Лабораторна робота №1.

Знайомство з С++.Виконання програми простої структури. Використання основних операторів мови С++

Ціль:Знайомство з середовищем програмування, створення, налагодження та виконання простої програми, що містить введення/виведення інформації та найпростіші обчислення. Отримання навичок у виборі та використанні операторів С++; знайомство з ітераційними процесами. Робота складається із двох частин.

Робота складається з двох частин:

Перша

Друга

Приклад рішення

1 частина роботи. Постановка задачі

1 завдання- обчислити значення виразу при різних речових типах даних (float та double). Обчислення слід виконувати двома способами:

- з використанням проміжних змінних для поелементних дій та
- без використання проміжних змінних (записавши весь вираз одним оператором)

Порівняти та пояснити отримані результати.

2 завдання- Обчислити значення виразів. Обчислення потрібно виконати двома способами:

- з використанням проміжних змінних для поелементних дій та
- без використання проміжних змінних (записавши весь вираз одним оператором)

Порівняти та пояснити отримані результати.

Варіанти

№	Завдання 1	Завдання 2
1	$\frac{(a+b)^2-(a^2+2ab)}{b^2}$,	1) n+++m 2) m-- >n

	за a=1000, b=0.0001	3) n-- >m
2	$\frac{(a-b)^2-(a^2-2ab)}{b^2}$, за a=1000, b=0.0001	1) ++n*++m 2) m++<n 3) n++>m
3	$\frac{(a+b)^3-(a^3+3a^2b)}{3ab^2+b^3}$, за a=1000, b=0.0001	1) n---m 2) m--<n 3) n++>m
4	$\frac{(a+b)^3-(a^3)}{3ab^2+b^3+3a^2b}$, за a=1000, b=0.0001	1) n++*m 2) n++<m 3) m-- >m
5	$\frac{(a-b)^3-(a^3-3a^2b)}{b^3-3ab^2}$, за a=1000, b=0.0001	1) - -m-++n 2) m*n<n++ 3) n-- > m++
6	$\frac{(a-b)^3-(a^3-3ab^2)}{b^3-3a^2b}$, за a=1000, b=0.0001	1) m-++n 2) ++m>--n 3) --n<++m
7	$\frac{(a-b)^3-(a^3)}{b^3-3ab^2-3a^2b}$, за a=1000, b=0.0001	1) m+--n 2) m++<++n 3) n--< --m
8	$\frac{(a+b)^4-(a^4+4a^3b+6a^2b^2)}{4ab^3+b^4}$, за a=100, b=0.001	1) n++-m 2) m-- >n 3) n-- >m
9	$\frac{(a+b)^4-(a^4+4a^3b)}{6a^2b^2+4ab^3+b^4}$, за a=100, b=0.001	1) ++n*++m 2) m++<n 3) n++>m
10	$\frac{(a-b)^4-(a^4-4a^3b+6a^2b^2)}{b^4-4ab^3}$,	1) n---m 2) m--<n

	за a=100, b=0.001	3) n++>m
11	$\frac{(a-b)^4-(a^4-4a^3b)}{6a^2b^2-4ab^3+b^4}$, за a=100, b=0.001	1) n++*m 2) n++<m 3) m-- >m
12	$\frac{(a+b)^2-(a^2+2ab)}{b^2}$, за a=1000, b=0.0001	1) - -m-++n 2) m*n<n++ 3) n-- > m++
13	$\frac{(a-b)^2-(a^2-2ab)}{b^2}$, за a=1000, b=0.0001	1) m-++n 2) ++m>--n 3) --n<++m
14	$\frac{(a+b)^3-(a^3+3a^2b)}{3ab^2+b^3}$, за a=1000, b=0.0001	1) m+--n 2) m++<++n 3) n--< --m
15	$\frac{(a+b)^3-(a^3)}{3ab^2+b^3+3a^2b}$, за a=1000, b=0.0001	1) n++-m 2) m-- >n 3) n-- >m
16	$\frac{(a-b)^3-(a^3-3a^2b)}{b^3-3ab^2}$, за a=1000, b=0.0001	1) ++n*++m 2) m++<n 3) n++>m
17	$\frac{(a-b)^3-(a^3-3ab^2)}{b^3-3a^2b}$, за a=1000, b=0.0001	1) n---m 2) m--<n 3) n++>m
18	$\frac{(a-b)^3-(a^3)}{b^3-3ab^2-3a^2b}$, за a=1000, b=0.0001	1) n++*m 2) n++<m 3) m-->m
19	$\frac{(a+b)^4-(a^4+4a^3b+6a^2b^2)}{4ab^3+b^4}$,	1) --m-++n 2) m*n<n++

