Лабораторна робота 7 (С3)

Робота містить ТРИ завдання:

1) "Рядкове введення-виведення"

2) "Рядки"

3) "Блокове введення-виведення"

Приклади програм лабораторної роботи №3

Ціль:Вивчення символьних та рядкових змінних та способів їх обробки у мові С. Робота з текстовими файлами, введення-виведення текстової інформації та її зберігання на зовнішніх носіях. Робота з двійковими файлами, організація введення-виведення структурованої інформації та її зберігання на зовнішніх носіях.

1 частина роботи "Строкове введення-виведення"

"Рядкове введення-виведення"

Ціль:Робота з текстовими файлами, введення-виведення текстової інформації та її зберігання на зовнішніх носіях.

Постановка задачі

Створити текстовий файл F1 не менше ніж з 10 рядків і записати в нього інформацію
Виконати завдання. При розробці алгоритму вважати, що рядків у вихідному файлі заздалегідь невідомо.

Варіанти

1.варіант:
1)Скопіюйте у файл F2 тільки парні рядки з F1.
2)Підрахувати розмір файлів F1 та F2 (в байтах).

2. варіант:
1)Скопіювати у файл F2 тільки рядки з F1, які починаються з літери «А».
2)Підрахувати кількість слів у F2.

3. варіант:
1)Скопіювати у файл F2 тільки ті рядки з F1, які починаються і закінчуються на ту саму літеру.
2)Підрахувати кількість символів у F2.

4. варіант:
1)Скопіювати з файлу F1 у файл F2 рядки, починаючи з 4-го рядка.
2)Підрахувати кількість символів у останньому слові F2.

5. варіант:
1)Скопіювати з файлу F1 у файл F2 рядки, починаючи з K до +5. Значення K прочитати з консолі.
2)Підрахувати кількість голосних букв у файлі F2.

6. варіант:
1)Скопіювати з файлу F1 у файл F2 рядки, починаючи з N до K. значення N та K прочитати з консолі.
2)Підрахувати кількість приголосних букв у файлі F2.

7. варіант:
1)Скопіювати з файлу F1 у файл F2 усі рядки, крім тих, що починаються на літеру A.
2)Підрахувати кількість символів у першому слові F2.

8. варіант:
1)Копіювати з файлу F1 у файл F2 усі рядки, які не містять цифри.
2)Підрахувати кількість рядків, які починаються на літеру "A" у файлі F2.

9. варіант:
1)Копіювати з файлу F1 у файл F2 усі рядки, які містять лише одні цифри.
2)Знайти найдовше слово у файлі F2.

10. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, які не містять російських літер.
2)Знайти найкоротше слово у файлі F2.

11. варіант:

1)Скопіювати з файлу F1 у файл F2 всі рядки, крім того рядка, який найкоротший.
2)Надрукувати номер цього рядка.

12. варіант:

1)Скопіювати з файлу F1 у файл F2 всі рядки, крім того рядка, в якому найбільше цифр 5.
2)Надрукувати номер цього рядка.

13. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, що починаються на літеру «А» і розташовані між рядками з номерами N1 та N2. Значення N1 та N2 прочитати з консолі.
2)Визначити номер рядка файлу F2, в якому найбільше цифрових символів.

14. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, що не містять літеру «А» і розташовані між рядками з номерами N1 та N2. Значення N1 та N2 прочитати з консолі
2) Визначити номер рядка файлу F2, в якому найменше цифрових символів.

15. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, що закінчуються на літеру «А» і розташовані між рядками з номерами N1 та N2. Значення N1 та N2 прочитати з консолі
2)Визначити номер рядка файлу F2, в якому найбільше літер «А»

16. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, що починаються на літеру «А» і закінчуються на літеру «С», розташовані між рядками з номерами N1 та N2. Значення N1 та N2 прочитати з консолі.
2)Визначити кількість символів у першому та останньому рядку файлу F2.

17. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, що починаються на літеру «А», розташовані між рядками з номерами N1 і N2, а потім усі рядки від N2+3 і до останнього. Значення N1 та N2 прочитати з консолі.
2)Визначити кількість символів у першому та останньому рядку файлу F2.

18. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, в яких немає однакових символів.

2)Визначити кількість літер A у першому рядку файлу F2.

19. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, крім рядків із парною кількістю символів.
2)Визначити кількість літер qwer у першому рядку файлу F2.

20. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, в яких є однакові слова.
2)Визначити кількість символів "цифра" в останньому рядку файлу F2.

21. варіант:

1)Скопіювати з файлу F1 у файл F2 усі рядки, в яких є буква, що збігається з першою літерою першого рядка файлу F1.
2)Визначити кількість символів @ # у рядках файлу F2.

22. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, в яких є літера, що збігається з останньою літерою першого рядка файлу F1.
2)Визначити номер рядка файлу F2, в якому найбільше символів.

23. варіант:

1)Скопіювати у файл F2 тільки парні рядки з F1.
2)Визначити номер рядка файлу F2, в якому найменше символів.

24. варіант:

1) Скопіювати з файлу F1 у файл F2 усі рядки, в яких міститься хоча б два однакові символи.
2) Визначити номер рядка файлу F2, в якому найбільше символів.

