал для нього важливий, і вибирає на екрані потрібний об'єкт за допомогою миші або натискання клавіші. Всі інтерактивні презентаційні програми керують подіями. Це означає, що коли відбувається певна подія (натискання клавіші, позиціювання курсора на екранний об'єкт тощо), програма виконує відповідну дію. Інтерактивна презентація дає змогу шукати потрібну інформацію, заглиблюючись у неї настільки, наскільки це було передбачено розробником презентації.

Автоматична презентація — це закінчений інформаційний продукт. Його можна перенести на носій інформації і розіслати потенційним споживачам, щоб дістати уявлення про їхню зацікавленість.

Залежно від *сфери застосування* розрізняють такі типи презентацій:

- торгові;
- маркетингові;
- навчальні;
- корпоративні.

Торгові презентації дають змогу розповісти про ключові характеристики, властивості та переваги товару, послуги тощо, тобто за короткий термін надати потенційному покупцеві всю необхідну інформацію.

Маркетингові презентації використовують під час підготовки умов для майбутніх торгових презентацій. Вони призначені для масової аудиторії споживачів (проводяться на виставках-ярмарках або в офісі покупця), для агентів з продажу тощо.

Навчальні презентації допомагають зручно й наочно подати навчальний матеріал.

Корпоративні презентації використовують із метою донесення інформації до акціонерів корпорації, зокрема за допомогою гіпертекстової гіпермедійної системи Word Wide Web (WWW). Гіпертекстові документи (щорічні звіти, проспекти тощо) містять текстові посилання на інші документи, розміщені на WWW-серверах у всьому світі.

За *технологіями створення* розрізняють три види презентацій:

- слайдові;
- потокові;
- 3D-презентації.

Слайдові презентації — набір слайдів до певної теми, що використовують здебільшого для супроводу доповіді. Користувач може керувати процесом демонстрування слайдів, показувати їх протягом різних проміжків часу, переходити від одного слайда до іншого в довільному порядку або призупиняти їх показ.

Потокова презентація — це звичайний відеофільм або відеокліп, який неперервно демонструють на екрані засобами передавання потокових даних з Інтернету, носія інформації тощо. Технічним засобом для її показу може бути комп'ютер, відеоплеєр та ін.

3D-презентації створюють за допомогою панорамної відеотехніки і 3D-моделювання.

15.2. ЕТАПИ СТВОРЕННЯ ПРЕЗЕНТАЦІЇ ТА ВИМОГИ ДО ЇЇ ОФОРМЛЕННЯ

Процес розроблення презентації охоплює такі етапи:

- визначення теми;
- підбір відповідного матеріалу з теми (текст, графічні об'єкти, анімація, відео, звук);
- розподіл матеріалу за слайдами;
- вибір розмітки слайдів для розміщення матеріалів;
- вибір оформлення слайдів у середовищі програми для створення презентацій;
- додавання ефектів анімації;
- перегляд презентації;

- виправлення виявлених помилок і недоліків;
- збереження та публікація презентації.

Певні вимоги передбачено *до структури та змісту* презентацій. Матеріал потрібно викладати стисло (12–18 слайдів), структуровано, максимально інформативно. Заголовки мають бути короткими й лаконічними, привертати увагу аудиторії та узагальнювати основні положення слайда. Важливу інформацію (наприклад, висновки, визначення, правила тощо) треба подавати великим і виділеним шрифтом у лівому верхньому кутку слайда. Другорядну інформацію бажано розміщувати внизу слайда. Усі слайди презентації мають бути витримані в одному стилі. Проте, якщо потрібно привернути увагу до окремих ідей, проблем, для певних слайдів можна застосувати особливе оформлення (інший колір, графічні елементи, анімацію).

Для наочного подання матеріалу використовують графічні об'єкти, таблиці, діаграми, схеми маркованих і нумерованих списків. Підписи треба розміщувати під ілюстраціями. Текст має складатися з коротких слів і простих речень, без орфографічних, граматичних і стилістичних помилок.

Під час створення презентацій потрібно зважати на *фізіологічні особливості сприйняття кольорів і форм:*

- теплі кольори (червоний, оранжевий, жовтий) сприяють збудженню і діють як подразники;
- холодні кольори (фіолетовий, синій, блакитний, синьо-зелений, зелений) заспокоюють, викликають сонливий стан;
- нейтральними є світло-рожевий, жовто-зелений, коричневий кольори;
- деякі поєднання кольорів не тільки стомлюють зір, а й можуть спричинити стрес (наприклад, зелені символи на червоному фоні);
- найкращі поєднання кольорів шрифту й фону: білий на темно-синьому, чорний на білому, жовтий на синьому;
- кольорова схема має бути однаковою для всіх слайдів;
- будь-який малюнок фону підвищує стомлюваність очей і знижує ефективність сприйняття інформації;
- чіткі, яскраві малюнки, які швидко змінюються, краще запам'ятовуються;
- будь-який рухомий (анімований) другорядний об'єкт знижує якість сприйняття матеріалу, відвертає увагу.

15.3. ОСНОВНІ ЕЛЕМЕНТИ PowerPoint

Прикладні програми, призначені для створення комп'ютерних презентацій, — це *системи оброблення презентації,* або *редактори презентацій* (Microsoft Office PowerPoint, OpenOffice.org Impress, Powerbullet Presenter, ProShow Producer, PPT CREATE, Quick Slide Show, MySlideShow, Prezi, Canva, Keynote, Piktochart, Haiku Deck, Office Sway, Google презентації, Seidat, Adobe Flash, Microsoft Movie Maker, Macromedia Flash, AnFX Visual Design, Virtual Tour Builder та ін.).

Програма *Microsoft Office PowerPoint* входить до складу пакета MS Office.

Завантаження програми PowerPoint здійснюють стандартними способами:

- на панелі завдань операційної системи активізувати *Пуск* та виконати команду *Всі програми — Microsoft Office — Microsoft PowerPoint* або просто *Пуск — Microsoft PowerPoint;*
- якщо на робочому столі є ярлик PowerPoint, то для запуску програми достатньо двічі клацнути мишею на кнопці ;
- відкрити створену презентацію — це також запустить PowerPoint, а презентація буде доступною для редагування.

Файли презентацій мають розширення **.ppt** і **.pptx.**

Після завантаження програми на екрані монітора з'являється вікно PowerPoint (рис. 15.1):

1. Панелі швидкого доступу до функцій, які часто використовують: *Зберегти, Скасувати, Повторити* і *Настроїти панель швидкого доступу.*
2. Рядок заголовка, що містить *назви поточного документа* та *програми.*
3. Стандартні кнопки *Згорнути, Поновити / Розгорнути, Закрити.*
4. Рядок з іменами вкладок: *Файл, Основне, Вставлення, Конструктор, Переходи, Анімація, Показ слайдів, Рецензування, Подання, Записування, Довідка.*
5. Стрічка.
6. Слайд — для відображення у вигляді ескізів усіх слайдів презентації.
7. Поле редагування слайда — основна частина вікна програми, в якій здійснюють усі операції з презентацією.
8. Лінійки.

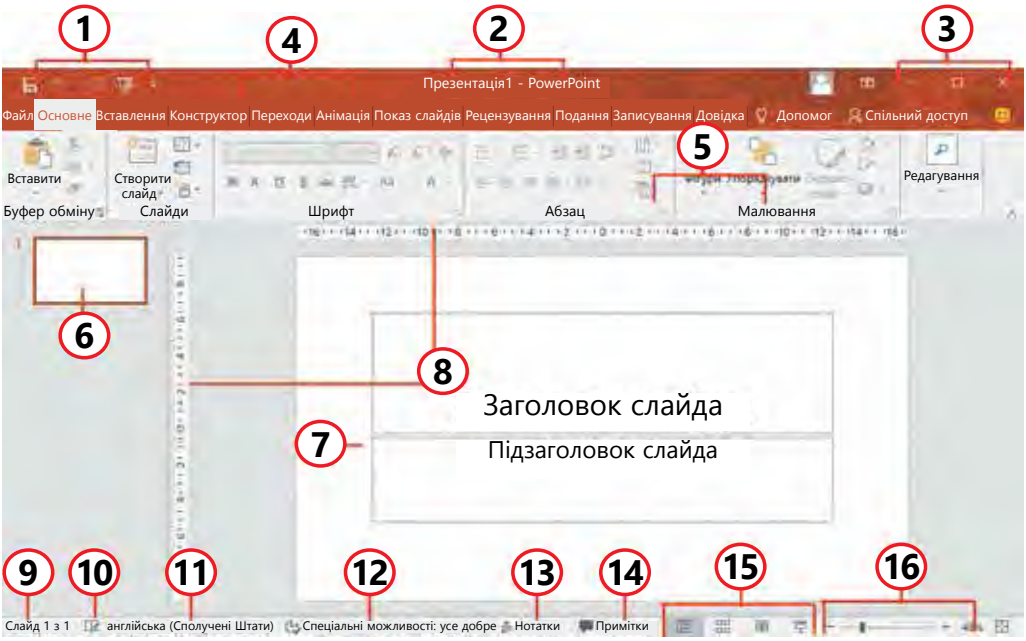


Рис. 15.1

9. Інформація про поточний слайд, номер слайда, кількість слайдів.
10. Словник.
11. Мова.
12. Спеціальні можливості.
13. Нотатки до слайда.
14. Примітки.
15. Кнопки режимів перегляду слайдів: *Звичайний* використовують під час створення, редагування та форматування слайдів презентації; *Сортувальник слайдів* — для визначення порядку на екран виводять ескізи слайдів; *Подання читання* — подання для проведення презентації; *Показ слайдів* — демонстрація презентації, починаючи з поточного слайда (клавіша *F5*).
16. Панель для зміни масштабу зображення.

Розглянемо основні функції вкладок PowerPoint:

- **Файл** — стандартний набір команд: *Відомості* про файли презентацій, *Зберегти* або *Зберегти як, Закрити, Відкрити, Створити, Друк, Спільний доступ, Експорт, Обліковий запис, Відгук* тощо;
- **Основне** — елементи створення та форматування об'єктів: слайдів, розділів, тексту, написів, зображень, ліній, різних геометричних фігур та ін.;
- **Вставлення** — елементи керування для вставлення об'єктів;
- **Конструктор** — елементи для розроблення загальної концепції презентації: підбір стилю, схем кольорів для слайдів, теми, вибір розміру слайдів під час підготовки презентації до друку, налаштовування відступів від кожного краю за допомогою команди *Параметри сторінки,* зміни розташування слайда на аркуші за допомогою команди *Орієнтація слайдів;*
- **Переходи** — налаштування ефектів переходу між слайдами;
- **Анімація** — інструменти, які дають змогу зробити матеріал більш живим і насиченим;
- **Показ слайдів** — елементи для перегляду та підготовки до показу готової презентації в повноекранному режимі;
- **Рецензування** — елементи керування, призначені для перевірки орфографії, перекладу тексту, створення приміток тощо;
- **Подання** — команди, за допомогою яких можна змінювати режими перегляду презентації, налаштовувати зразки, відображення вертикальної і горизонтальної лінійок, сітки, масштаб, встановлювати кольорову гаму слайдів;
- **Записування** — команди для запису екрана, автовідтворення мультимедіа, збереження та експорту презентації;
- **Розробник** — прихована вкладка, що містить елементи для роботи з графікою, аудіо- та відеоматеріалом, налаштування додаткових елементів керування. Щоб її активувати, потрібно відкрити на стрічці вкладку *Файл* і вибрати команду *Параметри*. У діалоговому вікні перейти в розділ *Налаштування стрічки*. У правій області встановити прапорець *Розробник* і клацнути на кнопці ОК. Праворуч від вкладки *Записування* у стрічці з'явиться ця вкладка.

Щоб **створити презентацію**, потрібно відкрити на стрічці вкладку **Файл** і вибрати команду *Створити* — *Нову презентацію* або один із шаблонів (наприклад, *Галерея*) (рис. 15.2).

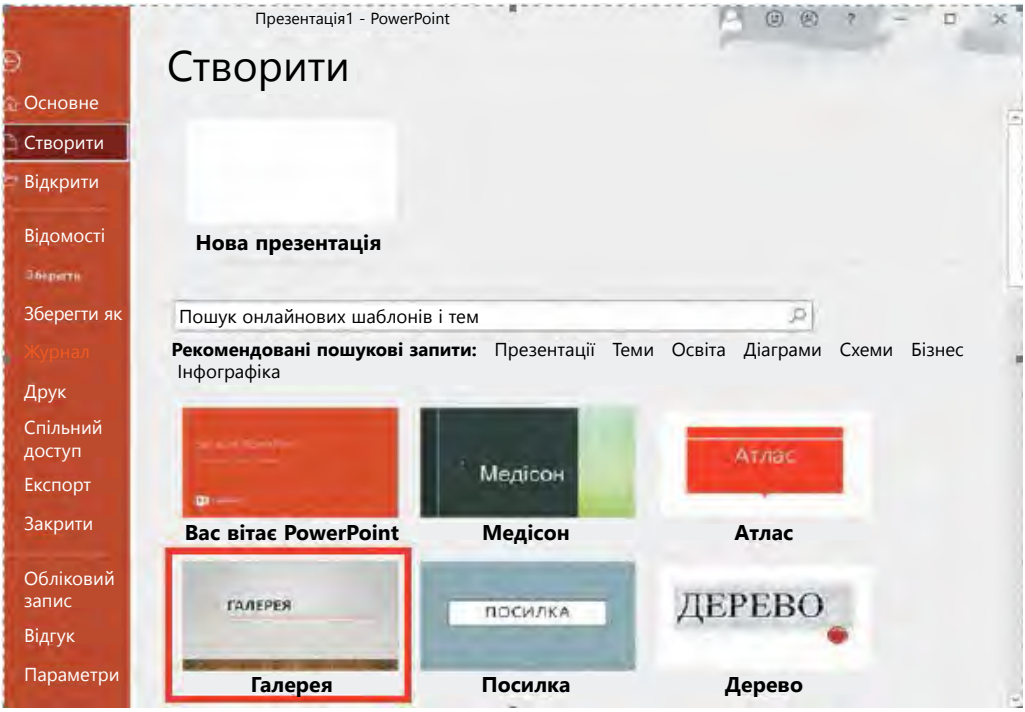


Рис. 15.2

Далі вибрати макет шаблону *Галерея* і натиснути на кнопку *Створити* (рис. 15.3).

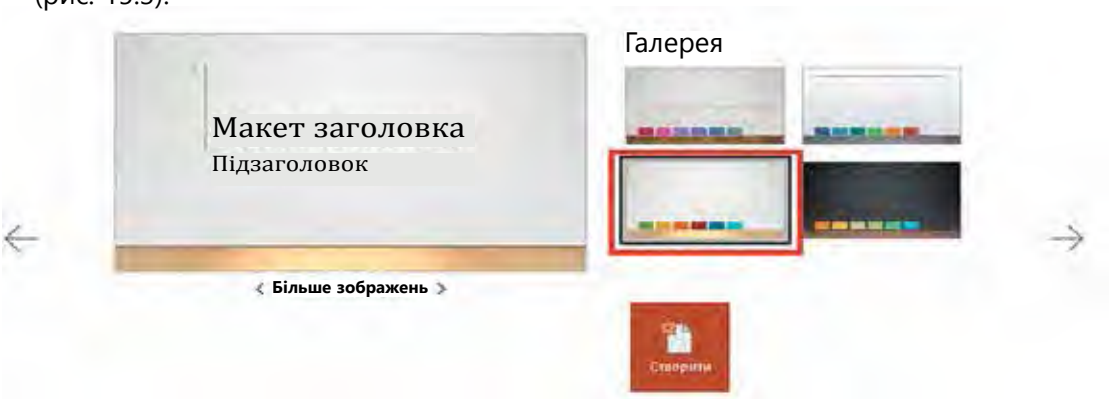


Рис. 15.3

Наступний крок — у текстовому полі *Заголовок слайда* ввести текст (рис. 15.4).