	за a=100, b=0.001	3) n-->m++
20	$\frac{(a+b)^4-(a^4+4a^3b)}{6a^2b^2+4ab^3+b^4}$, за a=100, b=0.001	1) m-++n 2) ++m>--n 3) --n<++m
21	$\frac{(a-b)^4-(a^4-4a^3b+6a^2b^2)}{b^4-4ab^3}$, за a=100, b=0.001	1) n++-m 2) m-- >n 3) n-- >m
22	$\frac{(a-b)^4-(a^4-4a^3b)}{6a^2b^2-4ab^3+b^4}$, за a=100, b=0.001	1) ++n*++m 2) m++<n 3) n++>m
23	$\frac{(a+b)^3-(a^3+3a^2b)}{3ab^2+b^3}$, за a=1000, b=0.0001	1) n---m 2) m--<n 3) n++>m
24	$\frac{(a+b)^3-(a^3)}{3ab^2+b^3+3a^2b}$, за a=1000, b=0.0001	1) n++*m 2) n++<m 3) m-- >m
25	$\frac{(a-b)^3-(a^3-3a^2b)}{b^3-3ab^2}$, за a=1000, b=0.0001	1) - -m-++n 2) m*n<n++ 3) n-- > m++

Методичні вказівки

1. Для введення та виведення даних використовувати операції >> та << і стандартні потоки cin та cout. Або використовувати функції scanf та printf (#include < iostream.h > або <stdio.h>).
2. Для обчислення ступеня можна використовувати функцію pow(x,y) із бібліотечного файлу math.h.
3. При виконанні завдання 1 треба використовувати допоміжні змінні зберігання проміжних результатів. Наприклад: c=pow(a,3);d=3*a*a*b;e=3*a*b*b;f=pow(b,3);

4. Для виконання 2-го завдання рекомендується “розставити дужки” у розрахованому виразі та з'ясувати тип одержуваного результату.

Короткі теоретичні відомості наведено**тут**.

Зміст звіту

1. Постановка задачі.
2. Програма вирішення завдання1.
3. Результати роботи програми даних типу float.
4. Результати роботи програми даних типу double.
5. Пояснення результатів.
6. Програма вирішення завдання2.
7. Результати роботи програми.
8. Пояснення результатів.

2 частина роботи. Постановка задачі

Постановка задачі

Використовуючи оператор циклу, знайти суму елементів, зазначених у конкретному варіанті. Результат надрукувати, забезпечивши відповідним заголовком.

Варіанти

1) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{5^n}{n^n}$$

2) Знайти суму з точністюε=10-6, загальний член якого

$$a_n = \frac{6^n}{n^n}$$

3) Знайти суму 100 членів ряду, в якому

$$a_n = e^{-\sqrt{n}} + \sqrt{n}$$

4) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{2^n}{n^n}$$

5) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{(-1)^{n-1}}{n^n}$$

6) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{1}{2^n} + \frac{1}{3^n}$$