25. варіант:

1)Копіювати з файлу F1 у файл F2 усі рядки, в яких немає однакових символів.
2) Визначити номер рядка файлу F2, в якому найбільше символів "цифра"

Методичні вказівки

Алгоритм розв'язання задачі зводиться до циклу рядкового читання рядків з вихідного файлу, перетворення прочитаного рядка, прийняття рішення про розміщення рядка у вихідний файл та його запис у вихідний файл.

Вихідний файл необхідно створити за допомогою будь-якого текстового редактора або взяти текст з Інтернету (наприклад **https://www.ukrlib.com.ua/**)

Коротка інформація про файли тут.

Нагору

2 частина роботи. "Рядки"

Ціль: Вивчення символьних та рядкових змінних та способів їх обробки у мові С.

Постановка задачі

Задано текстовий файл (створити будь-яким способом, наприклад, створити у будь-якому текстовому редакторі). Введіть його ім'я з консолі. Якщо файл не існує – повідомити про це. Файл підготувати так, щоб можна було перевірити працездатність Вашої програми.

Якщо потрібно за умовою завдання, для кожного рядка текстового файлу знайти слова, з яких вона складається. Слова - це набір символів, розділених один від одного одним або декількома пробілами (або іншими загальноприйнятими символами роздільника - двокрапка, кома, туга з комою тощо).

Виконати обробку кожного рядка файлу відповідно до завдання свого варіанта.

Варіанти

1. Перевірити чи є рядок паліндромом. (Паліндром - це вираз, який читається однакова зліва направо і праворуч наліво. Прогалини зазвичай не враховуються).

2. Надрукувати найдовше і найкоротше слово в рядку.

3. Надрукувати всі слова, які не містять голосних літер (працюйте з англійською абеткою).

4. Надрукувати всі слова, які містять одну цифру.

5. Друкувати всі слова, які збігаються з її першим словом.

6. Перетворити рядок таким чином, щоб спочатку в ньому були надруковані лише літери, а потім тільки цифри, не змінюючи порядку символів у рядку.

7. Перетворити рядок так, щоб усі літери були відсортовані за зростанням (працюйте з англійським алфавітом).

8. Перетворити рядок так, щоб усі цифри в ньому були відсортовані за спаданням.

9. Перетворити рядок так, щоб усі слова в ньому стали ідентифікаторами змінних, слова, що починаються з цифри або з символу, не англійська літера - видалити.

10. Надрукувати всі слова-паліндроми, які є в цьому рядку (див. 1 варіант).

11. Перетворити рядок таким чином, щоб на початку були записані слова, що містять тільки цифри, потім слова, що містять тільки літери, а потім слова, які містять і літери і цифри.

12. Перетворити рядок таким чином, щоб усі слова в ньому були відображені у дзеркальному вигляді. Наприклад, слово 12345abcd має бути відображене dcba54321 (дзеркально).

13. Перетворити рядок таким чином, щоб символи кожного слова в ньому були відсортовані за зростанням (працюйте з англійською абеткою).

14. Перетворити рядок таким чином, щоб символи кожного слова в ньому були відсортовані за спаданням (працюйте з англійською абеткою).

15. Перетворити рядок таким чином, щоб у ньому залишилися лише слова, що містять літери, решту слів видалити.

16. Перетворити рядок таким чином, щоб у ньому залишилися лише слова, що містять цифри, решту видалити.

17. Визначте, скільки разів у рядку зустрічаються слова, в яких містяться літери a, b, c.

18. Видалити з рядка всі слова, які є ідентифікаторами змінних.

19. Визначити скільки слів у рядку, що починаються з літери. (працюйте з англійською абеткою).

20. . Визначити скільки слів у рядку, що починаються з цифри.

19. Визначте, скільки разів у рядку зустрічається слова, що містять літеру a.

21. Визначте, скільки разів у рядку зустрічається слово "End".

22. Визначте, скільки разів у рядку зустрічається слово "привіт".

23. Визначте скільки разів у рядку зустрічається слово, яким немає "цифрових" символів.

24. Визначте скільки разів у всіх рядках файлу зустрічається задане слово. Задане слово ввести з консолі.

25. Визначте, скільки разів у всіх рядках файлу зустрічається слово, що містить заданий символ. Вказаний символ ввести з консолі.

Методичні вказівки

Алгоритм розв'язання задачі зводиться до розгляду рядка як масиву символів. Переглядаючи послідовно символи заданого рядка або рядків, необхідно виконати необхідне завдання.
У разі формування нового рядка не забувайте останнім символом у формованому рядку записати символ '\0'.

Коротка інформація про рядки тут.

Нагору

3 частина роботи. "Блокове введення-виведення"

Ціль: Робота з двійковими файлами, організація введення-виведення структурованої інформації та її зберігання на зовнішніх носіях.

Постановка задачі

Сформувати двійковий файл із елементів, заданої у варіанті структури.
Створити щонайменше ДЕСЯТЬ записів.
Відкрити створений файл, прочитати всі записи в масив.
Вивести всі дані на консоль, нумеруючи кожен запис.
Потім виконати видалення та додавання елементів відповідно до свого варіанта,
використовуючи для пошуку елементів, що видаляються або додаються, розроблену функцію.
Записати отримані записи у той же файл.
Формування запису, друк запису на консоль,
додавання та/або видалення елементів масиву записів
оформити так само у вигляді функцій.
Передбачити повідомлення про помилки під час відкриття файлу та
виконання операцій введення/виведення.
Всю необхідну інформацію для додавання/видалення записів ввести з консолі.