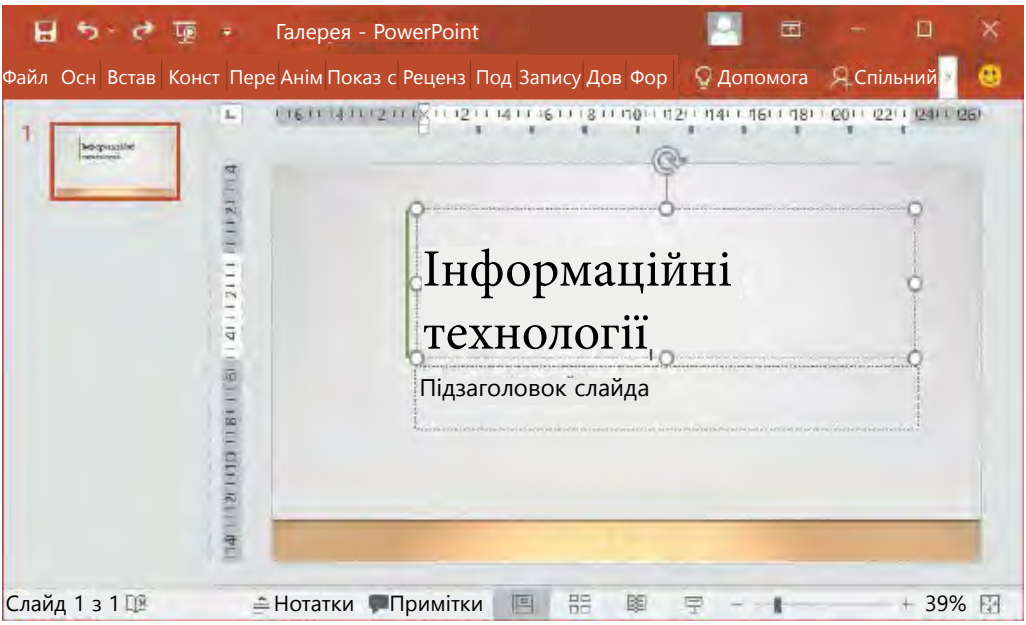


Рис. 15.4

15.4. РОБОТА ЗІ СЛАЙДАМИ ПРЕЗЕНТАЦІЇ

Щоб **додати новий слайд**, необхідно вибрати слайд, за яким потрібно вставити новий, і виконати такі дії:

- відкрити вкладку **Основне** — *Створити слайд* і вибрати макет (рис. 15.5);
- відкрити вкладку **Вставлення** — *Новий слайд* і вибрати макет;
- натиснути клавішу *Enter*;
- натиснути комбінацію клавіш *Ctrl+M;*
- у контекстному меню вибрати команду *Новий слайд.*

Щоб **продублювати слайд**, необхідно виділити відповідний ескіз слайда й виконати такі дії:

- відкрити вкладку **Основне** — *Створити слайд — Дублювати вибрані слайди* (рис. 15.5);
- відкрити вкладку **Вставлення** — *Новий слайд — Дублювати вибрані слайди;*
- у контекстному меню вибрати команду *Дублювати слайд*.

Копія слайда з'явиться одразу під оригіналом.

Якщо потрібно перемістити слайд, лівою кнопкою миші його перетягують у потрібне місце. Для переміщення декількох слайдів — натискаючи на клавішу *Ctrl*, вибрати слайди, які потрібно перемістити. Далі відпустити клавішу *Ctrl*, а вибрані слайди перетягнути до нового розташування як єдину групу.

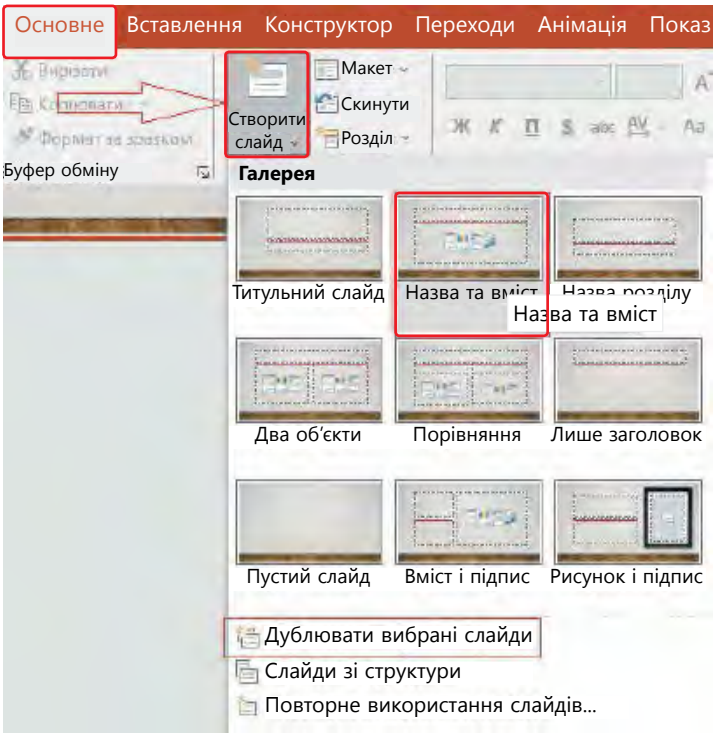


Рис. 15.5

Щоб **вилучити слайди**, потрібно виконати один із таких варіантів:

- виділити слайд і в контекстному меню вибрати команду *Видалити слайд;*
- виділити слайд і натиснути на клавішу *Delete;*
- якщо потрібно вилучити кілька слайдів — натискаючи клавішу *Ctrl*, вибрати слайди для видалення, відпустити клавішу *Ctrl* і натиснути на клавішу *Delete*.

Слайд, який не має відображатися під час показу слайдів, можна **приховати**:

- відкрити вкладку **Показ слайдів** — *Приховати слайд;*
- виділити слайд і в контекстному меню вибрати команду *Приховати слайд*.

Для **форматування тексту** використовують: *спливну панель,* вкладку **Основне** або комбінацію клавіш (рис. 15.6).

Форматування шрифту передбачає його накреслення (жирний *(Ctrl+B),* курсив *(Ctrl+I),* підкреслення *(Ctrl+U),* гарнітуру, розмір, колір, ефекти, міжсимвольний інтервал).

Форматування абзацу — це вирівнювання (за лівим краєм *(Ctrl+L)*, по центру *(Ctrl+E),* за правим краєм *(Ctrl+R),* за шириною *(Ctrl+J)*), налаштування інтервалів перед абзацом і після абзацу, міжрядкового інтервалу, відступів, напрямку тексту.

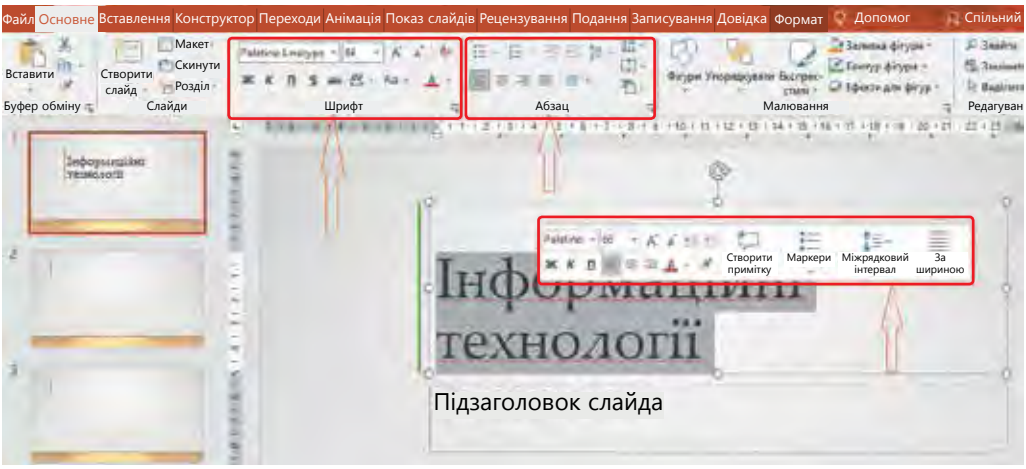


Рис. 15.6

Для створення списку потрібно виділити текст і натиснути на кнопку *Маркери* або *Нумерація* (рис. 15.7).

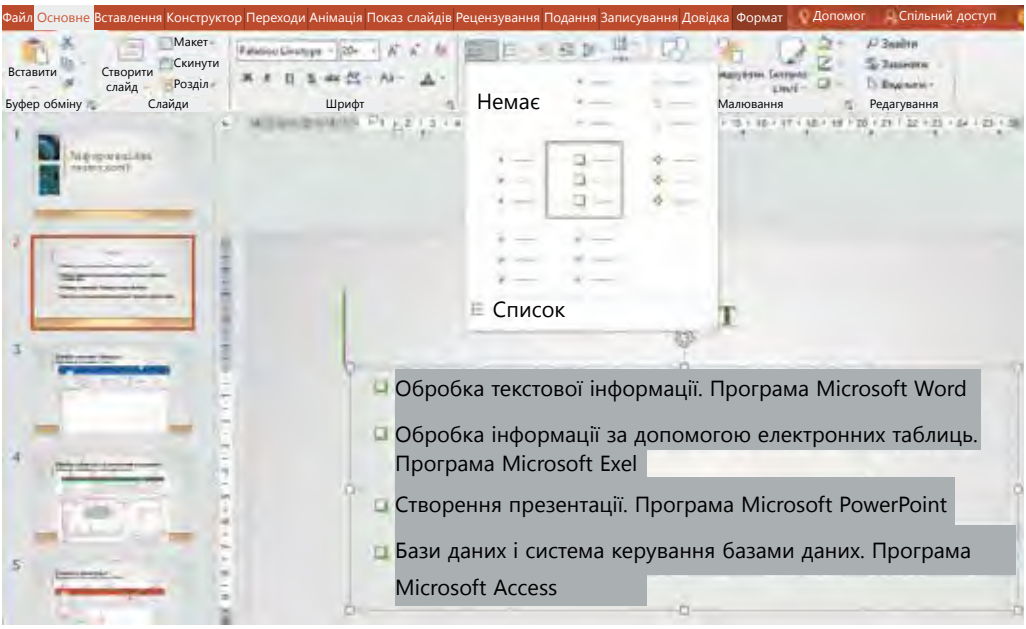


Рис. 15.7

Графічні об'єкти (зображення, фігури або діаграми) додають на вкладці **Вставлення** із груп *Зображення* та *Ілюстрації* (рис. 15.8).

Команда *Зображення — Цей пристрій...* відкриває вікно *Вставлення рисунка,* у якому вибирають файл із потрібним рисунком із комп'ютера. Команда *Зображення — З мережі Інтернет...* відкриває вікно *Вставлення зображень* для пошуку і вставлення зображення із різних джерел мережі Інтернет.

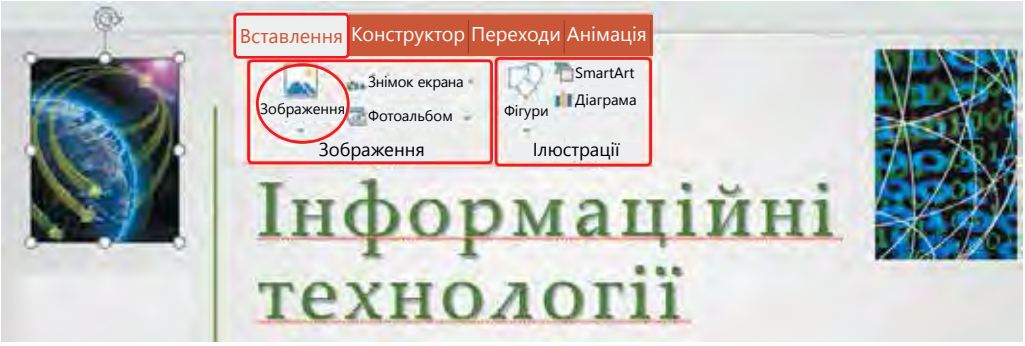


Рис. 15.8

Команда *Фігури* в групі *Ілюстрації* на вкладці *Вставлення: Фігури, SmartArt* або *Діаграма* виводить на екран фігури (рис. 15.9).

Рис. 15.9

Зображення, фігури, графічні об'єкти або діаграму можна редагувати, тобто змінювати їхні розмір, колір, заливку, тип лінії тощо. Для цього потрібно двічі клацнути лівою кнопкою миші на створеній фігурі й вибрати вкладку **Формат** або *Формат фігури* (рис. 15.10).

До **нотаток доповідача** можна додати цікаві та корисні факти й звертатися до них під час презентації (кнопка *Нотатки* у рядку стану) (рис. 15.11).

Із фрагментами тексту, графічними зображеннями, розміщеними на слайдах, можна пов'язувати гіперпосилання: виділити об'єкт і виконати *Вставлення — Посилання (Ctrl+K)*. Вибрати у списку вікна *Додавання гіперпосилання Зв'язати з* (рис 15.12) тип об'єкта, на який указуватиме посилання (файл, веб-сторінка, місце в документі, новий документ, електронна пошта) або ввести повне ім'я об'єкта в поле *Адреса* й натиснути кнопку *OK*.

Особливістю комп'ютерної презентації є можливість **додавання анімаційних ефектів** до тексту й графічних об'єктів, що розміщені на слайдах. Це забезпечує кращу наочність і динамічність презентації. Для створення анімаційного ефекту потрібно виділити об'єкт або текст, перейти на вкладку *Анімація* й вибрати параметри відповідного ефекту (рис. 15.13).

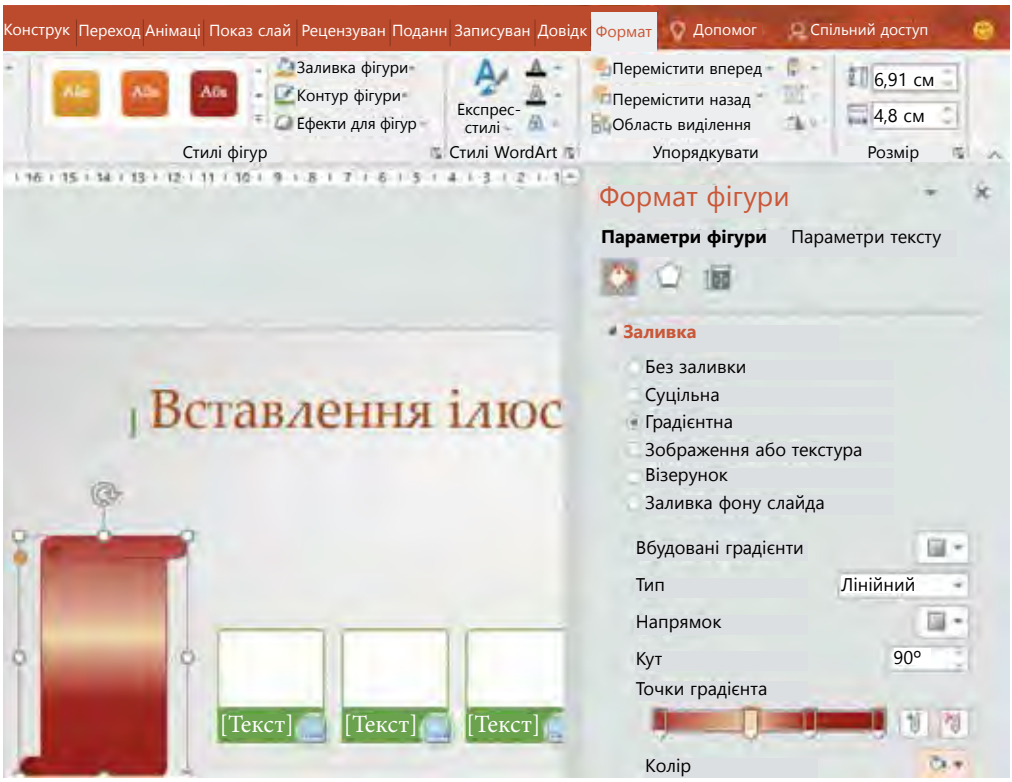


Рис. 15.10

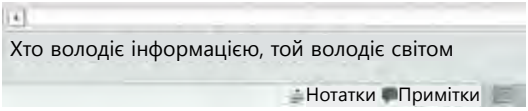


Рис. 15.11

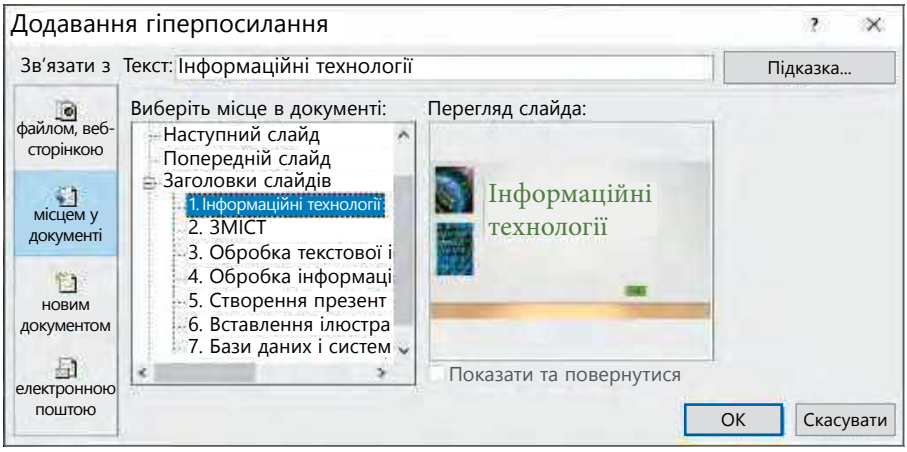


Рис. 15.12

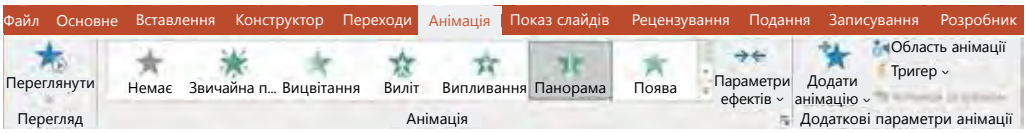


Рис. 15.13

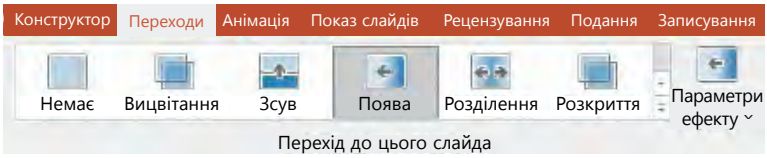


Рис. 15.14

Щоб додати анімаційний ефект до переходу з одного слайда до наступного під час презентації на вкладці *Переходи* необхідно вибрати перехід і натиснути на кнопку *Параметри ефектів* (рис. 15.14): *Справа, Зліва, ...* . Для скасування переходу — вибрати елемент *Немає*.