7) Знайти суму ряду з точністю ε=10-6, загальний член якого

$$a_n = \frac{n!}{(6n)!}$$

8) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{(2n-1)}{2^n}$$

9) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{10^n}{n!}$$

10) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{n!}{(2n)!}$$

11) Знайти суму з точністюε=10-6, загальний член якого

$$a_n = \frac{n!}{2^{n^2}}$$

12) Знайти суму з точністюε=10-6, загальний член якого

$$a_n = \frac{2^n n!}{n^n}$$

13) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{3^n n!}{(3n)!}$$

14) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{n!}{3n^n}$$

15) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{(n!)^2}{2^{n^2}}$$

16) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \lg(n!)e^{-n\sqrt{n}}$$

17) Знайти суму з точністюε=10-6, загальний член якого

$$a_n = 10^{-n}(n-1)!$$

18) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{n^3}{(3n-3)!}$$

19) Знайти суму ряду з точністюε=10-6, загальний член якого

$$a_n = \frac{n}{(n-1)^2}$$

20) Знайти суму з точністюε=10-6, загальний член якого

$$a_n = e^n \cdot 100^{-n^2}$$

21) Знайти суму 13 членів ряду, в якому

$$a_n = \frac{\ln(n!)}{n^2}$$

22) Знайти суму 15 членів ряду, в якому

$$a_n = \frac{n^{\ln n}}{(\ln n)^n}$$

23) Знайти суму 10 членів ряду, в якому

$$a_n = \frac{n!}{n^{\sqrt{n}}}$$

24) Знайти суму 9 членів ряду, в якому

$$a_n = e^{-\sqrt{n}}$$

25) Знайти суму 7 членів ряду, в якому

$$a_n = n^2 e^{-\sqrt{n}}$$

Зміст звіту

1. Постановка задачі.
2. Текст програми.
3. Результат розв'язання конкретного варіанта.

Методичні вказівки

1. При визначенні суми членів ряду слід використовувати рекурентну формулу для отримання наступного члена ряду.

Наприклад, потрібно знайти суму ряду з точністю ε=10-6, загальний член якого $a_n = \frac{2(n!)^2}{(3(2n)!)!}$.
Для отримання рекурентної формули обчислимо відношення:

$$\frac{a_{n+1}}{a_n} = \frac{2((n+1)!)^2 \cdot 3(2n)!}{3(2n+2)! \cdot 2(n!)^2} = \frac{n+1}{2(2n+1)},$$

звідки:

$$a_{n+1} = a_n \cdot \frac{(n+1)}{2(2n+1)}.$$

2. При складанні програми вважати, що точності досягнуто, якщо an <ε

Короткі теоретичні відомості наведено **тут**.

---

1. Короткі теоретичні відомості

Мова Сі створена 1972 р.Деннісом Рітчі під час розробки ОС Unix. Він проектувався як інструмент системного програмування з орієнтацією розробки добре структурованих програм. Таким чином він поєднує в собі, з одного боку, засоби мови програмування високого рівня: опис типів даних, оператори for, while, if і т. д., а з іншого боку, містить засоби мови типу Ассемблер: регістрові змінні, адресну арифметику, можливість роботи з полями біт і т. д.

1.1. Структура програми

Програма мовою Сі має таку структуру:

```
#директиви препроцесора
. . . . . .
#директиви препроцесора
функція а ( )
     оператори
функція в ( )
     оператори
void main () // функція, з якої починається виконання програми
     оператори
          описи
          присвоєння
          функція
          порожній оператор
               складовий
               вибору
               циклів
               переходу
```

Директиви препроцесора – керують перетворенням тексту програми до її компіляції.Вихідна програма, підготовлена мовою Сі як текстового файла проходить 3 етапи обробки:

1) препроцесорне перетворення тексту;
2) компіляція;
3) компонування (редагування зв'язків чи складання).
Після цих 3 етапів формується машинний код програми, що виконується.

Завдання препроцесора – перетворення тексту програми до її компіляції. Правила препроцесорної обробки визначає програміст за допомогою директив препроцесора. Директива починаєтьсяз #. Наприклад,
1) #define – вказує правила заміни у тексті.

#define ZERO 0.0

Означає, що кожне використання у програмі імені ZERO замінюватиметься на 0.0.
2) #include< ім'я файлу заголовка> - призначена для включення в текст програми тексту з каталогу «Заголовних файлів», що поставляються разом зі стандартними бібліотеками. Кожна бібліотечна функція Сі має відповідний опис в одному із заголовних файлів. Список заголовних файлів визначено стандартом мови. Вживання директиви include не підключає відповідну стандартну бібліотеку, а лише дозволяють вставити в текст програми опис із зазначеного заголовного файлу. Підключення кодів бібліотеки здійснюється на етапі компонування, тобто після компіляції. Хоча у заголовних файлах містяться всі описи стандартних функцій, код програми включаються тільки ті функції, які використовуються в програмі.