Варіанти

1. Структура "Абітурієнт":
   - прізвище, ім'я, по батькові;
   - рік народження;
   - оцінки вступних іспитів (3);
   - середній бал атестату.

   Видалити елемент із зазначеним номером, додати елемент після елемента із зазначеним прізвищем.

2. Структура "Співробітник":
   - прізвище, ім'я, по батькові;
   - посада
   - рік народження;
   - вести.

   Видалити елемент із зазначеним прізвищем, додати елемент після елемента із зазначеним номером.

3. Структура "Держава":
   - назву;
   - столиця;
   - чисельність населення;
   - зайнята площа.

   Видалити всі елементи, у яких чисельність менша за задану, додати елемент після елемента із зазначеним номером.

4. Структура "Людина":
   - прізвище, ім'я, по батькові;
   - домашню адресу;
   - номер телефону;
   - вік.

   Видалити всі елементи із заданим віком, додати елемент після елемента із заданим номером.

5. Структура "Людина":
   - прізвище, ім'я, по батькові;
   - рік народження;
   - зростання;
   - вага.

   Видалити всі елемент із зазначеним зростанням і вагою, додати елемент після елемента із зазначеним прізвищем.

6. Структура "Школяр":

- прізвище, ім'я, по батькові;
- клас;
- номер телефону;
- оцінки з предметів (математика, фізика, російська, література).

Видалити всі елементи, які мають 2 хоча б по одному предмету, додати елемент на початок файлу.

7. Структура "Студент":
- прізвище, ім'я, по батькові;
- домашню адресу;
- група;
- рейтинг.

Видалити всі елементи, у яких рейтинг менше заданого, додати 1 елемент до кінця файлу.

8. Структура "Покупець":
- прізвище, ім'я, по батькові;
- домашню адресу;
- номер телефону;
- номер кредитної картки.

Видалити 3 елементи з початку файлу, додати 3 елементи до кінця файлу.

9. Структура "Пацієнт":
- прізвище, ім'я, по батькові;
- домашню адресу;
- номер медичної картки;
- номер страхового полісу.

Видалити елемент із заданим номером медичної карти, додати 2 елементи на початок файлу.

10. Структура "Інформація":
- носій;
- обсяг;
- назву;
- автор.

Видалити перший елемент із заданим обсягом інформації, додати елемент перед елементом із зазначеним номером.

11. Структура "Відеокасету":
- назва фільму;
- режисер;
- тривалість;

- ціна.

Видалити всі елементи з ціною вище заданої, додати 3 елементи до кінця файлу.

12. Структура "Музичний диск":
- назву;
- автор;
- тривалість;
- ціна.

Видалити перший елемент із заданою тривалістю, додати 2 елементи після елемента із заданим номером.

13. Структура "Спортивна команда":
- назву;
- місто;
- кількість гравців;
- кількість набраних очок.

Видалити всі елементи з кількістю очок менше заданого, додати 2 елементи на початок файлу.

14. Структура "Стадіон":
- назву;
- адреса;
- місткість;
- види спорту.

Видалити елемент із заданою назвою, додати 2 елементи після елемента із зазначеним номером.

15. Структура "Автомобіль":
- марка;
- рік випуску;
- ціна;
- колір.

Видалити всі елементи, у яких рік випуску менше заданого, додати елемент на початок файлу.

16. Структура "Власник автомобіля":
- прізвище, ім'я, по батькові;
- номер автомобіля;
- телефон;
- номер техпаспорту.

Видалити елемент із заданим номером, додати 2 елементи перед елементом із заданим прізвищем.

17. Структура "Фільм":
   - назву;
   - режисер;
   - рік випуску;
   - вартість.

Видалити всі елементи, у яких вартість перевищує задану, додати елемент на початок файлу.

18. Структура "Книга":
   - назву;
   - автор;
   - рік видання;
   - кількість сторінок.

Видалити 3 елементи з початку файлу, додати елемент перед елементом із зазначеною назвою.

19. Структура "Фільм":
   - назву;
   - режисер;
   - країна;
   - прибуток, що приноситься.

Видалити 2 елементи з кінця файлу, додати елемент після елемента із зазначеною назвою.

20. Структура "Держава":
   - назву;
   - державну мову;
   - грошова одиниця;
   - курс валюти щодо $.

Видалити елемент із зазначеною назвою, додати 2 елементи до кінця файлу.

21. Структура "Автомобіль":
   - марка;
   - серійний номер;
   - реєстраційний номер;
   - рік випуску.

Видалити 3 елементи з початку файлу, додати елемент поле елемента із зазначеним реєстраційним номером.

22. Структура "Власник автомобіля":
   - прізвище, ім'я, по батькові;
   - номер автомобіля;
   - номер техпаспорту;

- відділення реєстрації ДАІ

Видалити елемент із заданим номером, додати 2 елементи перед елементом із заданим прізвищем.