Для **демонстрації** презентації використовують вкладку *Показ слайдів*. Щоб запустити презентацію з першого слайда, у групі *Початок показу слайдів* треба вибрати пункт *З початку* (рис. 15.15). Якщо відображається не перший слайд і потрібно почати саме з нього — вибрати пункт *З поточного слайда*.

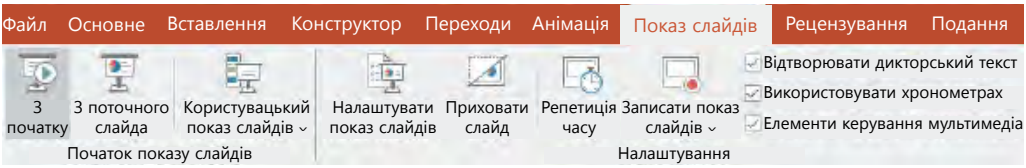


Рис. 15.15

Для створення показу слайдів, який автоматично запускатиметься після відкриття файлу презентації, потрібно виконати такі дії:

- вибрати **Файл** — *Зберегти як;*
- вибрати теку, у якій потрібно зберегти презентацію;
- у полі *Ім'я файлу* ввести ім'я презентації;
- у розділі *Тип файлу* вибрати *Демонстрація PowerPoint*.

ТЕСТОВІ ЗАВДАННЯ

ЧИ МОЖНА В ПРОГРАМІ PowerPoint ЗМІНЮВАТИ ФОН КОЖНОГО ОКРЕМОГО СЛАЙДУ?

1. Ні, не можна
2. Так, можна
3. Тільки для окремих слайдів
4. Тільки для всіх слайдів

У Microsoft PowerPoint ФАЙЛИ МАЮТЬ РОЗШИРЕННЯ:

1. .txt
2. .pptx
3. .pcx
4. .ppc

Розділ 16
СИСТЕМА ОБРОБЛЕННЯ МАТЕМАТИЧНОЇ ІНФОРМАЦІЇ MathCAD

16.1. ОСОБЛИВОСТІ ОБРОБЛЕННЯ МАТЕМАТИЧНОЇ ІНФОРМАЦІЇ

Математичні обчислення і розрахунки завжди супроводжували технічний прогрес. Із розвитком новітніх технологій, створенням і впровадженням нових машин та пристроїв роль математичного оброблення інформації неухильно зростає. Нагадаємо, що перші комп'ютери були призначені саме для виконання складних математичних обчислень і тому мали назву «електронно-обчислювальні машини». І нині оброблення математичної інформації, математичні обчислення становлять вагому частину інформаційних технологій.

Для виконання складних розрахунків і обчислень розроблено низку пакетів спеціальних програм.

Однією з найпотужніших математичних програм є система *MatLab*. За її допомогою можна виконувати складні розрахунки в таких галузях, як оброблення сигналів, статистика, дискретна математика, економіка тощо. Для кожної прикладної галузі у MatLab розроблено набір спеціальних функцій і інструментів. Наприклад, програма Simulink дає змогу моделювати функціонування реальних пристроїв і систем.

Для здійснення математичних операцій у символьному вигляді призначена система *Maple*. За допомогою цієї системи можна виконувати складні перетворення математичних виразів, спрощувати їх, розв'язувати рівняння і системи рівнянь. Символьний процесор Maple використовують в інших математичних системах оброблення, зокрема в *MatLab* і *MathCAD*.

Для складних математичних обчислень застосовують систему *Mathematica*, що виконує також символьні перетворення, працює з текстовою і графічною інформацією.

Розглянемо докладніше можливості системи оброблення математичної інформації *MathCAD*.

16.2. ЗАГАЛЬНІ ВІДОМОСТІ ПРО СИСТЕМУ MathCAD

Система MathCAD (англ. Mathematical Computer Aided Design — математична система автоматичного проєктування) призначена для виконання складних математичних обчислень, розрахунків і перетворень.

MathCAD має такі функції:

- подання величин цілими, раціональними, ірраціональними, комплексними числами;

- подання величин логічними змінними;
- подання чисел у десятковій, двійковій, вісімковій, шістнадцятковій системі числення;
- подання чисел у формі з фіксованою і рухомою точками;
- виконання арифметичних операцій із цілими, раціональними, ірраціональними, комплексними числами;
- можливість задавати точність обчислень і точність подання результатів обчислень;
- оброблення інформації за допомогою великої кількості вбудованих функцій;
- можливість створювати функції користувача;
- розв'язок лінійних і нелінійних рівнянь та систем рівнянь;
- виконання диференціювання, інтегрування та інших операцій вищої математики;
- розв'язок диференціальних рівнянь;
- виконання операцій у символьному вигляді.

Для роботи в MathCAD не потрібні знання і навички програмування, операції записуються у звичному вигляді. Це дає змогу не лише фахівцям у певній галузі (математикам, інженерам, економістам, хімікам), а й студентам і учням розв'язувати задачі без спеціальної підготовки, звичним способом.

Особливістю системи MathCAD є те, що операції в математичних виразах виконуються одразу після їх введення. У разі будь-якої зміни початкових даних починають перераховуватися всі математичні вирази, до яких ці змінені дані входять прямо або опосередковано.

Крім зазначених математичних функцій, MathCAD має засоби для подання інформації в графічному вигляді за допомогою графіків, діаграм тощо.

Математичні обчислення і їхні результати, подані в графічній формі, супроводжуються, як правило, текстовою інформацією: викладенням засадних положень, умов, пояснень, доведень, коментарів, висновків. Текстовий редактор, убудований до MathCAD, дає змогу готувати наукові статті й звіти, монографії, курсові та дипломні проєкти, електронні книжки з математичних, технічних, економічних та інших дисциплін, у яких математичне оброблення становить велику частину.

MathCAD, як і багато інших програм, що працюють у середовищі Windows, підтримує технологію OLE, тож у документ можна вставляти об'єкти, створені за допомогою інших прикладних програм, — кольорові та чорно-білі графіки, діаграми, схеми, фотографії.

Вікно MathCAD і його елементи

Система MathCAD працює у середовищі Windows, тому має графічний інтерфейс, схожий з іншими прикладними програмами Windows. Вікно системи складається з таких елементів (рис. 16.1):

- рядок заголовка;
- рядок меню;

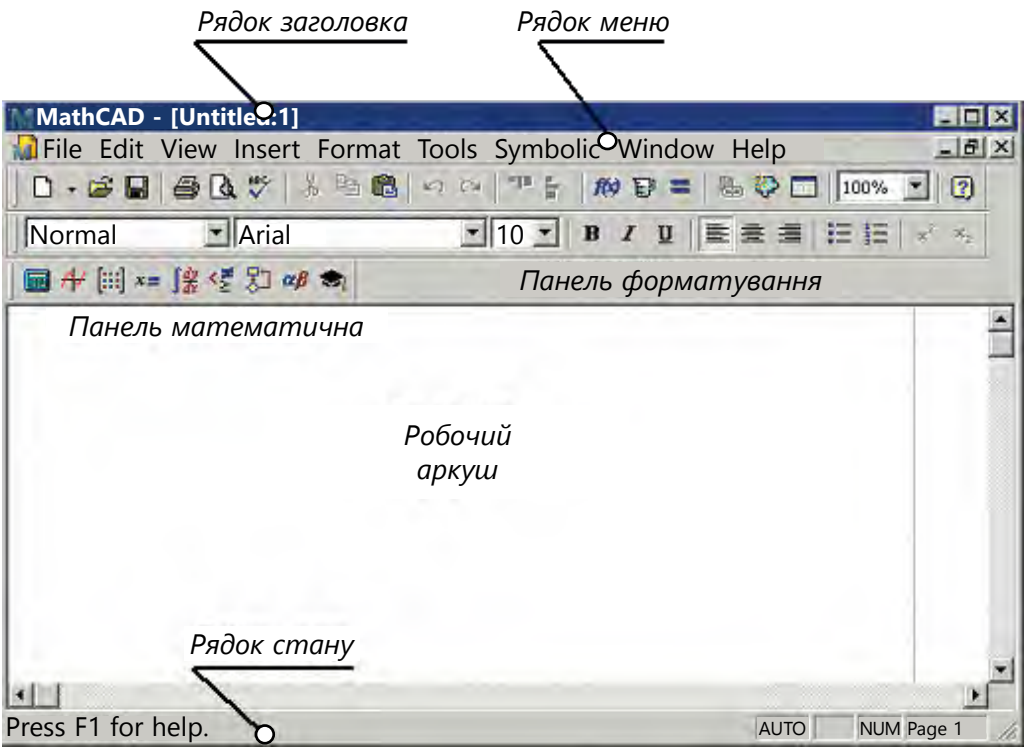


Рис. 16.1

- панель інструментів;
- панель форматування;
- рядок стану;
- вікно документа.

Рядок заголовка містить системну кнопку з піктограмою (картинкою) системи MathCAD, яка відкриває меню керування вікном із командами *Розгорнути, Згорнути, Закрити*. Ці команди можна виконати також за допомогою відповідних кнопок праворуч у рядку заголовка. Поруч із системною кнопкою — назва прикладної програми MathCAD Professional і назва документа.

У рядку меню розміщено кнопки з випадними меню: *File (Файл), Edit (Правка), View (Вид), Insert (Вставка), Format (Формат), Math (Математика), Symbolic (Символьний), Window (Вікно), Help (Допомога)*.

У **рядку стану** в нижній частині вікна виводиться інформація про команди.

На панелі інструментів у вигляді кнопок винесено найчастіше вживані команди з меню.

Основними **панелями інструментів** (рис. 16.2) є *Standart (Стандартна), Formatting (Форматування)* і *Math (Математична)*.

Стандартна панель інструментів містить кнопки керування операціями, які є спільними для багатьох прикладних програм Windows.

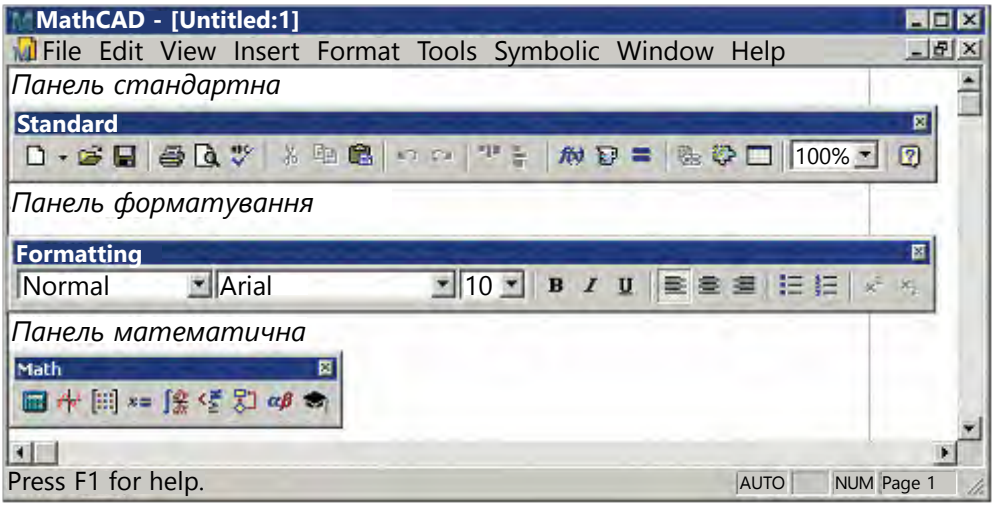


Рис. 16.2

Файлові операції з документами:

- *New (Новий)* — створити новий документ;
- *Open (Відкрити)* — завантажити раніше створений документ, який зберігається в пам'яті;
- *Save (Зберегти)* — зберегти в пам'яті документ з останніми змінами.

Друкування і перевірка орфографії:

- *Print (Надрукувати)* — надрукувати документ на принтері;
- *Print Preview (Перегляд перед друком)* — переглянути документ, перш ніж надрукувати;
- *Check Spelling (Перевірка орфографії)* — перевірити орфографію.

Редагування документів:

- *Cut (Вирізати)* — вилучити з документа виділену частину і розмістити її в буфері обміну *(Clipboart);*
- *Copy (Копіювати)* — копіювати виділену частину документа або весь документ і розмістити в буфері обміну *(Clipboart);*
- *Paste (Вставити)* — перенесення вмісту буфера обміну в документ.

Скасування і відтворення операції:

- *Undo (Скасувати)* — скасувати останню операцію і відповідні зміни в документі;
- *Redo (Відтворити)* — відтворити скасовану операцію.

Розміщення блоків у документі:

- *Align Across (Вирівняти за горизонталлю)* — вирівняти виділені в документі блоки за горизонталлю;
- *Align Down (Вирівняти за вертикаллю)* — вирівняти виділені в документі блоки за вертикаллю.

Операції з математичними виразами:

- *Insert Function (Вставити функцію)* — вставити у вираз математичну вбудовану функцію зі списку;

- *Insert Unit (Вставити одиницю вимірювання)* — вставити одиницю вимірювання фізичної величини;
- *Calculate (Обчислити)* — обчислити виділений математичний вираз.

Вставка об'єктів:

- *Insert Hiperlink (Вставити гіперпосилання);*
- *Insert Component (Вставити компонент);*
- *Insert Table (Вставити таблицю);*
- *Zoom (Масштаб)* — встановити масштаб зображення документа, вибравши його значення зі списку;
- *Help (Допомога)* — отримати потрібну довідку з довідкової бази даних.

Панель математичних інструментів (Math) містить засоби для оброблення математичної інформації, які згруповано за певними ознаками і розміщено на таких панелях (*Toolbar): Calculator (Калькулятор), Graph (Графіки), Vector and Matrix (Вектори і матриці), Evaluation (Оцінювання), Calculus (Обчислення), Boolean (Булеві), Programming (Програмування), Greek (Грецькі символи), Symbolic (Символьні обчислення)* (рис. 16.3). Щоб вивести будь-яку панель на екран, необхідно вибрати відповідну команду з випадного меню **View** — *Tool* або натиснути відповідну кнопку на Панелі математичних інструментів *(Math),* де зведено всі панелі.

Рис. 16.3

Документ MathCAD. Математична, текстова та графічна інформація обробляється у формі окремих електронних об'єктів — документів *MathCAD*. У вікні *MathCAD* можна відкрити одне або кілька *вікон документа*. Кожен документ зберігається й обробляється у вигляді окремого файлу, що має розширення **.mcd.**

На екрані дисплея документ MathCAD має вигляд *робочого аркуша.*

Робочий документ поділено на *сторінки*, які в разі потреби можна надрукувати. Розміри сторінок можна вибрати із списку стандартних або задати інші параметри.

Робочий аркуш поділено вертикальною лінією на дві частини. Ліва частина виводиться на друк, а праву можна не виводити й розміщувати на ній допоміжну інформацію, проміжні громіздкі розрахунки, коментарі. Тому праву частину іноді називають прихованими сторінками.

Інформацію подано у вигляді блоків інформації. *Блок* (англ. region) — це місце на робочому аркуші, обмежене невидимим прямокутником, на якому розмі-

щена математична, графічна, текстова інформація. З кожним блоком можна працювати самостійно: переміщувати, копіювати, редагувати тощо. Інформацію у блоках обробляють у такій послідовності: зліва направо і згори вниз.

Блоки можна виділити більш світлим забарвленням, якщо вибрати команду *Regions* із меню **View**.

Щоб створити блок, необхідно клацнути лівою кнопкою миші у вибраному місці. У цьому місці з'явиться маленький хрестик (+) червоного кольору. Тепер потрібно почати вводити інформацію з клавіатури. Автоматично, з першим введеним із клавіатури символом, MathCAD створить блок. Стандартно тип блоку буде математичним, але автоматично зміниться на текстовий, якщо натиснути на клавішу *Пробіл*.

Щоб виділити блок будь-якого типу, треба клацнути на ньому лівою кнопкою миші. Навколо блоку з'явиться чорна *рамка*. Якщо блок текстовий або графічний, то рамка матиме *маркери* — маленькі чорні квадрати в правому нижньому кутку, а також посередині на нижній і правій сторонах рамки. За допомогою маркерів можна змінювати розміри рамки й, відповідно, блоку: вказівник миші підвести до маркера (вказівник змінить форму на двоспрямовану стрілку), клацнути лівою кнопкою миші й, тримаючи її натиснутою, переміщувати маркер. Відпустивши кнопку, отримаємо блок потрібного розміру.

Щоб виділити не один, а кілька блоків, треба з натиснутою лівою кнопкою миші переміщувати вказівник так, щоб штриховий прямокутник, який з'являється, охопив потрібні блоки. Якщо тепер відпустити кнопку миші, то навколо блоків, які потрапили в зону дії виділення, з'являться штрихові рамки. Виділену в такий спосіб групу блоків можна переміщувати, копіювати, вилучати, вирізати, розміщувати у буфері обміну тощо.