Після виконання препроцессорної обробки тексті програми залишається жодної препроцессорной директиви. Програма є набір описів і визначень, і складається з набору функцій. Серед цих функцій завжди має бути функція з ім'ямmain. Без неї програма не може бути виконана. Перед ім'ям функції містяться відомості про тип значення, що повертається функцією (тип результату). Якщо функція не повертає, то вказується тип void: void main ( ). Кожна функція, у тому числі і main повинна мати набір параметрів, може бути порожнім, тоді в дужках вказується (void).

За назвою функції розміщується тіло функції. Тіло функції - це послідовність визначень, описів та операторів, що виконуються у фігурні дужки. Кожне визначення, опис або оператор закінчується крапкою з комою.

Визначення - вводять об'єкти (об'єкт - це іменована область пам'яті, окремий випадок об'єкта - змінна), необхідні представлення у програмі оброблюваних даних. Прикладом є

```
int y = 10; //іменована константа
float x; //Змінна
```

Описи – повідомляють компілятор про властивості та імена об'єктів та функцій, описаних в інших частинах програми.

Оператори визначають дії програми на кожному кроці її виконання.

1.2. Константи та змінні

Константа - це значення, яке може бути змінено. Синтаксис мови визначає 5 типів констант:

- символи;
- константи типу, що перераховується;
- речові числа;
- цілі числа;

- нульовий вказівник (NULL).

Змінні можна змінювати. При заданні значення змінної відповідну їй область пам'яті поміщається код цього значення. Доступ до значення можливий через ім'я змінної, а доступ до ділянки пам'яті – на його адресу. Кожна змінна перед використанням у програмі має бути визначена, тобто їй має бути виділена пам'ять. Розмір ділянкипам'яті, що виділяється для змінної та інтерпретація вмісту залежать від типу, зазначеного у визначенні змінної. Найпростіша форма визначення змінних:

тип список_імен_змінних;

Основні типи даних

тип даних	назва	розмір, біт	діапазон значень
unsigned char	беззнаковий цілий довжиною не менше 8 біт	8	0 . . 255
char	цілий довжиною не менше 8 біт	8	-128. . 127
enum	перерахований	16	-32768. . 32767
unsigned int	беззнаковий цілий	16	0 . . 65535
short int (short)	короткий цілий	16	-32768. . 32767
unsigned short	беззнаковий короткий цілий	16	0 . . 65535
int	цілий	16	-32768. . 32767
unsigned long	беззнаковий довгий цілий	32	0 . . 4294967295
long	довгий цілий	32	-214748348. . 2147483647
float	речовий одинарної точності	32	3.4E-38. . 3.4E+38
double	речовий подвійний точності	64	1.7E-308. . 1.7E+308
long double	речовий максимальної точності	80	3.4E-4932. . 1.1E+4932

Відповідно до синтаксису мови змінні автоматичної пам'яті після стандартного визначення мають невизначені значення. Змінним можна надавати початкові значення, явно вказуючи їх у визначеннях:
тип ім'я_змінної = початкове_значення;
Цей прийом називається ініціалізацією.
Приклади:
```
float pi = 3.14, cc = 1.3456;
unsigned int year = 1999;
```

1.3. Операції

Унарні:

&	отримання адреси операнда
*	звернення за адресою (розйменування)
-	унарний мінус, змінює знак арифметичного операнда
~	порозрядне інвертування внутрішнього двійкового коду (побитове заперечення)
!	логічне заперечення (НЕ). Як логічні значення використовується 0 - брехня і не 0 - істина, запереченням 0 буде 1, запереченням будь-якого ненульового числа буде 0.
++	збільшення на одиницю: префіксна операція - збільшує операнд до його використання, постфіксна операція збільшує операнд після використання.
- -	зменшення на одиницю: префіксна операція - зменшує операнд до його використання, постфіксна операція зменшує операнд після використання.
sizeof	обчислення розміру (в байтах) для об'єкта того типу, що має операнд