23. Структура "Стадіон":
   - назву;
   - рік побудови;
   - кількість майданчиків;
   - види спорту.

Видалити всі елементи, у яких рік споруди менший за заданий, додати 2 елементи перед елементом із зазначеним номером.

24. Структура "Студент":
   - прізвище, ім'я, по батькові;
   - номер телефону;
   - група;
   - оцінки з 3 основних предметів.

Видалити всі елементи з групи із зазначеним номером, у яких середня арифметична оцінка менша за задану, додати елемент після елемента із заданим прізвищем.

25. Структура "Студент":
   - прізвище, ім'я, по батькові;
   - група;
   - телефон;
   - кількість захищених лабораторних робіт

Видалити елементи, у яких кількість лабораторних робіт менше 5, додати елемент перед елементом із заданим прізвищем.

Методичні вказівки

1. Для заповнення файлу можна використовувати функцію, яка формує одну структуру, вказану у варіанті типу. Значення елементів структури вводяться із клавіатури. Для введення можна використовувати операцію >> та функцію gets() або fflush (stdin); scanf ("%d",&n);.

2. При введенні структур можна реалізувати один із таких механізмів:
   - введення заздалегідь обраної кількості структур (щонайменше 10);
   - введення до появи структури із заданою кількістю ознак;
   - діалог із користувачем про необхідність продовжувати введення.

3. Для запису структури у файл та читання структури з файлу використовувати функції блокового вводу/виводу fread та fwrite.

4. Для видалення/додавання елементів до файлу використовувати допоміжний файл.

5. Коротка інформація та приклад тут.

Нагору

Приклади програм лабораторної роботи №3(С)

---

0) Демонстраційна програма для роботи №3 варіант 25

```
/**@
читаємо рядки з файлу
та розбиваємо прочитані рядки на слова
**/
// додаємо до "початкового коду" програми ПРОТОТИПИ бібліотечних функцій
#include <iostream>
#include <stdio.h>
#include <math.h>
```

```
#include <string.h>
using namespace std; // використовуємо "стандартний простір імен" за умовчанням
/* Прототипи функцій користувача */

// Головна програма
int main(int argc, char *argv[], char *env[]) { // "загальний вигляд" заголовка ГОЛОВНОЇ процедури
    setlocale(LC_ALL, ""); // для консолі
    FILE * h; // "покажчик" на файл(ову табличку)
    char name[]="main.cpp"; //lab3-0.txt"; // name - ПОВНЕ (з шляхом) ім'я файлу, рядки якого читатимемо
    char buf [512]; // буфер (рядок), куди читаємо черговий рядок з файлу
    int nstr; // Номер прочитаного рядка
    int lstr; // Довжина прочитаного рядка
    int nword; // Номер слова рядка
    char * p; // покажчик на чергові слова
    char separator[] =".,/\\| \n\t\r!'\";+=-"; // це символи роздільники слів
    h=fopen(name, "r+t");
    if (h == NULL ) {
        sprintf(buf,"Помилка %d відкриття файлу %s\n", errno,name);
        perror(buf);
        return -1;
    };
    // fgets(куди , мах скільки , звідки) якщо поверне NULL , то кінець файлу
    nstr = 0;
    while (fgets(buf, 512, h)! = NULL) { // цикл по рядках з файлу
        nstr++;
        lstr = strlen (buf);
        if (lstr >=1 && buf[lstr-1] == '\n' )buf[lstr-1]='\0';
        // у buf черговий прочитаний рядок
        printf("стор № %4d < %s >\n", nstr, buf);
        ////////////////////////////////////////////
        nword = 0;
        // Розіб'ємо рядок на слова це символи роздільники слів
                p = strtok (buf, separator);
        while (p! = NULL) { // цикл за словами рядка
           nword++;
           printf(" слово %3d < %s >\n", nword, p);
           p = strtok (NULL, separator);
        };
        ////////////////////////////////////////////
```

```
    };
    fclose(h);
};
```