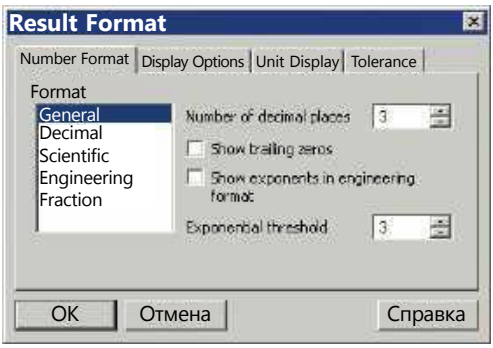
Виділену групу блоків можна вирівняти вздовж горизонтальної *(Align Across)* або вертикальної *(Align Down)* ліній. Для цього потрібно клацнути мишею на відповідній кнопці панелі інструментів або скористатися відповідною командою з меню **Format (Формат)**.

16.3. ЗАСОБИ ОБРОБЛЕННЯ МАТЕМАТИЧНОЇ ІНФОРМАЦІЇ MathCAD

У математичних блоках розрізняють такі математичні об'єкти: числа, змінні, функції, вектори, матриці. Вони входять до складу арифметичних і алгебраїчних виразів, а також рівнянь.

MathCAD може оперувати з такими числами: цілими, раціональними, ірраціональними, комплексними. Формат чисел можна задати на панелі *Result Format,* яка відкривається вкладкою **Format** — *Result...* на рядку меню (рис. 16.4, *а*).

Цілі числа записують без десяткової точки. Раціональні числа MathCAD подає у вигляді двох цілих чисел. Для ірраціональних чисел є дві форми представлення: децимальна (англ. decimal) і експоненціальна (англ. exponential або scientific notation). Комплексні числа складаються з дійсної і уявної частин.



а

220	Ціле число
21.0897	Дійсне число у десятковій формі запису
$2.596 \cdot 10^2$	Дійсне число в експоненціальній формі запису
$\frac{7}{11}$	Раціональне число
$5.9 + 6.7 \cdot j$	Комплексні числа
$9.6 - 7.8 \cdot j$	

б

Рис. 16.4

Ознакою уявної частини є уявна одиниця, яка позначається літерою *i* або *j* (рис. 16.4, *б*).

Числа, з'єднані символами *математичних дій (операцій),* утворюють **арифметичні вирази**. Їх вводять у будь-якому місці робочого аркуша в математичний блок із клавіатури або за допомогою спеціальних засобів — *панелей* (англ. toolbar). Щоб вставити цифру або символ операції в математичний блок, треба клацнути на ньому мишею. У табл. 16.1 наведено оператори найпоширеніших арифметичних дій із числами, які містяться на панелі **Calculator**.

Таблиця 16.1

Оператори панелі *Calculator*

Символ операції	Назва операції	Комбінація клавіш
+	Додавання	+
−	Віднімання	–
×	Множення	Shift+*
/	Ділення	/
:=	Надання значення	Shift+:
=	Обчислення результату	=
√	Добування кореня квадратного	\
ⁿ√	Добування кореня *n*-ї степені	Ctrl+\
|×|	Модуль числа	|
n!	Факторіал	!

Арифметичний вираз обчислюється одразу після того, як тільки після нього вводять знак =. Цей знак є сигналом для математичного процесора виконувати обчислення, записані ліворуч від нього (рис. 16.5). Якщо змінити будь-яке число в арифметичному виразі, процесор автоматично перерахує і виведе результат. У цьому режимі процесор працює як звичайний калькулятор.

$$(23 \cdot 0.698^2) + \frac{254}{78.0095}$$ Арифметичний вираз

$$(23 \cdot 0.698^2) + \frac{254}{78.0095} = 14.462$$ Обчислене значення математичного виразу

Рис. 16.5

Змінною (англ. variables) у MathCAD називають величину, яка змінює своє значення в процесі виконання програми. Змінною може бути число, арифметичний або алгебраїчний вираз, якому надано унікальне *ім'я*, тобто змінна в MathCAD має те саме значення, що і в алгебрі. Ім'я змінної може складатися з довільної кількості допустимих символів: букв (зазвичай латинського алфавіту), цифр, знаків _ і %. Першим символом в імені обов'язково має бути буква. Ім'я змінної має бути коротким і легко запам'ятовуватися. Не треба для цього використовувати назви змінної, вбудованої функції або константи. Ім'я змінної вводять, як правило, з клавіатури.

Операція присвоювання. Після імені змінної обов'язково має стояти знак присвоювання :=. Праворуч від знака присвоювання може стояти або цифра, або алгебраїчний вираз. Якщо праворуч від знака присвоювання стоїть число, то це означає, що змінній надано значення, виражене цим числом. Якщо ж праворуч стоїть алгебраїчний вираз, то це означає, що змінна визначається через інші змінні й числа, що входять до складу алгебраїчного виразу. Змінні можуть входити до складу математичного виразу, бути аргументом функції або операндом.

Операція виконання. Щоб отримати значення змінної, необхідно після імені змінної ввести знак =. Після введення цього знака MathCAD одразу ж виводить результат. До цього всі змінні, що входять до складу алгебраїчного виразу, мають бути визначені через інші змінні або через числа. Якщо цю умову не виконано, після введення знака = ім'я змінної або алгебраїчний вираз змінює колір на яскраво-червоний і виводиться повідомлення «This variable or function is not defined above» («Цю змінну або функцію вище не визначено»).

Крім звичайних змінних у MathCAD використовують так звані *рангові змінні*, тобто низку чисел з однаковим інтервалом між ними. Щоб задати рангову змінну, потрібно вибрати і ввести її ім'я, знак присвоювання, початкове значення, після нього поставити кому (,), ввести значення інтервалу між числами, далі натиснути клавішу (;) і, нарешті, ввести кінцеве значення змінної. На рис. 16.6 показано послідовність уведення рангової змінної.

Змінну величину, яка крім свого значення характеризується одиницею вимірювання, називають *розмірною змінною*. Тобто це фізична величина, яка дорівнює добутку значення величини на одиницю вимірювання величини.

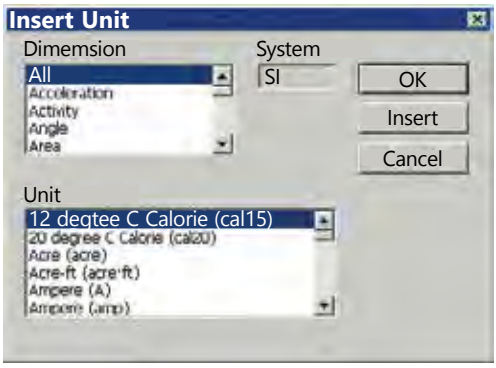
Одиниці вимірювання в міжнародній системі SI розміщено на вкладці *Insert Unit* (рис. 16.7, *а*), яку можна викликати зі списку **Insert** — *Unit* рядка меню, або з панелі *Standart* кнопкою , або комбінацією клавіш *Ctrl+U*.

За допомогою розмірних змінних зручно розв'язувати фізичні задачі. На рис. 16.7, *б* наведено приклад представлення напруги як розмірної змінної.

Числа, змінні, вбудовані й задані споживачем функції можуть входити до складу **алгебраїчних виразів** — імен змінних, функцій і цифр, з'єднаних символами математичних операцій. Щоб отримати значення алгебраїчного виразу, треба після алгебраїчного виразу ввести знак = (рис. 16.8).

X	Ім'я змінної
X :=	Знак присвоювання
X := 0,	Увести початкове значення
X := 0,0.1	Увести крок зміни
X := 0,0.1..	Натиснути клавішу ;
X := 0,0.1.. 12.5	Увести кінцеве значення

Рис. 16.6



а

U	Увести ім'я розмірної змінної Розмірна змінна — фізична величина: електрична напруга
U :=	Увести знак присвоювання
U := 220	Присвоїти електричній напрузі 220 одиниць
U := 220 · V	Увести одиницю електричної напруги у системі SI — вольт (V)

б

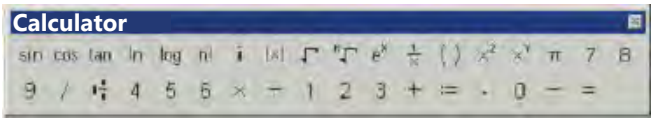
Рис. 16.7

$a_1 \cdot x_2 + b_1 \cdot x + c_1$	Алгебраїчний вираз: Квадратний поліном
$a_1 := \frac{3}{4}$ $b_1 := 5.7$ $c_1 := 357.6 \cdot 10^{-2}$	Значення коефіцієнтів
$f_1(x) := a_1 \cdot x^2 + b_1 \cdot x + c_1$	Визначення функції
$f_1(3.087 \cdot 10^2) = 7.323 \times 10^4$	Значення функції при значенні аргумента $3.087 \cdot 10^2$

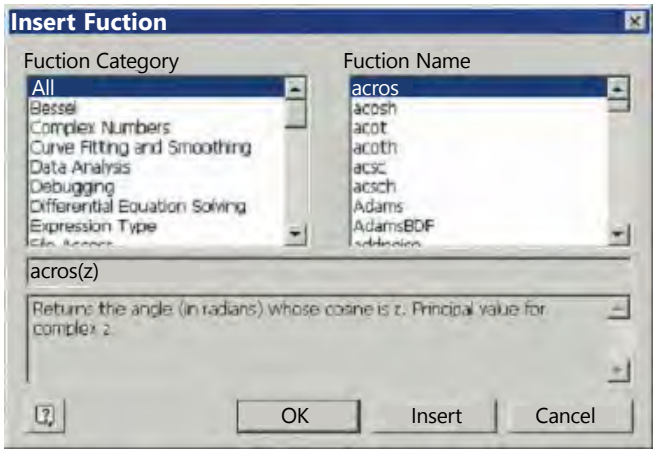
Рис. 16.8

Функція — це залежність однієї змінної від однієї чи кількох інших змінних, які називають **аргументами**. У MathCAD функція, як і змінна, має унікальне *ім'я*. Вимоги до імені функції такі самі, як і до імені змінної. Після імені у дужках зазначають аргументи або аргумент функції. При зверненні до функції вона повертає результат.

Функції поділяють на *вбудовані* й *задані користувачем*. MathCAD надає в розпорядження користувача широкий набір убудованих елементарних функцій, функцій вищої математики, спеціальних функцій, зокрема логарифмічну, показникову, тригонометричні, обернені тригонометричні, прямі й обернені гіперболічні та ін. Елементарні математичні функції розміщено на панелі *Calculator* (рис. 16.9, *а*), а функції всіх категорій, які застосовують у різноманітних галузях математики, — на панелі *Insert Function* (рис. 16.9, *б*). Панель *Insert Function* можна викликати на вкладці *Insert — f(x) Functio* рядка меню, або на вкладці *f(x)* на панелі *Standart*, або комбінацією клавіш *Ctrl+E*.



а



б

Рис. 16.9

Користувач може створити свою функцію будь-якого ступеня складності: ввести її ім'я і аргументи, розділені комою, потім ввести знак присвоювання і після цього — математичний вираз, що визначає цю функцію. Приклад обчислення значення функції наведено на рис. 16.10.

$z_2(x,y) := x^2 + \sqrt{y}$	Завдання функції
$x := 37.345$ $y := 6.839$	Завдання значень аргументам
$z_2(x,y) = 1.397 \times 10^3$	Обчислення значення функції

Рис. 16.10

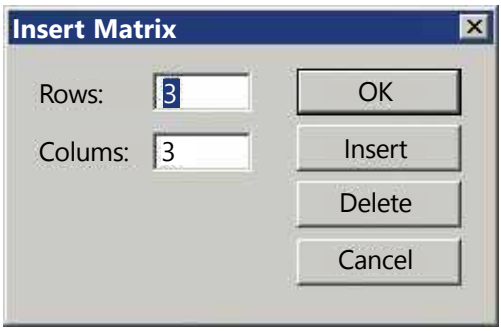
Функцію можна визначити через інші функції, які були визначені раніше, а також через вбудовані функції.

Треба зауважити, що імена змінних і функцій сприймаються однаково, тому їм потрібно давати різні імена.

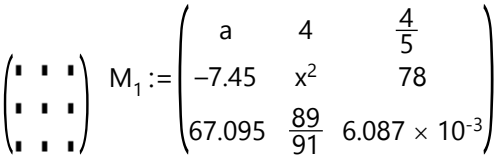
Вектори й матриці. Масив (сукупність) чисел, поданих у формі прямокутної таблиці, називають *матрицею*. У MathCAD матриця має те саме значення, що й у математиці.

Щоб створити матрицю, необхідно вибрати команду *Matrix* із меню *Insert* або скористатися комбінацією клавіш *Ctrl+M*. Після вибору цієї команди на екрані з'являється діалогове вікно, у якому в полях *Rows (Рядки)* і *Colums (Стовпці)* треба набрати потрібне число рядків і стовпців матриці

(рис. 16.11, *а*). Після натискання на кнопку діалогового вікна на екрані з'являється трафарет матриці з маленькими чорними прямокутниками на місці елементів матриці (рис. 16.11, *б*). Тепер можна послідовно, один за одним вводити елементи матриці, спочатку виділивши відповідний чорний прямокутник, клацнувши на ньому лівою кнопкою миші. Елементом матриці може бути число, змінна або рядкова змінна.



а



б *в*

г

Рис. 16.11

Матриці, як і будь-якій змінній, можна дати *ім'я* за допомогою оператора := (рис. 16.11, *в*). Будь-який елемент створеної матриці можна скоригувати, для чого достатньо клацнути мишею на потрібному елементі й ввести новий елемент замість старого.

Під *вектором* розуміють масив (сукупність) чисел, розміщених у певній послідовності. Вектор можна розглядати як матрицю, у якої є тільки один стовпець (вектор-стовпець) або тільки один рядок (вектор-рядок). Вектор має багато спільного з ранговою змінною.

Із матрицями можна виконувати операції, які розміщені на вкладці *Matrix* (рис. 16.11, *г*). Вкладку можна викликати з рядка меню командами **View** — *Toolbars* — *Matrix*.

Операції з математичними об'єктами. Крім елементарних арифметичних і алгебраїчних операцій, у MathCAD можна виконувати операції вищої математики, які містяться на вкладці *Calculus* (рис. 16.12). Вкладку можна викликати з рядка меню командами **View** — *Toolbars* — *Calculus* або з панелі математичних операцій *Math*.

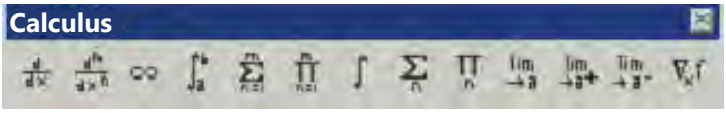


Рис. 16.12

Із вкладки *Calculus* можна викликати такі операції:

$\frac{d}{dx}$ — взяття похідної;

$\frac{d^n}{dx^n}$ — взяття похідної *n*-го порядку;

∞ — нескінченність;

— визначений інтеграл;

— сума елементів;

— добуток елементів;

— невизначений інтеграл;

— сума елементів рангової змінної;

— добуток елементів рангової змінної;

— двостороння границя;

— правостороння границя;

— лівостороння границя;

— градієнт.

Розв'язок рівнянь і систем рівнянь застосовують у багатьох прикладних галузях, це один із найважливіших розділів математики.

Корінь одного рівняння з одним невідомим знаходять за допомогою функції *root(),* яку викликають із панелі *Insert Function* із категорії *Solving*. Аргументами *root()* є функція, корінь якої потрібно добути, аргумент, початкове і кінцеве значення діапазону. На рис. 16.13 наведено приклад застосування функції *root()* для знаходження кореня полінома третьої степені в проміжку [-6,6].

Якщо потрібно знайти всі корені полінома *n*-ї степені, застосовують функцію *polyroots()*. На рис. 16.13, *а* подано приклад застосування функції *polyroots()* для знаходження всіх коренів полінома третьої степені.

Систему рівнянь розв'язують засобами MathCAD всередині *блоку розв'язку* (англ. solve block) у такій послідовності:

1. Припускають, що невідомі величини мають певні значення, які потрібно задати на початковому етапі. Це роблять для того, щоб ітераційний процес знаходження розв'язків почався з конкретних значень.

2. Уводять командне слово *Given (Дано)* нижче початкових заданих значень невідомих.

3. Уводять *обмеження (рівняння, нерівності)* у довільній послідовності нижче командного слова *Given*. У рівняннях слід використовувати лише логічний знак рівності =, який уводять комбінацією клавіш *Ctrl*+=.