Бінарні операції.
Адитивні:

+	бінарний плюс (складення арифметичних операндів)
-	бінарний мінус (віднімання арифметичних операндів)

Мультиплікативні:

*	множення операндів арифметичного типу
/	розподіл операндів арифметичного типу (якщо операнди

	цілі, то виконується цілісний поділ)
%	отримання залишку від поділу цілих операндів

Операції зсуву (визначені тільки для цілих операндів).
Формат вираження з операцією зсуву:
операнд_лівий операція_зсуву операнд_правий

<<	зрушення вліво бітового уявлення значення лівого цілого чисельного операнда на кількість розрядів, що дорівнює значенню правого операнда
>>	зрушення вправо бітового уявлення значення правого цілісного операнда на кількість розрядів, що дорівнює значенню правого операнда

Порозрядні операції:

&	порозрядна кон'юнкція (І) бітових уявлень значень цілісних операндів
|	порозрядна диз'юнкція (АБО) бітових уявлень значень цілісних операндів
^	порозрядне виключає АБО бітових уявлень значень цілісних операндів

Операції порівняння:

<	менше, ніж
>	більше, ніж
<=	менше чи одно
>=	більше чи одно
==	одно
!=	не одно

Логічні бінарні операції:

&&	кон'юнкція (І) цілісних операндів або відносин, цілісний результат брехня(0) або істина(1)
||	диз'юнкція (АБО) цілісних операндів або відносин, цілісний результат брехня(0) або істина(1)

Умовна операція.
На відміну від унарних і бінарних операцій, у ній використовується три операнда.
Вираз1? Вираз2: Вираз3;
Першим обчислюється значення виразу1. Якщо воно істинно, то обчислюється значення 2, яке стає результатом.
Якщо обчисленні выражения1 вийде 0, то ролі результату береться значення выражения3.
Наприклад:
x<0? -x: x; // обчислюється абсолютне значення x.
Операція явного (перетворення) приведення типу.

Існує дві форми: канонічна та функціональна:
1) (ім'я_типу) операнд
2) ім'я_типу (операнд)
Пріоритети операцій.

Ранг	Операції
1	( ) [ ] -> .
2	! ~ - ++ -- & * (тип) sizeof тип( )
3	* / % (мультиплікативні бінарні)
	+ - (Адитивні бінарні)
5	<<>> (порозрядного зсуву)
6	< > <= >= (відносини)
7	== != (відносини)
8	& (порозрядна кон'юнкція "І")
9	^ (порозрядне виключає «АБО»)
10	| (порозрядна диз'юнкція «АБО»)
11	&& (кон'юнкція «І»)
12	|| (Диз'юнкція «АБО»)
13	?: (умовна операція)
14	= *= /= %= -= &= ^= |= <<= >>= (операція присвоєння)
15	, (Оперка кома)

1.4. Вирази

З констант, змінних, роздільників та знаків операцій можна конструювати вирази. Кожен вираз складається з одного або декількох операндів, символів операцій та обмежувачів, якими найчастіше виступають квадратні дужки. Якщо вираз формує ціле чи речове число, це арифметичне вираз. В арифметичних виразах допустимі операції: + - */%.

Ставлення - це пара арифметичних виразів, об'єднаних знаком операції відносини. Логічний тип у Сі відсутня, тому прийнято, що відношення має ненульове значення, якщо воно істинне і 0, якщо воно є хибним.