нагору

---

1) Рішення першой частини роботи

```
/**
25. Випадок лаб. раб № 3:
1) Скопіювати з файлу F1 у файл F2 усі рядки, в яких немає однакових символів.
2) Визначити номер рядка файлу F2, в якому найбільше символів "цифра"
**/
// додаємо до "початкового коду" програми ПРОТОТИПИ бібліотечних функцій
#include <iostream>
#include <stdio.h>
#include <math.h>
#include <string.h>
using namespace std; // використовуємо "стандартний простір імен" за умовчанням

/* Прототипи функцій користувача */
bool isDUBL(char * s); // повертає true якщо у рядку s є однакові символи
int kolDIG(char * s); // повертає кількість символів цифра у рядку s

// Головна програма
int main(int argc, char *argv[], char *env[]){ // "загальний вигляд" заголовка ГОЛОВНОЇ процедури
    setlocale(LC_ALL, ""); // для "консолі"

    //1) Скопіювати з файлу F1 у файл F2 всі рядки, у яких немає однакових символів.

    FILE * F1, * F2; // "Покажчик" на файли 1 і 2
    char name1[]="F1.txt"; // name1 - ПОВНЕ (з шляхом) ім'я 1 файлу, рядки якого читатимемо
    char name2[]="F2.txt"; // name2 - ПОВНЕ (з шляхом) ім'я 2 файли, рядки якого будемо писати
    char buf [512]; // буфер (рядок), куди читаємо/пишаємо черговий рядок з файлу
    int nstr1; // Номер прочитаного рядка F1
```

```
    int nstr2; // Номер прочитаного рядка F2
    int kdig; // кількість символів цифра у рядку
    int kdig_max; // максимальна кількість символів цифра у рядках
    int n_str_dig_max; //Номер рядка у файлі F2,в якому мах кількість символів цифра
    int lstr; // Довжина рядка
    F1 = fopen (name1, "r");
    if (F1 == NULL ) {
        sprintf(buf,"Помилка %d відкриття файлу %s\n", errno,name1);
        perror(buf);
        return -1;
    };
    F2 = fopen (name2, "w");
    if (F2 == NULL ) {
        sprintf(buf,"Помилка %d відкриття файлу %s\n", errno,name2);
        perror(buf);
        fclose(F1);
        return -1;
    };
    // куди мах скільки звідки якщо поверне NULL, то кінець файлу
    nstr1=0;
    nstr2=0;
    while (fgets (buf, 512, F1)! = NULL) { // цикл по рядках з файлу F1
        nstr1++;
        lstr = strlen (buf); // "обріжемо" ПЕРЕКЛАД РЯДКУ
        if (lstr >=1 && buf[lstr-1] == '\n' )buf[lstr-1]='\0';
        //У buf черговий прочитаний рядок F1
        printf("F1: стор № %4d < %s >\n", nstr1, buf); // для налагодження
        ////////////////////////////////////////////
        if ( ! isDUBL(buf) ) {
           nstr2++;
           printf(" стор № %d(F2) стор № %4d(F1) < %s > copy F1->F2\n", nstr2, nstr1, buf); // для
налагодження
           fprintf(F2, %s\n, buf); // додамо переклад рядка fputs (buf, F2);

        };
        ////////////////////////////////////////////
    };
    fclose(F1);
    fclose(F2);
```

```
    // 2) Визначити номер рядка файлу F2, в якому найбільше символів "цифра"
    F2=fopen(name2,"r");
    if (F2 == NULL ) {
        sprintf(buf,"Помилка %d відкриття файлу %s\n", errno,name2);
        perror(buf);
        fclose(F1);
        return -1;
    };
    kdig_max=0; //максимальна кількість символів цифра у рядках
    n_str_dig_max=-1; //номер рядка в якому мах кількість символів цифра
    nstr2=0; // Номер рядка файлу F2
    if ( fgets(buf,512,F2) == NULL ) {
        printf("файл %s не містить рядків....\n", name2);
        return 0;
    };
    // є хоча б один рядок у F2
    nstr2++;
    kdig = kolDIG(buf); // кількість символів цифра у рядку
    // Припустимо, що
    kdig_max = kdig; // максимальна кількість символів цифра у рядках
    n_str_dig_max = nstr2; //номер рядка в якому мах кількість символів цифра
    // Перевіримо припущення
    while ( ( fgets(buf,512,F2) != NULL ) ) { // для кожного рядка з файлу F2
         nstr2++; // Номер поточного рядка
         kdig = kolDIG(buf); // кількість символів цифра у поточному рядку
         if (kdig > kdig_max) {kdig_max = kdig; n_str_dig_max = nstr2; };
    };
    fclose (F2);
    printf( " У файлі %s\n рядок %d\n містить мах кількість %d символів цифра\n",
                        name2, n_str_dig_max, kdig_max);

};
bool isDUBL(char * s){
     // повертає true якщо у рядку s є однакові символи
     int ls = strlen (s);
     // "Пряме ... рішення"
     for (int i=0; i < ls; i++)
```

```
        for(int j=i+1; j < ls; j++)
            if (s[i] == s[j]) return true;
     return false;
};
int kolDIG(char * s) {
  // повертає кількість символів цифра у рядку s
  int kd=0;
  for(unsigned int i=0; i < strlen(s); i++ )
    if ( '0' <= s[i] && s[i] <= '9' ) kd++;
  return kd;
};
```