Коефіцієнти полінома

$a_3 := 0.8 \quad a_2 := 0 \quad a_1 := -12 \quad a_0 := 5$

Поліном третього степеня

$f(x) := a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x^1 + a_0 \cdot x^0$

Знаходження коренів полінома у проміжку (-6,6)

$\mathrm{root}(f(x), x, -6, 6) = -4.067$

Корінь полінома дорівнює –4.067

а

Рис. 16.13

Коефіцієнти полінома

$a_3 := 0.8 \quad a_2 := 0 \quad a_1 := -12 \quad a_0 := 5$

Поліном третього степеня

$f(x) := a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x^1 + a_0 \cdot x^0$

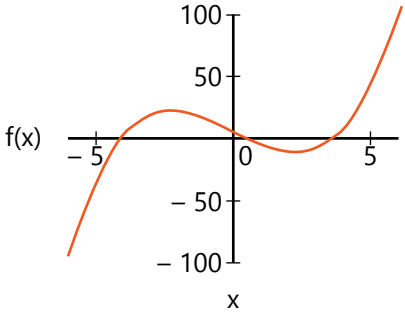
Вектор коефіцієнтів полінома

$$k := \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Поліном третього степеня має три корені:
-4.067, 0.422, 3.645

$$\text{polyroots}(k) = \begin{pmatrix} -4.067 \\ 0.422 \\ 3.645 \end{pmatrix}$$

Графік полінома



б

Рис. 16.13

4. Уводять командне слово Find*(x*1*, x*2*, … x*n*)* нижче обмежень. *x*1*, x*2*, … x*n — невідомі, значення яких потрібно знайти.

На рис. 16.14 наведено приклад розв'язування системи лінійних рівнянь із галузі електротехніки. Коефіцієнти в правій частині рівняння — це *власні R*11*, R*22*, R*33 та *взаємні R*12= *R*21*, R*23= *R*32*, R*13= *R*31 опори контурів кола. Одиниця вимірювання опорів — оми (Ω). Коефіцієнти в правій частині рівнянь — електрорушійні сили у контурах *E*11*, E*22*, E*33. Одиниця вимірювання електрорушійних сил — вольти (V). Невідомі величи-

Задаємо значення коефіцієнтів у лівій частині рівнянь

$R11 := 10 \cdot \Omega \quad R22 := 15 \cdot \Omega \quad R33 := 20 \cdot \Omega$

$R12 := -38 \cdot \Omega \quad R23 := -45 \cdot \Omega \quad R31 := -56 \cdot \Omega$

$R21 := R12 \quad R32 := R23 \quad R13 := R31$

Задаємо значення коефіцієнтів у правій частині рівнянь

$E11 := 6 \cdot V \quad E22 := 12 \cdot V \quad E33 := -18 \cdot V$

Задаємо початкові значення невідомих величин

$I11 := 1 \cdot mA \quad I22 := 2 \cdot mA \quad I33 := 3 \cdot mA$

Розв'язуємо систему лінійних рівнянь

Given

$R11 \cdot I11 + R12 \cdot I22 + R13 \cdot I33 = E11$

$R21 \cdot I11 + R22 \cdot I22 + R23 \cdot I33 = E22$

$R31 \cdot I11 + R32 \cdot I22 + R33 \cdot I33 = E33$

$$\text{Find}(I11, I22, I33) = \begin{pmatrix} 59.328 \\ 217.626 \\ -244.224 \end{pmatrix} mA$$

Рис. 16.14

ни — струми у контурах *I*11, *I*22, *I*33. Одиниця вимірювання струмів — міліампери (mA).

Математичні операції у символьному вигляді. Для розв'язання багатьох прикладних задач необхідно виконати перетворення математичних виразів у символьному вигляді, наприклад, перш ніж виконати складні обчислення і розрахунки, доцільно спростити математичний вираз або рівняння. Для виконання таких операцій у складі MathCAD передбачено ядро символьного процесора системи Maple.

Символьні операції можна задати такими способами:

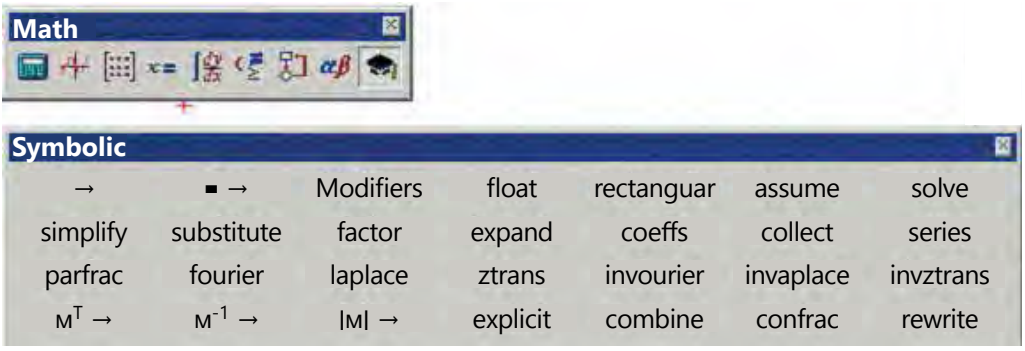
- за допомогою символьного знака рівності;
- вибрати операцію з випадного меню **Symbolics**;
- вибрати потрібну символьну операцію з панелі *Symbolic*.

Символьний знак рівності має вигляд стрілки → і викликається комбінацією клавіш *Ctrl*+. (точка). Символьний процесор працює як інтерпретатор, тобто перетворення здійснюються одразу ж після введення символьного знака рівності. Якщо в лівій частині від символьного знака рівності виконати будь-яку зміну виразу, то весь робочий аркуш нижче від внесеної зміни одразу ж перераховується. На рис. 16.15 наведено приклади перетворення виразів під дією операцій інтегрування і диференціювання.

Другим способом здійснити символьні перетворення є використання вкладки *Symbolics* із рядка меню.

$\rightarrow$	Знак символьного перетворення
$\int_a^b x^2\,dx$	Символьний вираз інтегрування функції
$\int_a^b x^2\,dx \rightarrow \frac{b^3}{3} - \frac{a^3}{3}$	Символьне перетворення
$\frac{d}{dx}(a \cdot x^2 + b \cdot x + c)$	Символьний вираз диференціювання функції
$\frac{d}{dx}(a \cdot x^2 + b \cdot x + c) \rightarrow b + 2 \cdot a \cdot x$	Символьне перетворення

а



б

Рис. 16.15

Третій спосіб — викликати панель *Symbolic*, натиснувши на кнопку *Symbolic Keyword Toolbar* на панелі інструментів *Math* (рис. 16.15, *б*).

Розглянемо кілька прикладів застосування символьних перетворень.

Громіздкі вирази доцільно *спрощувати*, вибравши на панелі *Symbolic* ключове слово *(Keyword) — simplify (спростити)*. На рис. 16.16, *а* зображено спрощення дробу, а на рис. 16.16, *б* — визначника матриці.

На рис. 16.16, *в* наведено приклад *розв'язування квадратного рівняння,* ключове слово — *solve (розв'язати)*. На рис. 16.16, *г* — розкладання виразу за степенями, ключове слово — *expand (розширити)*.

Якщо потрібно здійснити складне перетворення, то *застосовують кілька ключових слів*. На рис. 16.16, *д* показано приклад, де визначник матриці спо-

$$\frac{\left[\frac{L1-(R2+R3)+C1-R2-R3-R4}{R2-R4}\right]}{\left(\frac{C1-L1-R3}{R4}\right)} \text{ simplify } \rightarrow \frac{R1}{L1} + \frac{R2+R3}{C1-R2-R3}$$

а

$$\left\|\begin{bmatrix} \left(1+p-C1-R3+\frac{R3}{R2}\right) & 0 & 0 \\ -\left(p-C1-R3+\frac{R3}{R2}\right) & -\left[p-L1+\frac{R5-(p-L1)}{R4}+R5\right] & 1 \\ \frac{R3}{R2}-\left(p-C1+\frac{1}{R2}\right) & -\left[\frac{(p-L1)}{R4}+1\right] & 0 \end{bmatrix}\right\| \text{ simplify } \rightarrow \frac{(R4+L1-p)-(R2+R3+C1-R2-R3-p)}{R2-R4}$$

б

$$(a\cdot x^2+b\cdot x+c) \text{ solve,x } \rightarrow \begin{pmatrix} -\frac{\frac{b}{2}-\frac{\sqrt{b^2-4-a-c}}{2}}{a} \\ -\frac{\frac{b}{2}+\frac{\sqrt{b^2-4-a-c}}{2}}{a} \end{pmatrix}$$

в

$$(a\cdot x^2+b\cdot x+c)^2 \text{ expand,x } a^2\cdot x^4+2\cdot a\cdot b\cdot x^3+2\cdot a\cdot c\cdot x^2+b^2\cdot x^2+2\cdot b\cdot c\cdot x+c^2$$

г

$$\left\|\begin{bmatrix} \left(1+p-C1-R3+\frac{R3}{R2}\right) & 0 & 0 \\ -\left(p-C1-R3+\frac{R3}{R2}\right) & -\left[p-L1+\frac{R5-(p-L1)}{R4}+R5\right] & 1 \\ \frac{R3}{R2}-\left(p-C1+\frac{1}{R2}\right) & -\left[\frac{(p-L1)}{R4}+1\right] & 0 \end{bmatrix}\right\|$$

$$\left|\begin{matrix}\text{simplify} \\ \text{collect,p}\end{matrix}\right. \rightarrow \left(\frac{C1-L1-R1}{R4}\right)-p^2+\left[\frac{L1-(R2+R3)+C1-R2-R3-R4}{R4}\right]-p^2+\frac{R3+R2}{R2}$$

д

Рис. 16.16

чатку спрощується *(simplify),* а потім зводяться подібні коефіцієнти *(collect)* перед степенем полінома.

16.4. СТВОРЕННЯ ГРАФІКІВ ЗАСОБАМИ MathCAD

Інформація, подана у вигляді графіків, краще сприймається, має більшу наочність, дає змогу виявити взаємозв'язок між досліджуваними величинами. Система MathCAD надає в розпорядження користувача широкий набір засобів для візуалізації результатів аналізу шляхом створення двовимірних і тривимірних графіків різноманітної форми.

Засоби створення графіків розміщено на панелі *Graph*, яку можна викликати з рядка меню **View** — *Toolbars* — *Graph* або з панелі інструментів *Math* кнопкою *Graph* (рис. 16.17).

Рис. 16.17

На панелі містяться кнопки для побудови таких графіків:

X-Y Plot — двовимірний графік у декартовій системі координат;

Polar Plot — двовимірний графік у полярній системі координат;

Surface Plot — тривимірний графік поверхні;

Contour Plot — тривимірний графік поверхні за допомогою ліній рівня;

3D Bar Plot — тривимірна стовпчаста діаграма;

3D Scatter Plot — тривимірна діаграма, утворена множиною розсіяних точок;

Vector Field Plot — тривимірна діаграма поля векторів.

Для *побудови двовимірного графіка в декартовій системі координат* спочатку потрібно вивести його шаблон одним із трьох способів: із рядка меню **View** — *Toolbars* — *Graph* — *X-Y Plot;* з панелі *Graph* кнопкою ; комбінацією клавіш *Ctrl+@*. Функцію, яку потрібно зобразити на графіку, необхідно ввести вище і лівіше шаблону. На шаблоні внизу посередині розташований маленький чорний прямокутник, у який потрібно ввести аргумент функції. Ліворуч посередині в чорний прямокутник ввести саму функцію. Після введення аргументу й самої функції треба клацнути лівою кнопкою миші

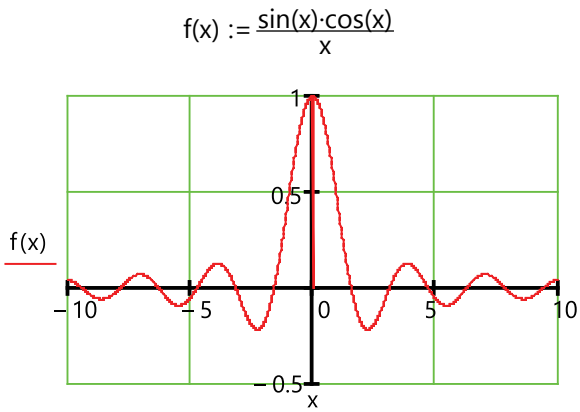


Рис. 16.18

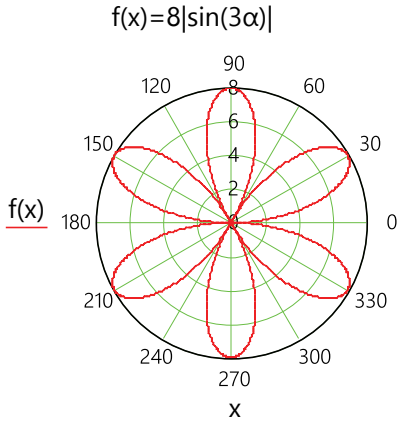


Рис. 16.19

в будь-якому пустому місці робочого аркуша, і графічний процесор побудує графік введеної функції. На рис. 16.18 показано побудований у MathCAD графік функції $f(x)=\sin(x)\cos(x)/x$.

Шаблон для *побудови двовимірного графіка у полярній системі координат* викликають аналогічно попередньому. Внизу шаблона вводять аргумент функції, яким у полярній системі є кут, а ліворуч — функцію, тобто залежність радіуса-вектора від кута. На рис. 16.19 показано побудований у полярній системі координат графік функції $f(x)=8|\sin(3\alpha)|$.

16.5. ВИКОРИСТАННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ У MathCAD

У наукових статтях, технічних звітах, дисертаціях, курсових і дипломних проєктах поряд із математичними виразами, рівняннями, формулами необхідні пояснення, коментарі, виклад засадних положень, аналіз отриманих результатів та інша текстова інформація.

Введення тексту. Текстову інформацію в MathCAD оформлено у вигляді текстових блоків. Для створення текстового блоку потрібно клацнути мишею в тому місці робочого аркуша, де потрібно розмістити текстовий блок. Далі вибрати команду *Text Region (Текстовий блок)* із меню *Insert (Вставити)*. Цю саму команду можна викликати за допомогою клавіатури комбінацією клавіш *Ctrl* + “. Далі вводити символи з клавіатури. Після введення першого символу навколо нього з’явиться прямокутний контур із трьома маркерами. Із введенням наступних символів контур автоматично розширюватиметься. Текстовий блок створюється автоматично, якщо після першого введеного слова натиснути клавішу *Пробіл*.

Щоб закінчити введення інформації в текстовий блок, необхідно клацнути мишею в будь-якому місці робочого аркуша поза блоком. Натискування на клавішу *Пробіл*, як це часто передбачено в інших прикладних програмах,

не призводить до закінчення операції введення, а натомість буде введено новий рядок тексту.

Форматування тексту. До складу MathCAD входить текстовий редактор, який дає змогу створювати текстові документи, придатні для публікації. Панель форматування, яку можна викликати із рядка меню **View** — *Toolbars* — *Formatting,* містить інструменти для надання текстовій інформації потрібного вигляду:

- *Style (Стиль)* — стиль тексту, тобто сукупність параметрів оформлення текстової інформації;
- *Font (Шрифт)* — вибір зі списку потрібного шрифту;
- *Font Size (Розмір шрифту)* — вибір зі списку потрібного розміру шрифту;
- *Bold (Напівжирний), Italic (курсив), Underline (Підкреслений)* — зображення символів шрифту:
- *Align Left (Вирівнювання ліворуч), Align Center (Вирівнювання по центру), Align Right (Вирівнювання праворуч)* — вирівнювання тексту:
- *Bullets (Маркований список), Numbering (Нумерований список)* — списки;
- *Superscript (Верхній індекс), Subscript (Нижній індекс)* — індекси.

ТЕСТОВІ ЗАВДАННЯ

ДЛЯ РОЗВ'ЯЗУВАННЯ СИСТЕМ РІВНЯНЬ ВИКОРИСТОВУЮТЬ ТАКІ ЗАРЕЗЕРВОВАНІ СЛОВА:

1. Begin, End
2. Start, End
3. Given, Find
4. Given, End
5. Begin, Find

ДЛЯ НАДАННЯ ЗМІННІЙ ЗНАЧЕННЯ ВИКОРИСТОВУЮТЬ КОМБІНАЦІЮ КЛАВІШ:

1. Shift + ;
2. Ctrl + /
3. Alt + :
4. Shift + %

Розділ 17
КОМП'ЮТЕРНІ МЕРЕЖІ. ІНТЕРНЕТ

17.1. КОМП'ЮТЕРНІ МЕРЕЖІ

Застосування комп'ютерів у всіх сферах людської діяльності дало змогу обробляти великі масиви інформації. Ускладнення виробництва, використання новітніх технологій зумовлюють різке зростання інформаційних потоків. Крім того, у сучасному суспільстві відбуваються інтенсивні процеси глобалізації. Тож важливим є питання об'єднання комп'ютерів у систему, створення єдиного інформаційного простору.

Комп'ютерна мережа — це сукупність взаємопов'язаних через канали комп'ютерів, а також апаратних, програмних, інформаційних та інших засобів для забезпечення цієї взаємодії (рис. 17.1).