1.5. Введення та виведення

1.5.1. Введення та виведення у стандартному Сі

Обмін даними із зовнішнім світом програма на стандартному Сі реалізує за допомогою бібліотеки функцій введення-виведення

```
#include <stdio.h>
```

1) printf (<форматний рядок>,<список аргументів>);

<форматний рядок> - рядок символів, укладених у лапки, який показує, як мають бути надруковані аргументи. Наприклад:

```
printf ( “Значення числа Пі дорівнює %f\n”, pi);
```
Форматний рядок може містити
1) символи, що друкуються текстуально;
2) специфікації перетворення
3) керуючі символи.
Кожному аргументу відповідає своя специфікація перетворення:
%d - десяткове ціле число;
%f - число з плаваючою точкою;
%c – символ;
%s – рядок.
\n - символ керування новий рядок.
2) scanf (<форматний рядок>,<список аргументів>);
Як аргументи використовуються покажчики. Наприклад:
```
scanf(“%d%f”, &x,&y);
```

1.5.2. Введення та виведення в Сі++

Використовується файл бібліотекиiostream.h, в якому визначено стандартні потоки введення даних від клавіатури cin та виведення даних на екран дисплея cout, а також відповідні операції
1) << - операція запису даних у потік;
2) >> - Операція читання даних із потоку.
Наприклад:
```
#include <iostream.h>;
. . . . . .
cout << “\nВведіть кількість елементів: ”;
cin >> n;
```

Приклад рішення варіанта лабораторної роботи №1(С)

```
// додаємо до "початкового коду" програми ПРОТОТИПИ бібліотечних функцій
#include <iostream>
#include <stdio.h>
#include <math.h>
using namespace std; // використовуємо "стандартний простір імен" за умовчанням
```

```
int main(int argc, char *argv[], char *env[]) { // "загальний вигляд" заголовка ГОЛОВНОЇ
процедури


    cout << "Привіт із кодеблоку" << endl;

    printf("Робота № 1 Варіант № 25\n");
    /* це багатостроковий коментар
      посилання на завдання на сайті
http://kit.znu.edu.ua/iLec/2sem/PROGR2/Lr2c/1/PM_%D0%A1++_1.htm
     1 завдання - обчислити значення виразу при різних речових типах даних (float та double).
     Обчислення слід виконувати двома способами:
     з використанням проміжних змінних для поелементних дій та
     без використання проміжних змінних (записавши весь вираз одним оператором)
     Порівняти та пояснити отримані результати.
     ((ab)**3 - (a**3 - 3 * a**2 *b) ) /(b**3 - 3* a * b**2)
     Примітка: 1) Якщо "уважно" подивитися та "розкрити дужки",
                  то цей вираз дорівнює МІНУС ОДИНИЦІ
               2) За умови завдання цього робити НЕ ТРЕБА,
                  тренуємося у написанні найпростішої програми

    */
    float a, b, res1, res2;
    float p1, p2, p3, p4, p5, p6, p7, p8, p9;
    a=1000.0; b = 0.0001;
    // a = 10.0; b = 1.;
    /** проміжні обчислення **/
    p1 = (ab); p2 = p1 * p1 * p1; //(ab)**3
    p3=a*a*a; p4=3*a*a*b; p5 = p3-p4; // (a**3 - 3 * a**2 *b)
    p6 = p2-p5; // чисельник дробу
    p7 = b * b * b; p8=3*a*b*b;
    p9 = p7-p8; // знаменник дробу
    res1 = p6/p9;
    printf("результат res1=%g\n", res1);
    /** без проміжних обчислень **/
```

```
    res2=( pow((ab),3) - ( pow(a,3) - 3*pow(a,2)*b ) ) / (pow(b,3)-3*a*pow(b,2));
    printf("результат res2=%g\n", res2);
    /*-----------------------------------------*/
    double ad, bd, resd1, resd2;
    double pd1, pd2, pd3, pd4, pd5, pd6, pd7, pd8, pd9;
    ad=1000.0; bd = 0.0001;
    // ad=10.0; bd = 1.; !!! СПРОБУЙТЕ З ЦИМИ ЗНАЧЕННЯМИ
    /** проміжні обчислення **/
    pd1=(ad-bd); pd2 = pd1 * pd1 * pd1; //(ab)**3
    pd3=ad*ad*ad; pd4=3*ad*ad*bd; pd5 = pd3-pd4; // (a**3 - 3 * a**2 *b)
    pd6 = pd2-pd5; // чисельник дробу
    pd7=bd*bd*bd; pd8=3*ad*bd*bd;
    pd9 = pd7-pd8; // знаменник дробу
    resd1 = pd6/pd9;
    printf("результат resd1=%g\n", resd1);
    /** без проміжних обчислень **/
    resd2=( pow((ad-bd),3) - ( pow(ad,3) - 3*pow(ad,2)*bd ) ) / (pow(bd,3)-3*ad*pow(bd,2));
    printf("результат resd2=%g\n", resd2);
    /* 2 завдання
       1) --m-++n
       2) m*n m++
    */
    int r, n, m;
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
    r=--m-++n;
    printf("після виконання n=%dm=%dr = --m-++n = %d\n", n, m, r);
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
    r=(-m) - (++n);
    printf("після виконання n=%dm=%dr = (--m) - (++n) = %d\n", n, m, r);
    printf("--------------------\n"); // Інші варіанти розміщення дужок
                                      // Очевидно неможливі і порушують синтаксис
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
    r= n*m < n++;
    printf("після виконання n=%dm=%dr = n*m < n++ = %d\n", n, m, r);
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
```

```
    r=(n*m) <(n++);
    printf("після виконання n=%dm=%dr = (n*m) <(n++) = %d\n", n, m, r);
    printf("--------------------\n"); // Інші варіанти розміщення дужок
                                      // Очевидно неможливі і порушують синтаксис
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
    r= n-> m++;
    printf("після виконання n=%dm=%dr = n--> m++ = %d\n", n, m, r);
    n=2; m=3; printf("до виконання n=%dm=%d\n", n, m);
    r=(n--)> (m++);
    printf("після виконання n=%dm=%dr = (n--) > (m++) = %d\n", n, m, r);
    printf("--------------------\n"); // Інші варіанти розміщення дужок
                                      // Очевидно неможливі і порушують синтаксис

    /** 2 частина роботи. Постановка задачі
        Використовуючи оператор циклу, знайти суму елементів,
        зазначених у конкретному варіанті.
        25) Знайти суму 7 членів ряду, в якому
        an=n**2 * exp( - n**(1/2) )
    **/
   printf("Робота № 1 ДРУГЕ ЗАВДАННЯ  \n");
   int k;
   double sum, an;
   k=0;
   sum = 0.0;
   while (k <=7) {
     an = (double) pow (k, 2) * exp (-pow ((double) k, 0.5L));
     sum+=an;
     printf("k=%4d an=%g sum=%g\n", k, an, sum); // Віддачна друк
     k++;
   };
   printf("sum=%g\n", sum);
    return 0;
}
```