нагору

---

2) Рішення**другий**частини роботи

```
/**
25. Визначте скільки разів у всіх рядках файлу зустрічається слово,
  що містить заданий символ.
  Вказаний символ ввести з консолі.
  **/
  // додаємо до "початкового коду" програми ПРОТОТИПИ бібліотечних функцій
#include <iostream>
#include <stdio.h>
#include <math.h>
#include <string.h>
using namespace std; // використовуємо "стандартний простір імен" за умовчанням
/* Прототипи функцій користувача */
bool isCHAR(char c, char * s); // true якщо у рядку s зустрічається символ c
// Головна програма
int main(int argc, char *argv[], char *env[]) { // "загальний вигляд" заголовка ГОЛОВНОЇ процедури
```

```
    setlocale(LC_ALL, ""); // для "консолі"
    FILE * h; // "покажчик" на файл(ову табличку)
    char name [256]; // name - ПОВНЕ (з шляхом) ім'я файлу, рядки якого читатимемо
    char buf [512]; // буфер (рядок), куди читаємо черговий рядок з файлу
    int nstr; // Номер прочитаного рядка
    int lstr; // Довжина прочитаного рядка
    int nword; // Номер слова рядка
    int ws; // скільки разів у всіх рядках файлу зустрічається слово,
                // Що містить заданий символ.
    char * p; // покажчик на чергові слова
    char separator[]=".,/\\| \n\t\r!'\";+=-"; // це символи роздільники слів
    char x_char; // символ, який шукаємо у словах
    printf("введіть повне ім'я файлу:"); fflush (stdin); scanf("%s", name);
    printf("*** працюємо з файлом %s\n", name);
    printf("введіть символ, який шукаємо у словах файлу: ");
    fflush (stdin); scanf("%c", &x_char);
    printf("*** шукаємо слова, що містять символ %c\n", x_char);
    //---------------------------------------------------------------
    h=fopen(name, "r+t");
    if (h == NULL ) {
        sprintf(buf,"Помилка %d відкриття файлу %s\n", errno,name);
        perror(buf);
        return -1;
    };

    nstr = 0;
    ws=0;
// куди мах скільки звідки якщо поверне NULL, то кінець файлу
    while (fgets(buf, 512, h)! = NULL) { // цикл по рядках з файлу
        nstr++;
        lstr = strlen (buf);
        if (lstr >=1 && buf[lstr-1] == '\n' )buf[lstr-1]='\0';
        // у buf черговий прочитаний рядок
        ///printf("стор № %4d < %s >\n", nstr, buf);
        ////////////////////////////////////////
        nword = 0;
        // Розіб'ємо рядок на слова separator це символи роздільники слів
        p = strtok (buf, separator);
        while (p! = NULL) { // цикл за словами рядка
```

```
            nword++;
            ///printf(" слово %3d < %s >\n", nword, p);
            if (isCHAR(x_char, p)) ws++;
            p = strtok (NULL, separator);
         };
         ////////////////////////////////////////
    };
    fclose(h);
    printf("Кількість слів у записах файлу %s\n", name);
    printf(" у яких зустрічається символ %c\n", x_char);
    printf(" одно %d\n", ws );
};
bool isCHAR(char c, char * s) {
     // true якщо у рядку s зустрічається символ c
   for(unsigned int i=0; i < strlen(s); i++ )
       if (s[i] == c) return true;
   return false;
};
```