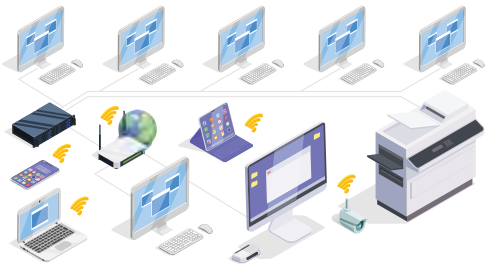


Рис. 17.1

Класифікація комп'ютерних мереж

Комп'ютерні мережі можна класифікувати за різними ознаками: ступенем територіального розподілу, топологією, методами з'єднання (комутації) комп'ютерів. За ступенем територіального розподілу комп'ютерні мережі поділяють на:

- *локальні* (місцеві) (local area network, LAN), що охоплюють порівняно невеликі території (кілька квадратних кілометрів), наприклад, територію підприємства, організації чи фірми, і характеризуються наявністю простої, але доволі швидкісної системи передавання даних;
- *регіональні* (regional area network, RAN), розташовані в межах визначеного територіального регіону (групи споріднених підприємств, міста, області тощо) і характеризуються більшою складністю порівняно з локальними мережами. Можуть використовуватися для об'єднання кількох локальних мереж в інтегровану систему;
- *глобальні* системи (global area network, GAN) охоплюють територію держави, кількох держав або навіть континентів.

Топологія мереж

Топологія — це геометрична схема з'єднання вузлів мережі. Найбільшого поширення набули мережі з такою топологією: кільце, шина (магістраль), зірка, дерево, повнозв'язна.

У мережах із *шинною топологією* канал зв'язку, що з'єднує комп'ютери в мережу, утворює розімкнуту лінію, на кінцях якої встановлюють спеціальні

заглушки (термінатори) (рис. 17.2). Дані від вузла-передавача поширюються каналом в обох напрямках до кінців каналу. Дані одночасно отримують усі вузли. Вузол-приймач розпізнає дані, призначені для нього, і читає їх. Шинна топологія характеризується стійкістю роботи в разі несправності окремих вузлів, можливістю нарощування. Разом із тим мережі цього типу мають невелику довжину.

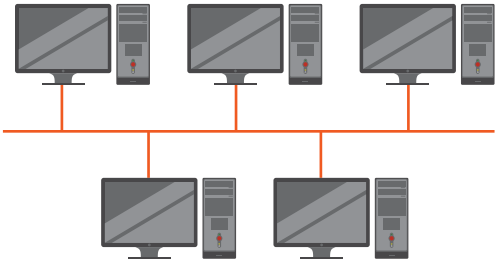


Рис. 17.2

У *кільцевій топології* вузли з'єднані послідовно один з одним і утворюють кільце. Сигнали передаються в одному напрямку кільцем і проходять через мережний адаптер кожного комп'ютера (рис. 17.3). Кожен комп'ютер є повторювачем: приймає сигнали і передає наступному. Кільцева топологія має низку переваг: простота виявлення несправних ділянок, автоматичне підтвердження приймання, можливість використання різних фізичних каналів на різних ділянках, забезпечення гарантованого доступу до каналу навіть у дуже навантаженій мережі.

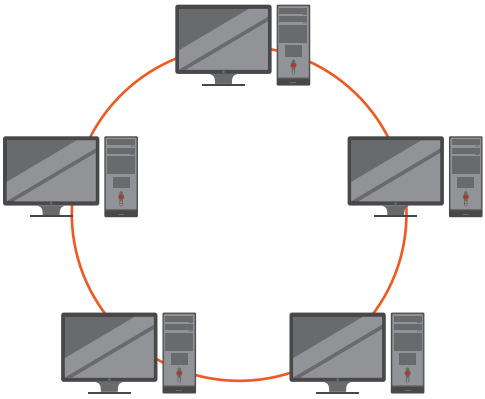


Рис. 17.3

Основними недоліками кільцевої топології є: залежність надійності мережі від усіх ділянок каналу, складність розширення мережі без переривання її функціонування, затримка сигналу в мережних адаптерах.

У мережах *із зіркоподібною топологією* усі комп'ютери за допомогою сегментів кабелю під'єднують до центрального компонента — *концентратора* (англ. hub) (рис. 17.4). Сигнал від комп'ютера-передавача надходить через концентратор до всіх інших. Мережа з такою топологією надає змогу використовувати в різних напрямках різні типи каналів. Вихід із ладу будь-якого комп'ютера, клієнта або кабельного з'єднання не впливає на роботу інших комп'ютерів мережі. Майже всі недоліки цієї топології пов'язані з центральним компонентом (концентратором): порушення роботи всієї мережі в разі виведення його з ладу, складність технології та висока вартість центрального вузла, обмежена кількість портів.

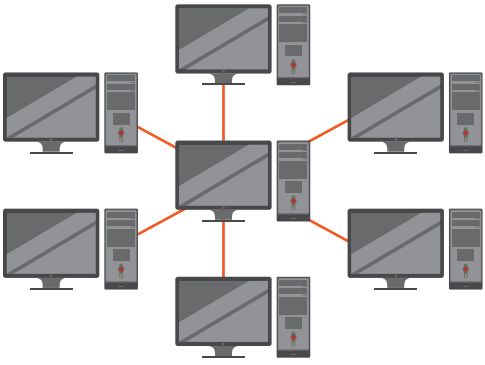


Рис. 17.4

У мережах *із деревоподібною*, або *ієрархічною, топологією* кожен вузол пов'язаний з одним керівним вузлом вищого рівня і одним чи кількома керованими вузлами нижчого рівня (рис. 17.5). Топологія деревоподібної мережі здебільшого відображає ієрархічну організаційну структуру установи, для якої цю мережу створено. Така мережа має переваги в аспекті простоти керування, можливості розширення. Однак якщо виникне несправність у вузлі, то всі вузли нижніх рівнів буде вимкнено з мережі.

У повнозв'язній мережі кожен вузол пов'язаний безпосередньо з усіма іншими вузлами (рис. 17.6). Перевагою такої мережі є висока надійність функціонування. Недолік — значне зростання кількості каналів зв'язку зі збільшенням кількості вузлів.

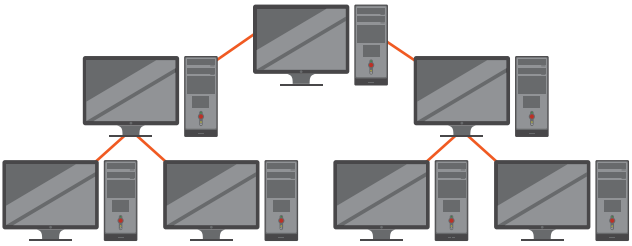


Рис. 17.5

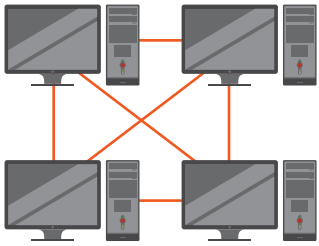


Рис. 17.6

Треба зазначити, що розглянуті топології характерні, здебільшого, для локальних мереж. Регіональні, а тим паче глобальні мережі мають у своєму складі ділянки з різними топологіями.

Взаємодія компонентів комп'ютерної мережі

Можливість функціонування різнотипних комп'ютерів у складі комп'ютерної мережі можна забезпечити в тому випадку, якщо ці комп'ютери відповідають певним *єдиним системним стандартам*, які забезпечують єдність інтерфейсів і правил взаємодії. Міжнародна організація зі стандартизації *ISO (International Standards Organization)* розробила еталонну модель взаємодії відкритих систем *OSI (open system interconnection reference model)*. *Відкрита система* — це система, що взаємодіє з іншими системами відповідно до прийнятих стандартів (рис. 17.7).

Згідно з моделлю відкритих систем, стандартизацію апаратних і програмних засобів проводять на підставі *протоколів* — ієрархічної взаємопов'язаної системи правил взаємодії.

Основою моделі відкритих систем OSI є багаторівнева організація взаємодії. Відповідно до цієї моделі в комп'ютерній мережі можна викорис-

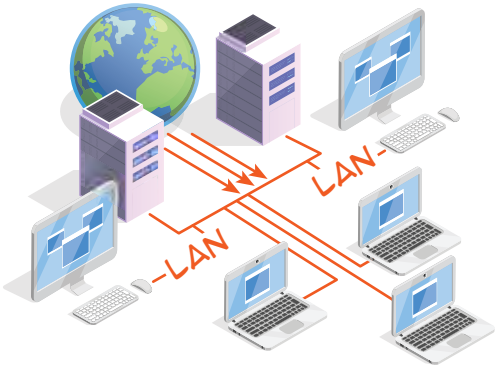


Рис. 17.7

товувати до семи рівнів взаємодії: прикладний (англ. application), представницький (англ. presentation), сеансовий (англ. session), транспортний (англ. transport), мережний (англ. network), канальний (англ. data link), фізичний (англ. physical).

Прикладний рівень є найвищим в ієрархії, оскільки безпосередньо пов'язаний із прикладними процесами й обслуговує взаємодію між ними. Цей рівень забезпечує взаємодію (інтерфейс) між прикладними програмами і мережею. На цьому рівні обробляють дані у тому вигляді, як вони надходять від прикладного процесу і доставляються до інших прикладних процесів.

Представницький рівень забезпечує перетворення формату, синтаксису даних для їх передавання мережею. Іншими словами, здійснюється перетворення (трансляція) з різних мов, форматів, кодів, з якими працюють різнотипні комп'ютери, на форму, придатну для передавання мережею.

Сеансовий рівень забезпечує керування передаванням інформації між прикладними процесами. Він дає змогу звертатися до будь-якого процесу за його іменем, незалежно від того, на якому комп'ютері відбувається цей процес.

Транспортний рівень визначає адресацію фізичних пристроїв у мережі. Однак головним завданням цього рівня є забезпечення ефективних, зручних і надійних форм передавання інформації. Часто інформація ділиться на окремі блоки — кадри, або пакети, які пересилаються мережею від одного комп'ютера до іншого.

Мережний рівень слугує для визначення маршрутів руху пакетів каналами мережі, керування інформаційними потоками, виявлення помилок передавання і повідомлення про них.

На *канальному* рівні здійснюється відповідне оформлення пакетів даних і надійне передавання через фізичний канал.

Найнижчий, *фізичний* рівень забезпечує взаємодію комп'ютера з фізичним каналом. На цьому рівні формуються сигнали, тобто фізичні процеси, що несуть інформацію.

Відповідно до семирівневої моделі розрізняють і сім видів протоколів.

Устаткування комп'ютерних мереж

Для забезпечення обміну інформацією між комп'ютерами мережі й доступу до інформаційних ресурсів до складу комп'ютерної мережі входять багато різних пристроїв. Комп'ютери користувачів, які є складниками мережі, називають *робочими станціями*.

Сервери мережі — спеціалізовані комп'ютери, призначені для керування потоками інформації в мережі й розподілом її ресурсів.

Засоби передавання інформації. Комп'ютери мережі віддалені один від одного в просторі, тож для обміну сигналами їх необхідно з'єднати провідними і безпровідними середовищами передавання сигналів.

У *провідних* середовищах використовують такі види кабелів:

- неекранована (англ. unshielded) і екранована (англ. shielded) вита пара (англ. twisted pair);

- коаксіальний кабель (англ. coaxial cable);
- оптоволоконний кабель (англ. fiber cable).

Вибирають тип кабелю, зважаючи на такі його характеристики: простота установлення, завадостійкість, швидкість передавання, кількість вузлів, які можна під'єднати, максимальна відстань прокладання, ціна.

Вита пара — це два перевиті один навколо одного ізольовані електричні проводи (рис. 17.8, *а*). Скручування проводів зменшує рівень завад, що наводяться сусідніми кабелями і кабелем живлення. Використовують неекранована пару *UTP (unshielded twisted pair)* і екрановану пару *STP (shielded twisted pair)*. Неекранована скручена пара є порівняно дешевою, її широко застосовують у локальних комп'ютерних мережах і телефонних мережах. Максимальна довжина відрізка кабелю — 100 м. Основний недолік неекранованої скрученої пари — низька завадостійкість.

Екранована вита пара має екран, який зменшує вплив зовнішніх електромагнітних полів і тим самим підвищує завадостійкість.

Коаксіальний кабель складається із внутрішнього провідника (зазвичай мідної жили), оточеного шаром ізоляційного матеріалу (поліхлорвінілу, тефлону), зовнішнього провідника (спеціальний екран із переплетених мідних проводів) та оболонки (рис. 17.8, *б*). У локальних мережах застосовують два типи коаксіальних кабелів: тонкий і товстий.

а *б*

Рис. 17.8

Тонкий коаксіальний кабель — це гнучкий кабель діаметром 5 мм, простий у застосуванні, придатний для будь-якого типу мереж. Під'єднується безпосередньо до плати мережного адаптера комп'ютера. Максимальна відстань передавання сигналів становить 185 м. Хвильовий опір кабелю — 50 Ом.

Товстий коаксіальний кабель — порівняно жорсткий кабель діаметром 10 мм, за ціною дорожчий за тонкий кабель. Максимальна відстань передавання сигналів — до 500 м.

Оптоволоконний кабель складається з оптичного волокна (жили), яка покрита скляною оболонкою і має зовнішню захисну оболонку (рис. 17.9). Інформація передається у вигляді модульованих світлових імпульсів. Пропускна спроможність оптоволоконно-

Рис. 17.9

го кабелю сягає 200 тисяч Мбіт/с. Має високий ступінь захисту від завад, однак досить дорогий порівняно з іншими типами кабелів.

Крім провідних каналів зв'язку в комп'ютерних мережах, зокрема глобальних, застосовують безпровідний зв'язок. У безпровідних мережах дані передають за допомогою: інфрачервоного випромінювання; лазерів; радіохвиль.

Інфрачервоне випромінювання має широкий діапазон частот, що дає змогу передавати сигнали зі швидкістю 10 Мбіт/с на відстань до 30 м.

Канали передавання *на основі лазерів* мають велику пропускну спроможність, але передавання може відбуватися тільки в межах прямої видимості.

Радіоканал схожий на канал передання звичайної радіостанції. Пряма видимість є необов'язковою, покривається велика площа.

Мережні пристрої. Функціонування комп'ютерної мережі забезпечують спеціальні пристрої оброблення інформації.

Мережний адаптер (мережна карта) — це пристрій для створення взаємодії (інтерфейсу) між комп'ютером і каналом передавання даних мережі, призначений для виконання таких функцій:

- підготовка даних, що надходять від комп'ютера, до передавання через канал зв'язку;
- керування потоком даних між комп'ютером і каналом зв'язку;
- перетворення інформації, що надходить із каналу зв'язку, у форму, прийнятну для роботи центрального процесора.

Плата мережного адаптера складається з апаратної частини та вбудованих програм, записаних у постійний пристрій пам'яті. Плати вставляють у слоти розширення комп'ютерів і серверів.

Інформація на адаптер надходить 16-розрядними і 32-розрядними шинами комп'ютера. Адаптер перетворює інформацію на послідовність бітів, далі *трансивер* перетворює біти інформації на електричні чи оптичні сигнали, для їх передавання каналами зв'язку.

Концентратор (англ. hub) — центральний пристрій мережі з зіркоподібною топологією, який пересилає пакети інформації між комп'ютерами. *Активні* концентратори підсилюють отриманий сигнал, а якщо сигнал без підсилення передається на комп'ютери, то такі концентратори називають *пасивними*.

Комутатор (англ. switch) — пристрій, який працює на канальному рівні в межах одного сегмента мережі й передає дані лише безпосередньому адресату.

Комутація в комп'ютерних мережах

Комплекс заходів, призначених для передавання інформації (даних) між абонентами мережі, й задіяні для цього апаратні і програмні засоби мережі називають *комутацією*. Іншими словами, комутація — це технологія передавання даних у комп'ютерних мережах. Цей процес здійснюється послідовно через кілька вузлів. Кожен вузол комутує, тобто під'єднує вхідний канал, яким надходить пакет даних, до одного з вихідних каналів. Комутація — основне функціональне призначення вузла. Залежно від методів комутації мережі по-

діляють на мережі з комутацією каналів, комутацією повідомлень і комутацією пакетів.

У мережах із *комутацією каналів* між передавачем і приймачем повідомлень заздалегідь, до початку передавання встановлюється фізичне нерозривне з'єднання, тобто утворюється наскрізний тракт із послідовно з'єднаних окремих фізичних каналів.

Перевагами способу комутації каналів є:

- забезпечення незначних затримок сигналу, що особливо важливо для передавання відео- і аудіоінформації;
- немає небезпеки виникнення перевантаження чи блокування лінії.

До недоліків цього способу належать:

- неефективне використання ресурсів мережі, оскільки під час перерв у передаванні (коли абонент обробляє інформацію, що надійшла, готує відповідь) жодну з ділянок тракту не може бути використано для передавання даних між іншими абонентами мережі;
- зниження пропускної спроможності мережі в цілому через монопольне виділення каналів зв'язку;
- порівняно великий час встановлення зв'язку між абонентами;
- пропускна спроможність всього тракту передавання визначається каналом із мінімальною пропускною спроможністю.

У мережах із *комутацією повідомлень* інформація передається у вигляді *повідомлень*, які містять заголовок і дані, за маршрутом, що визначається кожним вузлом мережі. Маршрут — це шлях руху повідомлення між вузлами. Для проходження повідомлень мережею дані мають бути оформлені у певному *форматі*. Формат повідомлення і процедуру його застосування визначають мережні протоколи. Між передавачем і приймачем повідомлень фізичний наскрізний тракт передавання не встановлюється, а передавач передає повідомлення в сусідній вузол комутації. Повідомлення приймається цим вузлом і розміщується у його пам'яті. Вузол комутації за адресою приймача, що міститься в заголовку, визначає маршрут, тобто наступний вузол, куди має бути передано повідомлення. Процес приймання, оброблення і передавання повідомлення повторюється послідовно всіма вузлами на маршруті від абонента-передавача до абонента-приймача (адресата) повідомлення.