Результат

```
Привіт із кодеблоку
Робота №1 Варіант №25
результат res1=2,13333e+006
результат res2=2,20703e+006
результат resd1=-1,00136
результат resd2=-1,00136

до виконання n=2 m=3
після виконання n=3 m=2 r = --m-++n = -1
до виконання n=2 m=3
після виконання n=3 m=2 r = (-m) - (++n) = -1
-----------------
до виконання n=2 m=3
після виконання n=3 m=3 r = n*m < n++ = 0
до виконання n=2 m=3
після виконання n=3 m=3 r = (n*m) <(n++) = 0
-----------------
до виконання n=2 m=3
після виконання n=1 m=4 r = n-- > m++ = 0
до виконання n=2 m=3
після виконання n=1 m=4 r = (n--) > (m++) = 0
-----------------
Робота № 1 ДРУГЕ ЗАВДАННЯ Варіант № 25
k= 0 an=0 sum=0
k= 1 an=0,367879 sum=0,367879
k= 2 an=0,972467 sum=1,34035
k= 3 an=1,59229 sum=2,93264
k= 4 an=2,16536 sum=5,098
k= 5 an=2,67195 sum=7,76995
k= 6 an=3,10815 sum=10,8781
k= 7 an=3,47665 sum=14,3548
sum=14,3548

Process returned 0 (0x0) execution time : 0.035 s
```

Press any key to continue.