Нагору

---

3)Рішення третьої частини роботи

Реалізуйте нижче написану програму та виконайте її

```
/**
Постановка задачі
Сформувати двійковий файл із елементів,
заданої у варіанті структури.
Створити щонайменше ДЕСЯТЬ записів.
Відкрити створений файл, прочитати всі записи в масив.
Вивести всі дані на консоль, нумеруючи кожен запис.
Потім виконати видалення та додавання елементів у
відповідно до свого варіанта,
використовуючи для пошуку видалених або доданих
```

```
елементів розробленої функції.
Записати отримані записи у той же файл.
Формування запису, друк запису на консоль,
додавання та/або видалення елементів масиву записів
оформити так само у вигляді функцій.
Передбачити повідомлення про помилки під час відкриття файлу та
виконання операцій введення/виведення.
Всю необхідну інформацію для додавання/видалення записів ввести з консолі

  25. Структура "Студент":
       прізвище, ім'я, по батькові;
       група
       телефон
      кількість лабораторних робіт.

      Видалити елементи, у яких кількість заданих лабораторних робіт менше 5,
      додати елемент перед елементом із заданим прізвищем.
  **/
// додаємо до "початкового коду" програми ПРОТОТИПИ бібліотечних функцій
#include <iostream>
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include <windows.h>
#include <conio.h>
#include <io.h>
using namespace std; // використовуємо "стандартний простір імен" за умовчанням
/* глобальні змінні */
struct stud {// опис структури
char f [20]; //прізвище
char i [20]; //ім'я
char o [20]; //по батькові
char grup [10]; // група
char tel [13]; // Телефон
int k; // кількість робіт
};
// максимальне число елементів групи
#define max_GR 100
```

```
// ім'я файлу для збереження списку...
#define file_name "lab3_3.dat"

/* Прототипи функцій користувача */
void show (stud * GR, int kGR); //показ всього списку з GR
void show1 (stud * GR, int k, int kGR); //Показ k елемента списку
stud vvesti(); //Введення елемента з клавіатури
stud vvesti1(int k); //генерації елемента для налагодження та тестування
int search (stud * GD, int kGR, char * st); // пошук у списку прізвища з рядка st
void copy1 (stud * from, stud * to); //Копіювання одного запису
void create(stud * GR, int & kGR); // Створити групу
void create1(stud * GR, int & kGR); // Створити групу автоматично для тестування
bool savefile(stud * GR, int kGR); // Записати масив у файл
bool loadfile(stud * GR, int & kGR); // завантажити масив структур GR, з файлу з ім'ям з file_name
bool add1(stud * GR, int & kGR); // Вставити перед записом із заданим прізвищем
bool del1(stud * GR, int & kGR); // видалити всіх, хто має робіт < 5

// Функції перетворення кодувань 1251 <===> 866
char * Str2Ansi (const char * ASourceString, char * buf);
char * Str2OEM (const char * ASourceString, char * buf);

// Головна програма
int main(int argc, char *argv[], char *env[]) { // "загальний вигляд" заголовка ГОЛОВНОЇ процедури
    // Char buf [512];
    stud GR[max_GR]; // Список групи студентів
    int kGR; // кількість студентів у групі
    kGR=0; // ні чого не створено
    char c;

    //Меню в головній програмі
    while (1) {
     printf ( "\n 1. Показати все (Show all)"
              "\n 2. створити список студентів"
              "\n 3. зберегти список студентів у файл"
              "\n 4. завантажити список студентів з файлу"
              "\n----------------------------------"
              "\n 5. Вставити нового студента перед студ. із заданою фам. "
              "\n 6. Видалити студентів з кількістю робіт < 5"
              "\n----------------------------------"
```

```
          "\n 0. Закінчити\n ");
   while (true) { if ( kbhit() != 0 ) { c=getch(); break;}; };
  // fflush (stdin); scanf ("%c",&c);
  switch (c) {
  case '1': show(GR, kGR);
            break;
  case '2': printf("введіть: створити "вручну"-1"
                   "\"автомат\"-2 \"інший смвол\"-відмова\n");
             printf("(якщо список вже був створений, він буде замінений на новий)\n");
            // fflush (stdin); scanf ("%c",&c);
            while (true) { if ( kbhit() != 0 ) { c=getch(); break;}; };
            if(c == '1') {create(GR, kGR); break; };
            if(c == '2') {create1(GR, kGR); break; };
            printf("... відмова від створення нового списку\n");
            break;
  case '3':
            if (! savefile(GR, kGR)) {
              printf("зберегти список студентів у файл не вдалося ....\n");
            };
            break;
  case '4':
      if (! loadfile(GR, kGR)) {
              printf("завантажити список студентів з файлу не вдалося ....\n");
            };
             break;
  case '5':
             if ( ! add1(GR, kGR) ){
              printf("вставити новий запис не вдалося ....\n");
            };
            break;
  case '6':
             if (! del1 (GR, kGR)) {
              printf("видалити не вдалося ....\n");
            };
            break;
  case '0':
         return 0;
  };
};
```

```
};
stud vvesti() { //введення елемента з клавіатури
char buf[512], str[512];
stud p;
printf ("прізвище?"); fflush(stdin); scanf("%20s", str); strcpy(pf, Str2Ansi(str, buf));
printf ("ім'я?"); fflush(stdin); scanf("%20s", str); strcpy(pi, Str2Ansi(str, buf));
printf ("по батькові?"); fflush(stdin); scanf("%20s", str); strcpy(po, Str2Ansi(str, buf));
printf ("група?"); fflush(stdin); scanf("%6s", str); strcpy(p.grup, Str2Ansi(str, buf));
printf ("телефон?"); fflush(stdin); scanf("% 13s", p.tel); strcpy(p.tel, Str2Ansi(str, buf));
printf ("кількість робіт?"); fflush (stdin); scanf ("%d",&p.k);
return p;
};
stud vvesti1(int k) { //генерації елемента для налагодження та тестування
char str [512];
stud p;
sprintf(str,"прізвище_%d", k); strcpy(pf, str);
sprintf(str,"ім'я_%2d",k); strcpy(pi, str);
sprintf(str,"по батькові_%2d",k); strcpy(po, str);
sprintf (str, "6.1229"); strcpy(p.grup, str);
sprintf(str, "050xxxxxx_%2d",k); strcpy(p.tel, str);
pk = k;
return p;
};