Переваги цього способу:

- можливість використання на різних ділянках маршруту каналів зв'язку різної фізичної природи і з різною пропускною спроможністю;
- незалежні дії окремих вузлів мережі, наявність черг, можливість очікування під час ретрансляції забезпечує високий рівень використання каналів, а наявність буферної пам'яті вузла робить мережу тривкою до перевантажень;
- послідовне передавання повідомлень від вузла до вузла з ретрансляцією і контролем правильності на кожній ділянці зменшує ймовірність доставлення повідомлень не за адресою;

- можливість введення різноманітних категорій терміновості повідомлень і пов'язаний із цим пріоритет у використанні каналів.

Основним недоліком способу комутації повідомлень є неможливість ведення діалогу між абонентами у реальному часі.

Спосіб *комутації пакетів* передбачає поділ повідомлень на окремі частини — *пакети* (порції) однакового стандартного обсягу. Кожен пакет має власний заголовок, який містить номер пакета у повідомленні й адресу приймача повідомлень. Пакети передаються мережею незалежно один від одного, можливо навіть за різними маршрутами. Адресат (приймач) збирає повідомлення з пакетів, що надійшли мережею. Перевагою цього способу є:

- максимальне використання пропускної спроможності каналів за допомогою мультиплексування — розподілення часу роботи каналу для одночасного передавання кількох потоків даних;
- зменшення кількості помилок під час передавання внаслідок зменшення обсягу інформації за сеанс передавання;
- зменшення обсягу буферної пам'яті порівняно із комутацією повідомлень.

17.2. ГЛОБАЛЬНА КОМП'ЮТЕРНА МЕРЕЖА ІНТЕРНЕТ

Світова глобальна комп'ютерна мережа *Інтернет* — це об'єднання глобальних, регіональних і локальних мереж, окремих комп'ютерних систем і комп'ютерів, які використовують для обміну інформацією комплекс стандартних правил взаємодії (протоколів). Обсяг інформаційних ресурсів і кількість користувачів Інтернету є гігантськими й постійно збільшуються, розширюється і різноманітність послуг, що їх надає ця мережа. Зараз, за даними статистики, мережа Інтернет об'єднує понад 100 мільйонів користувачів у більш як 100 країнах світу, кожні 30 хвилин до Інтернету приєднується чергова комп'ютерна мережа.

Взаємодія об'єктів в Інтернеті

Як уже зазначено, Інтернет є системою, що об'єднує локальні, регіональні мережі, окремі комп'ютери. Всі ці об'єкти мають свої особливості й характеристики апаратних і програмних засобів.

Основним завданням, яке потрібно вирішити, створюючи такі неоднорідні системи, є забезпечення сумісності устаткування за електричними і механічними характеристиками і сумісності програмних засобів і даних за форматом, системою кодування тощо.

Розв'язання цієї проблеми у мережі Інтернет ґрунтується на *моделі відкритих систем (OSI),* згідно з якою стандартизація апаратури і програмного забезпечення здійснюється на підставі ієрархічної системи протоколів, тобто комплексу норм і правил взаємодії. У мережі Інтернет відповідно до моделі OSI можна використовувати до семи рівнів взаємодії.

Основою системи OSI в Інтернеті є комплекс протоколів TCP/IP. Основна властивість, яку забезпечує TCP/IP, — це можливість передавання даних від вузла до вузла в структурі, що складається з різнорідних елементів і постійно змінюється.

Протокол TCP (Transmission Control Protocol) — це протокол транспортного рівня, який визначає, як відбувається передавання інформації. Відповідно до протоколу TCP дані, які відправляються, поділяються на невеликі *пакети*, після чого до кожного пакета додаються адреси відправника та одержувача, а також відомості, необхідні для того, щоб правильно зібрати документ із пакетів на комп'ютері одержувача.

Протокол IP (Internet Protocol) — адресний протокол мережного рівня, що визначає, куди передається інформація. За протоколом IP здійснюють адресацію і маршрутизацію пакетів у мережі. Для цього в кожного користувача має бути своя унікальна адреса — IP-адреса, що слугує ідентифікатором комп'ютера споживача і складається з чотирьох чисел, розділених крапками. Чотири числа IP-адреси передаються чотирма байтами. На кожному вузлі мережі за чотирма числами IP-адреси визначається маршрут відправлення пакетів у конкретний момент, з урахуванням умов зв'язку, пропускної спроможності каналу, наявної черги. Тобто пакети навіть одного документа можуть передаватися незалежно один від одного різними маршрутами і каналами.

Незважаючи на те, що в мережі Інтернет використовують також протоколи інших рівнів, її називають TCP/IP-мережею, оскільки ці два протоколи є найважливішими.

Система адресації в Інтернеті

Для обміну інформацією в кожного комп'ютера в Інтернеті має бути своя адреса. Використовують два типи адрес: цифрові, або IP-адреси, і доменні (від англ. domain — «область, сфера»). Розглянемо структуру кожного з типів.

IP-адреса складається з чотирьох чисел, розділених крапками, наприклад **198.162.201.204.** Кожне з чисел займає один байт, тобто вісім двійкових розрядів, тому може мати значення від 1 до 255. Ліва частина IP-адреси — *мережний ідентифікатор* (англ. network ID), визначає конкретну мережу в Інтернеті. Права частина IP-адреси — *ідентифікатор комп'ютера* (англ. host ID), визначає конкретний комп'ютер у цій мережі. Залежно від того, скільки байтів у IP-адресі відведено під мережний ідентифікатор — один, два або три, розрізняють відповідно класи A, B, C IP-адрес.

IP-адреси використовують програмні засоби мережі для пересилання інформації між комп'ютерами. Однак для сприйняття користувачем IP-адреса є незручною. Людина краще сприймає і запам'ятовує слова, а не числа. Тому користувачі працюють із доменними адресами — унікальними іменами комп'ютерів у мережі. Доменна адреса, як і IP-адреса, складається з частин, розділених крапками. На відміну від IP-адреси, яка уточнює місце призначення зліва направо, доменна адреса робить це у зворотній послідовності —

справа наліво: спочатку йде ім'я комп'ютера, а потім ім'я мережі, у якій він розташований.

Щоб користувачам Інтернету було зручніше зв'язуватися один з одним, весь простір адрес поділено на області — домени (англ. domain). Можливим є також поділ за певними ознаками всередині доменів. Доменна адреса комп'ютера складається як мінімум із двох рівнів доменів:

Ім'я комп'ютера ... Домен другого рівня ● Домен першого рівня

Крайній праворуч — домен першого рівня, наступний ліворуч — домен другого рівня і так далі.

Домен першого рівня визначає країну або тип організації, якій належить комп'ютер. Існують скорочені до двох букв домени країн: Україна — *ua*, США — *us*, Франція — *fr* та ін. Домени організацій скорочують до трьох букв, наприклад університети та інші навчальні заклади — *edu*, урядові установи — *gov*, комерційні організації — *com* тощо.

Домен другого рівня визначає організацію, яка володіє або керує мережею, у якій розташований певний комп'ютер. Здебільшого ім'я цього домену збігається з назвою відповідної фірми або її торгової марки.

Ім'я комп'ютера вказує на конкретний комп'ютер у мережі, визначеної доменами першого і другого (а у деяких випадках наступних) рівнів. Воно реєструється тільки в цій мережі, й тільки ця мережа «відповідальна» за передавання інформації конкретному комп'ютеру-адресату.

Щоб мережний комп'ютер «зрозумів», куди треба передати повідомлення, доменну адресу має бути перетворено на IP-адресу. Таке перетворення здійснюється на спеціальних серверах мережі, де зберігаються таблиці відповідності між доменними адресами й IP-адресами, — *DNS-серверах* (від англ. domain name system — «система доменних імен»). DNS-сервери розміщені по всій мережі Інтернет. Кожен із них зберігає інформацію про велику кількість комп'ютерів. Якщо IP-адреса потрібного комп'ютера невідома DNS-серверу, він звертається до найближчого сусіднього DNS-сервера і так далі, доки не буде знайдено потрібну адресу. Адресу одного з DNS-серверів користувач має зазначити під час налаштування комп'ютера для роботи в Інтернеті.

17.3. СЛУЖБИ ІНТЕРНЕТУ

Користувач (клієнт) може скористатися різними послугами (сервісами), які надає йому Інтернет через свої служби. Розглянемо найвідоміші й найпоширеніші з них.

Всесвітня павутина

Всесвітня павутина, або *всесвітня мережа* (англ. WWW — World Wide Web), — це єдиний інформаційний простір, що складається з сотень мільйонів гіпертекстових електронних документів, які зберігаються на постійно під'єднаних до Інтернету комп'ютерах — WWW-серверах — у різних куточках

земної кулі. Для забезпечення взаємодії користувачів Інтернету з різних країн, які працюють на комп'ютерах різних типів, потрібно дотримуватися єдиних стандартів. Такі стандарти розробляє і впроваджує міжнародна організація — *Консорціум Всесвітньої павутини* (англ. World Wide Web Consortium, W3C).

Вебсторінка (англ. web page) — це окремий гіпертекстовий документ. Назва гіпертекстового документа має розширення **.html**, або **.htm**. Вебсторінка містить гіпертекст, тобто текстовий документ із гіперпосиланнями. Крім тексту на вебсторінці розміщуються таблиці, ілюстрації, мультимедійні та інші об'єкти. Група тематично об'єднаних сторінок утворюють *вебвузол*, або *вебсайт* (англ. website).

Гіпертекст (англ. hypertext) — це документ, у якому є *гіперпосилання* (англ. hyperlinks). Ще задовго до появи комп'ютерів автори у книжках і статтях користувалися *примітками* для пояснення певних термінів або понять, згаданих у тексті. Крім пояснення, і це дуже важливо, у примітці зазвичай наводять посилання на інші джерела (книжки, статті, дисертації тощо), де можна докладніше ознайомитися з певною темою. Гіперпосилання є подальшим розвитком цієї ідеї на новій технічній базі із застосуванням нових засобів.

Гіперпосилання реалізують так: слово або термін у гіпертекстовому документі виділяють кольором або формою шрифту. Із цим виділеним словом пов'язаний інший гіпертекстовий документ, який докладніше пояснює цей термін і може бути розміщений на сервері в іншій частині світу. Цей гіпертекстовий документ, своєю чергою, пов'язаний з іншим гіпертекстовим документом через гіперпосилання. Тобто у гіпертекстовому документі немає ні початку, ні кінця. Гіпертекстові документи утворюють єдину взаємопов'язану систему, яка нагадує павутину, що покриває весь світ.

Мова HTML (Hyper Text Markup Language) належить до декларативних мов програмування, тобто вона лише описує, як має відображатися на пристроях гіпертекстовий документ. Цей опис здійснюється за допомогою спеціальних службових слів — *дескрипторів* (англ. descriptor — «описувач») або *тегів* (англ. tag — «мітка, ярлик»). Щоб відрізнити теги від гіпертексту, їх беруть у кутові дужки < >.

Крім тексту на вебсторінках можуть розміщуватися графічні зображення (картини, фотографії), звукова та відеоінформація, анімаційні об'єкти тощо. Посилання також можуть бути на інформацію будь-якого виду.

Редактори HTML — це спеціальні засоби для створення гіпертекстових документів. Редактори HTML поділяють на два типи: текстові й візуальні.

Для створення вебсторінок використовують найпростіші текстові редактори Notepad, AkelPad та інші, але для цього потрібно досконало знати мову HTML. Текстові редактори дають змогу створювати компактний код, тому професіонали зазвичай працюють із такими редакторами.

Для роботи з візуальними редакторами не потрібно знати мову HTML, вони мають розвинуті засоби для полегшення процесу створення сайтів. Візуальні редактори працюють за принципом WYSIWYG (what you see is what

you get — «що бачиш — те й отримаєш»). До найпоширеніших візуальних редакторів належать Macromedia Dreamweaver MX і Microsoft FrontPage.

У всіх ресурсів Інтернету, зокрема у вебсторінок, є своя **адреса**, яку задають за допомогою URL (Uniform Resource Locator — уніфікований локатор (визначник положення) ресурсів) — стандарту, прийнятого в Інтернеті для визначення місця знаходження будь-якого ресурсу. URL складається з трьох частин:

Схема описує протокол прикладної програми. Найчастіше використовують протокол передавання гіпертекстових документів *http*. За назвою протоколу ставляться символи ://. *Хост* — це доменне ім'я комп'ютера, на якому розташований ресурс. *Шлях* — це повний маршрут до документа і, можливо, його ім'я.

Браузери (від англ. to browse — «переглядати») — спеціальні програми з графічним інтерфейсом, за допомогою яких переглядають вебсторінки і переходять від одного до іншого документа через гіперпосилання. На початку розвитку всесвітньої павутини було суперництво між двома браузерами: Internet Explorer (від англ. explorer — «дослідник») фірми Microsoft і Netscape Navigator (від англ. navigator — «навігатор») фірми Netscape. На сучасному етапі розвитку широко використовують браузери Google Chrome, Mozilla Firefox, Opera.

Щоб орієнтуватися в гігантських потоках інформації, які надходять до користувача через браузери, і вибрати потрібну, створено спеціальні *пошукові програми,* які розташовані на пошукових серверах. Отримати доступ до них просто, достатньо знати їхній URL.

Протокол FTP (File Transfer Protocol) — це протокол передавання файлів, одна з перших послуг Інтернету. Ця служба дає змогу надсилати й отримувати текстові файли і файли з інформацією у вигляді двійкових чисел будь-яким клієнтам Інтернету. Встановивши зв'язок із віддаленим комп'ютером, користувач може скопіювати довільний файл, що міститься на цьому комп'ютері, на свій комп'ютер, або зі свого на віддалений комп'ютер.

Gopher — це удосконалена система пересилання файлів. На відміну від FTP, вона працює з файлами, що розміщені на численних комп'ютерах, а також дає змогу за допомогою меню не тільки переглянути списки ресурсів, а й пересилати потрібний матеріал, причому знати, де він міститься, необов'язково. Gopher — це одна з найбільш повних систем перегляду, інтегрована з іншими системами (FTP, Telnet). Мережні засоби за допомогою розподілених індексів пов'язані в єдину систему — Gopherspace (Gopher-простір). Доступ до ресурсів Gopher-простору здійснюють за допомогою запропонованих меню, а пошук — за допомогою спеціальних пошукових програм.

Електронна пошта (E-mail) — одна з найпопулярніших і найпоширеніших послуг Інтернету. Потоки інформації в мережі складаються, в основному, з електронної кореспонденції, і багато людей в Інтернеті користуються здебільшого саме цією послугою. В Інтернеті кожне поштове повідомлення має дві частини: *заголовок* і *тіло*. Заголовок повідомлення містить службову інформацію: адресу відправника, адресу отримувача, дату і час відправки тощо. Тіло повідомлення містить основний текст.

Поштова адреса в Інтернеті складається з двох частин, розділених знаком @ : ім'я користувача @ ім'я комп'ютера. Існує багато прикладних програм, що реалізують поштову службу в Інтернеті: Mail, Eudora, Microsoft Exchange, Outlook Express та ін.

Mail Lists (списки розсилки) створено на основі протоколу електронної пошти. Підписавшись (безкоштовно) на цю послугу, можна регулярно отримувати електронною поштою повідомлення на певні теми: науково-технічні й економічні огляди, презентації нових апаратних і програмних засобів тощо.

Мережний протокол передавання новин (Network News Transfer Protocol, NNTP) розроблений із метою поширювати й отримувати статті з новинами. **UseNet** — одна зі сфер застосування NNTP. Цей сервіс слугує для обміну інформацією у формі статей, повідомлень, оголошень тощо. UseNet можна порівняти з електронною дошкою оголошень, на якій будь-який учасник може розмістити своє повідомлення.

Новини поділяють за темами на *групи новин,* або *телеконференції*. Вони працюють 24 години на добу, 365 днів на рік. Щоб отримати доступ до телеконференцій, потрібно мати на комп'ютері відповідну програму і передплатити конференції, які зацікавили.

Послуга Telnet (віддалений доступ) дає змогу одному комп'ютеру, під'єднаному до Інтернету, отримати віддалений доступ до іншого комп'ютера, також під'єднаного до Інтернету. Службу Telnet використовують, здебільшого, для доступу до баз даних, відкритих для загалу: каталогів бібліотек і електронних дошок оголошень. Часто за допомогою цієї служби дистанційно керують складними технічними об'єктами: телескопами, відеокамерами, промисловими роботами. У минулому цю послугу широко використовували для проведення складних математичних розрахунків і обчислень на віддаленому суперкомп'ютері.

Послуга IRC (Internet Relay Chat — трансляція бесід в Інтернеті) дає змогу спілкуватися з реальними людьми в реальному часі. Розмови (англ. chat — «дружня розмова, бесіда») відбуваються в так званих кімнатах (англ. room) або каналах (англ. channel), які можуть бути як відкритими (англ. public), так і закритими (англ. private). Кожна «кімната» має свою тему. Для участі в розмові потрібно під'єднатися до комп'ютера, на якому виконується серверна програма IRC. Сервери з'єднані між собою і утворюють кілька окремих мереж. Розмову можна вести тільки між людьми, які під'єднані до тієї самої мережі. Для участі в таких розмовах потрібно встановити на комп'ютері одну з програм — клієнтів IRC: Netscape Chat, mIRC, Microsoft Chat тощо.

17.4. ХМАРНІ КОМП'ЮТЕРНІ ТЕХНОЛОГІЇ

Хмарні комп'ютерні технології (англ. cloud computing) — це один із новітніх напрямів інформаційних технологій, який забезпечує новий рівень надання інформаційних послуг через Інтернет. За допомогою хмарних комп'ютерних технологій у будь-якому місці, у будь-який час є можливість отримати інформаційні ресурси і послуги: обчислювальні ресурси, прикладні програми, інтерфейси, пам'ять для зберігання будь-яких обсягів інформації (рис. 17.10).

Рис. 17.10

Термін «хмарні» означає, що деталі надання послуг приховані від користувача. Хмара — це інформаційні ресурси спільного користування: апаратне і програмне забезпечення, інтерфейси, платформи, мережі, об'єднані глобальною мережею Інтернет.

Хмарні технології звільняють користувача від необхідності розбиратися у деталях технології оброблення інформації, розподіляти ресурси між задачами, планувати оптимальне завантаження устаткування. Надіслану в хмару задачу може бути виконано на сервері, що розташований на іншому боці планети, в іншій країні, а користувач може і не здогадуватися про це.

Хмарні технології дають змогу орендувати через Інтернет обчислювальні потужності, дисковий простір пам'яті та інші ресурси, замість їх купівлі. Перевага такого підходу в тому, що користувач платить лише за ті послуги, які йому потрібні у певний час для розв'язання конкретної задачі (pay-as-you-go).

Національний інститут стандартів і технологій США встановив такі обов'язкові характеристики хмарних обчислень:

- самообслуговування на вимогу (англ. on demand self-service). Споживач самостійно визначає і змінює потреби в ресурсах, як-от тривалість доступу, обсяг пам'яті, швидкодія без узгодження з представником постачальника послуг;
- широкий доступ до мережі (англ. broad network access) — можливість доступу до мережі через стандартні механізми за допомогою різнома-

нітних пристроїв: мобільних телефонів, ноутбуків і нетбуків, персональних комп'ютерів, робочих станцій;

- об'єднання ресурсів (англ. resource pooling). Ресурси провайдера об'єднують для обслуговування кількох користувачів із використанням багатокористувацької моделі. Різноманітні фізичні й віртуальні ресурси динамічно призначають і перепризначають відповідно до потреб користувачів;
- гнучкість (англ. elasticity). Послуги можуть бути надані і відкликані, розширені і звужені в будь-який момент часу, без додаткових витрат на взаємодію з постачальником, як правило, в автоматичному режимі;
- облік спожитих ресурсів (англ. measured service). У хмарних технологіях здійснюється автоматичне керування і оптимізація ресурсів за допомогою обліку параметрів наданого ресурсу. Це забезпечує прозорість як для користувача, так і для постачальника.

У хмарних технологіях реалізуються декілька **моделей хмарних комп'ютерних технологій**.

Програмне забезпечення як послуга SaaS (англ. Software-as-a-Service) — користувач має можливість використовувати прикладне програмне забезпечення і бази даних, які доступні з різноманітних клієнтських пристроїв. Хмарні провайдери керують інфраструктурою і платформами, на яких працюють програми користувача.

Платформа як послуга PaaS (англ. Platform-as-a-Service) — модель, коли користувачеві надається можливість використовувати хмарну інфраструктуру, до складу якої входять операційні системи, інструментальні засоби створення, тестування та виконання прикладного програмного забезпечення, системи управління базами даних. У окремих хмарних технологіях надані ресурси масштабуються автоматично, звільняючи користувача від потреби розподіляти ресурси вручну.

Інфраструктура як послуга IaaS (англ. Infrastructure-as-a-Service) — можливість користуватися хмарною інфраструктурою: фізичними і, здебільшого, віртуальними машинами, серверами, пам'яттю, мережами тощо. IaaS-провайдери надають ці ресурси за вимогою (англ. on-demand) зі своїх великих сховищ (англ. pools), встановлених у центрах оброблення даних. Для під'єднання до хмари на великій території користувачі можуть скористатися або Інтернетом, або віртуальними локальними мережами.

Існує кілька **моделей розгортання** хмарної інфраструктури. Розглянемо особливості найпоширеніших з-поміж них.

Приватна хмарна (англ. private cloud) — це хмарна інфраструктура, яка призначена для використання лише однією організацією, що охоплює декілька користувачів (наприклад, підрозділів). Приватна хмара може перебувати у власності, керуванні та експлуатації як самої організації, так і третьої сторони (або деякої їх комбінації). Така хмара може фізично перебувати як в юрисдикції власника, так і поза нею.

Публічна хмара (англ. public cloud) — хмарна інфраструктура для вільного використання широким загалом. Публічна хмара може перебувати у власності, керуванні та експлуатації комерційних, академічних (освітніх і наукових)

або державних організацій (чи будь-якої їх комбінації). Публічна хмара перебуває в юрисдикції постачальника хмарних послуг.

Громадська хмара (англ. community cloud) — хмарна інфраструктура, яка призначена для використання конкретною спільнотою споживачів із організацій, що мають спільні цілі (наприклад, місію, вимоги щодо безпеки, політику та відповідність різноманітним вимогам). Громадська хмара може перебувати в спільній власності, керуванні та експлуатації однієї чи більше організацій зі спільноти або третьої сторони (чи деякої їх комбінації). Така хмара може фізично бути як у, так і поза юрисдикцією власника.

Гібридна хмара (англ. hybrid cloud) складається з двох або більше різнорідних хмарних інфраструктур (приватних, громадських, публічних), які залишаються унікальними сутностями, але з'єднані між собою стандартизованими або приватними технологіями, що дає змогу переносити ці прикладні програми (наприклад, використання ресурсів публічної хмари для балансування навантаження між хмарами).

17.5. ІНТЕРНЕТ РЕЧЕЙ

Інтернет речей (англ. Internet of Things, IoT) — сучасна інформаційна технологія взаємодії через Інтернет технічних пристроїв (речей — англ. things) без участі людини. Такі технічні пристрої сприймають інформацію з навколишнього середовища за допомогою вбудованих датчиків (сенсорів), обробляють цю інформацію і передають її через Інтернет на виконавчі пристрої. Виконавчі пристрої, сприйнявши інформацію, відповідно до закладених у них програм, виконують фізичну дію з об'єктами середовища. Наприклад, дрон із датчиками, пролітаючи над полем, вимірює вологість ґрунту, температуру та інші параметри, обробляє отриману від датчиків інформацію і дає команди через Інтернет сільськогосподарським машинам і механізмам обробляти певні ділянки поля. Крім інформації, отриманої від датчиків, враховують дані короткострокового і довгострокового прогнозів.

Сфери застосування

«Розумна оселя» (англ. smart home) — ця концепція полягає в інтеграції системи датчиків температури, вологості, освітлення, присутності, охоронної системи, системи обігріву житла і кондиціювання повітря в єдину систему під управлінням центрального комп'ютера. Така система підтримує кліматичні характеристики житла, вмикає і вимикає побутові пристрої, відслідковує систему безпеки без участі людини.

«Розумна охорона здоров'я». Медичні пристрої, призначені для вимірювання артеріального тиску, серцевого ритму, а також кардіостимулятори, слухові апарати тощо, можна використовувати для здійснення дистанційного моніторингу стану здоров'я та систем аварійного сповіщення.

У *сільському господарстві* застосовують технології Інтернету речей, як-от збирання даних про температуру, кількість опадів, вологість, швидкість вітру, зараження шкідниками та структуру ґрунту. Ці дані використовують для автоматизації

агротехніки, прийняття обґрунтованих рішень щодо поліпшення якості, мінімізації ризику та відходів і зменшення зусиль, необхідних для управління посівами.

Для *моніторингу навколишнього середовища,* наприклад, використовують датчики, які відстежують якість повітря і води, стан атмосфери і ґрунту. За допомогою таких датчиків здійснюють моніторинг місцезнаходження і переміщення об'єктів дикої природи.

Розроблення пристроїв з обмеженим споживанням енергії, під'єднаних до Інтернету, дає змогу реалізувати системи раннього попередження про землетруси або цунамі. Ця інформація стає в пригоді аварійним службам для надання ефективнішої допомоги.

Технології Інтернету речей

Засоби ідентифікації. Кожен об'єкт фізичного світу, який бере участь в Інтернеті речей, нехай навіть не під'єднаний до мережі, повинен мати унікальний ідентифікатор. Для автоматичної ідентифікації предметів використовують різноманітні системи: радіочастотну (до кожного об'єкта прикріплюють радіочастотну мітку), оптичну (штрих-коди, Data Matrix, QR-коди), інфрачервоні мітки тощо. Однак для забезпечення унікальності ідентифікаторів різних типів доводиться проводити роботу з їх стандартизації.

Засоби вимірювання. Завдання цих засобів — забезпечити перетворення інформації про зовнішнє середовище на дані, придатні для передавання їх засобами оброблення. Це можуть бути як окремі датчики температури, освітленості та ін., так і складні вимірювальні комплекси. Для досягнення автономності засобів вимірювання бажано забезпечити електроживлення датчиків за допомогою альтернативної енергетики (сонячні батареї тощо), щоб не витрачати час і кошти на підзаряджання акумуляторів або заміну батарей.

Засоби передавання даних. Для передавання даних можна використовувати будь-яку з наявних технологій. У разі застосування бездротових мереж особливу увагу приділяють підвищенню надійності передавання даних. Активно використовують технологію передавання даних лініями електропересилання, оскільки багато «речей» (як-от торгові автомати, банкомати тощо) під'єднані до електромереж.

Засоби оброблення даних. На сучасному етапі розвитку до Інтернету під'єднано десятки мільярдів пристроїв, які генерують десятки мільярдів терабайтів даних. Вважають, що головна частина Інтернету речей — це не датчики і засоби передавання даних, а хмарні системи, що забезпечують високу пропускну спроможність і здатні швидко реагувати на певні ситуації, зокрема екстремальні й аварійні. Допоможуть впоратися з величезними потоками інформації також туманні обчислення, які не конкуруватимуть із хмарними, а ефективно їх доповнюватимуть.

Виконавчі пристрої здатні перетворювати цифрові електричні сигнали, що надходять від інформаційних мереж, на дії. Наприклад, для того, щоб через смартфон можна було ввімкнути систему опалення в будинку, у ній має бути відповідний пристрій. Виконавчі пристрої часто конструктивно поєднані з датчиками.

17.6. ШТУЧНИЙ ІНТЕЛЕКТ

На сучасному етапі розвитку інформатики загальноприйнятого визначення поняття *«штучний інтелект»* (англ. artificial intelligence, AI) немає. Зазвичай штучний інтелект трактують як здатність комп'ютерних систем розв'язувати проблеми, які до цього до снаги були лише людині. Під терміном *інтелектуальний агент* розуміють розумні програми, що спостерігають за навколишнім середовищем і діють у ньому раціонально, тобто здатні до розуміння умов і на основі цього виконують відповідні дії, щоб досягнути певної мети. Про інтелектуальність агента можна говорити, якщо він взаємодіє з навколишнім середовищем приблизно так само, як діяла би людина.

В аспекті програм штучного інтелекту часто вживають термін *бот* (англ. bot, від robot — «робот») — це програма, що виконує автоматично або за заданим сценарієм певні дії і має в цьому якусь подібність із людиною.

Сфери застосування штучного інтелекту

Інформаційні технології штучного інтелекту бурхливо розвиваються, тому інформація про сфери їх застосування повсякчас оновлюється. Крім того, технології Інтернету речей тісно пов'язані зі штучним інтелектом: переважна більшість технічних пристроїв, пов'язаних між собою через Інтернет, оснащені програмами штучного інтелекту.

Розглянемо найпоширеніші сфери застосування штучного інтелекту.

Застосування методів штучного інтелекту в *промисловості* дає змогу значно скоротити частку ручної праці, зменшити терміни виготовлення, збільшити ефективність виробництва. Застосування інтелектуальних роботів допомагає реалізувати безлюдні виробництва, що особливо важливо, якщо йдеться про агресивні середовища.

Застосування методів штучного інтелекту в *медицині й охороні здоров'я,* зокрема для ранньої діагностики захворювань, дає змогу зберегти здоров'я багатьом пацієнтам. Постійний дистанційний моніторинг стану здоров'я за допомогою інтелектуальних датчиків значно збільшує можливості системи охорони здоров'я і заощаджує значні кошти.

Аналіз природних мов методами штучного інтелекту здійснюють за такими напрямами: машинний переклад, тобто переклад текстів з однієї природної мови на іншу за допомогою програм штучного інтелекту; забезпечення діалогу (інтерфейсу) між людиною і комп'ютером природною мовою; створення довідкових систем «запитання–відповідь», у яких комп'ютер відповідає людині природною мовою.

Розпізнавання образів (англ. pattern recognition) — застосування методів штучного інтелекту з метою класифікації та ідентифікації предметів, явищ, процесів, сигналів, ситуацій тощо, які мають кінцевий набір деяких властивостей і ознак. Першим етапом є вилучення найважливіших ознак об'єкта (англ. feature extraction), відтак здійснюють класифікацію та ідентифікацію об'єктів. Найчастіше застосовують розпізнавання таких образів: тексту, зокре-

ма рукописного; людських облич, що має важливе значення в охоронних системах; певних сигналів, наприклад звукових сигналів роботи серця, що дає змогу діагностувати серцеві захворювання; звуків природної мови; автомобільних номерів для автоматизованого керування руху транспортних засобів на дорогах; штрих-кодів у торгівлі для роботи торгових автоматів.

ТЕСТОВІ ЗАВДАННЯ

ОБ'ЄКТИ ІНТЕРНЕТУ ВЗАЄМОДІЮТЬ МІЖ СОБОЮ ЗГІДНО З ПРОТОКОЛОМ:

1. TCR/IP
2. TCP/IP
3. TCP/IR
4. TCP/TP

ДОПУСТИМОЮ IP-АДРЕСОЮ В ІНТЕРНЕТІ Є АДРЕСА:

1. 254.180.193.5
2. 253.180.193.6
3. 257.180.193.7
4. 251.180.193.8
5. 254.180.193.25

СПИСОК ВИКОРИСТАНОЇ ТА РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Абрамов В. О., Клименко С. Ю. Базові технології комп'ютерних мереж : навч. посіб. Київ : Київський університет імені Б. Грінченка, 2011. 234 с.
2. Бондаренко О. О., Ластовецький В. В., Пилипчук О. П., Шестопалов Є. А. Інформатика (рівень стандарту) : підруч. для 10 (11) класу закладів заг. сер. освіти. Харків : Ранок, 2018. 104 с.
3. Веселовська Г. В., Ходаков В. Є., Веселовський В. М. Комп'ютерна графіка : навч. посіб. для студентів ВНЗ. Херсон : Олді-Плюс, 2017. 581 с.
4. Городецька О. С., Гикавий В. А., Онищук О. В. Комп'ютерні мережі : навч. посіб. Вінниця : Вінницький національний технічний університет, 2017. 129 с.
5. Гуржій А. М., Карташова Л. А., Лапінський В. В., Руденко В. Д. Інформатика : підруч. для 7 кл. загальноосв. навч. закладів. Львів : Світ, 2015. 176 с.
6. Гуржій А. М., Коряк С. Ф., Самсонов В. В., Скляров О. Я. Архітектура, принципи функціонування та керування ресурсами IBM PC : навч. посіб. Харків : ВПК «Глобус», 2003. 511 с.
7. Гуржій А. М., Поворознюк Н. І., Самсонов В. В. Інформатика та інформаційні технології : підруч. для учнів проф.-тех. навч. закладів. Харків : Компанія СМІТ, 2003. 352 с.
8. Злобін Г. Г., Рикалюк Р. Є. Архітектура та апаратне забезпечення ПЕОМ : навч. посіб. для студентів ВНЗ. Київ : Каравела, 2016. 223 с.
9. Караванова Т. П. Інформатика: основи алгоритмізації та програмування. 777 задач з рекомендаціями та прикладами : навч. посіб. для 8–9 класів із поглибленим вивченням інформатики. Київ : Генеза, 2009. 286 с.
10. Кащеєв Л. Б., Коваленко С. В. Інформатика. Основи комп'ютерної графіки : навч. посіб. Харків : Ранок, 2011. 160 с.
11. Костриба О. В. Усі уроки інформатики. 9 клас. Харків : Вид. група «Основа», 2009. 191 с.
12. Кулаков Ю. О., Луцький Г. М. Комп'ютерні мережі. Київ : Юніор, 2005. 397 с.
13. Матвієнко М. П., Розен В. П., Закладний О. М. Архітектура комп'ютера : навч. посіб. для студентів вищ. навч. закладів. Київ : Ліра-К, 2016. 263 с.