void create(stud * GR, int & kGR) {
   // Створити групу
   char ans;
   stud p;
   kGR=0;
   printf("--- створення групи ---");
   while (kGR <= max_GR) {
     printf("вводьте дані про %d студента\n", kGR+1);
     p=vvesti();
     copy1(&p, &GR[kGR]);
     kGR++;
     //m:
     printf("для введення наступного введіть 1, для завершення - будь-який символ\n");
     while (true) { if ( kbhit() != 0 ) { ans=getch(); break;}; }; // fflush (stdin); scanf("%c", &ans);
     if (ans == '1') continue;
     break;
```

```
      //goto m;
    }
};
bool savefile(stud * GR, int kGR) {
    // записати масив структур GR, файл з ім'ям з file_name
    // Повернення - true - удача, false - не змогли
    //відкриваємо Двійковий файл на ЗАПИС
    char buf [512];
    FILE * fb;
    fb = fopen (file_name, "wb");
    if ( fb == NULL ) {
         sprintf(buf,"Помилка %d відкриття бінарного файлу %s для запису масиву\n", errno,file_name);
         perror(buf);
         return false;
     };
// int fwrite( void *ptr, int size, int n, FILE *fp) , де
// void *ptr - покажчик на область пам'яті, в якій розміщуються дані, що записуються в файл;
// int size - розмір одного записуваного елемента;
// int n - кількість записуваних елементів;
// FILE * fp - покажчик на файл, у якому виконується запис.
// У разі успішного запису інформації, функція повертає число записаних елементів, інакше повертає EOF.
     int k = fwrite (GR, sizeof (stud), kGR, fb);
     //printf(" ------------------------- k=%d\n", k);
     fclose(fb);
     if ( k != kGR) { printf("... вдалося записати %d елементів із %d\n", k, kGR);
                      return false;
     };
     printf("... список %d елементів збережений у файл\n",k);
     return true;
};

bool loadfile(stud * GR, int & kGR) {
    // завантажити масив структур GR, з файлу з ім'ям з file_name
    // Повернення - true - удача, false - не змогли
    //відкриваємо Двійковий файл на ЗАПИС
    char buf [512];
    FILE * fb;
    fb = fopen (file_name, "rb");
    if ( fb == NULL ) {
```

```
        sprintf(buf,"Помилка %d відкриття бінарного файлу %s для читання масиву\n", errno,file_name);
        perror(buf);
        return false;
    };
// 1) int fread( void *ptr, int size, int n, FILE *fp) , де
// void *ptr - покажчик на область пам'яті, в якій розміщуються дані, що зчитуються з файлу;
// int size - розмір одного елемента, що зчитується;
// int n - кількість елементів, що зчитуються;
// FILE * fp - покажчик на файл, із якого виробляється зчитування.
// У разі успішного зчитування інформації, функція повертає число прочитаних елементів (а не байтів), інакше
повертає EOF.
     kGR=0;
     int k;
     while (( k = free (&GR [kGR], sizeof (stud), 1, fb))! = 0) { // NULL
       //printf(" -------------------kGR=%d--- k=%d\n", kGR, k);
       kGR++;
     };
     printf("... список %d елементів завантажений з файлу\n", kGR);
     fclose(fb);
     return true;
};
bool add1(stud * GR, int & kGR) {
  // Вставити перед записом із заданим прізвищем
  // !!! як це не зручно робити !!!!!!
  char fam [21], fam1 [21], buf [512];
  if (kGR> = max_GR) {
    printf("... список вичерпаний....!\n");
    return false;
  };
  int kfam = -1;
  printf("Вкажіть прізвище, перед яким вставити запис\n");
  fflush(stdin); scanf("%20s", fam1);
  strcpy(fam, Str2Ansi(fam1, buf));
  // Пошук номера запису з прізвищем fam
  printf("... пошук номера запису з прізвищем %s\n", fam);
  for(int i=0; i < kGR; i++) {
    if ( strcmp(GR[i].f, fam) == 0 ) {
        kfam = i;
        break;
```

```
    };
  };
  // вставимо перед kfam записом
  if (kfam == -1) {
    printf("... запис з прізвищем %s не знайдено!\n", fam);
    return false;
  };
  printf("... запис знайдено:\n");
  show1(GR, kfam, kGR);
  if ( kfam == max_GR ) {
    printf("... запис із прізвищем %s останній у списку\n список вичерпаний....!\n", fam);
    return false;
  };
  stud p;
  printf("вводьте дані про елемент, що вставляється:\n");
  p=vvesti();
  //
  // 1) zap from 0 to kfam-1 ---> file
  // 2) zap p ----> file
  // 3) zap from kfam to kGR ---> file
  // 4) load file to GR
  //
  FILE * fb;
  char tname[]="$temp.dat";
  int kz;
  fb = fopen (tname, "wb");
   if ( fb == NULL ) {
        sprintf(buf,"Помилка %d відкриття тимчасового бінарного файлу %s для перетворення масиву\n", errno,
tname);
        perror(buf);
        return false;
    };
   // 1) !!!!!
   for ( int i=0; i <= (kfam-1); i++) {
        kz=fwrite(&GR[i], sizeof(stud), 1, fb);
        if ( kz != 1) { printf("... не вдалося записати %d елемент у тимчасовий файл %s\n", i, tname);
          fclose (fb);
          return false;
        };
```

```
   };
   // 2) !!!!!
   kz = fwrite (p, sizeof (stud), 1, fb);
   if ( kz != 1) { printf("... не вдалося записати НОВИЙ елемент у тимчасовий файл %s\n", tname);
     fclose (fb);
     return false;
   };
   // 3) !!!!!
   for ( int i=kfam; i < kGR; i++) {
        kz=fwrite(&GR[i], sizeof(stud), 1, fb);
        if ( kz != 1) { printf("... не вдалося записати %d елемент у тимчасовий файл %s\n", i, tname);
          fclose (fb);
          return false;
        };
   };
   fclose(fb);
   // 4) !!!!!
   fb = fopen (tname, "rb");
   if ( fb == NULL ) {
        sprintf(buf,"Помилка %d відкриття тимчасового бінарного файлу %s для перетворення масиву\n",
errno,tname);
        perror(buf);
        return false;
    };
    kGR=0;
    while ((kz = free (&GR [kGR], sizeof (stud), 1, fb))! = 0) { // NULL
       //printf(" -------------------kGR=%d--- k=%d\n", kGR, k);
       kGR++;
    };
     //printf("... список %d елементів завантажений з файлу\n", kGR);
    fclose(fb);
    printf("... новий елемент списку вставлений \n");
    return true;
}

void create1(stud * GR, int & kGR) {
   // Створити групу автоматично 10 елементів
   // Char ans;
   stud p;
```

```
   kGR=0;
   printf("--- створення групи автоматично ---");
   while (kGR < 10) { // max_GR) {
     //printf("вводьте дані про %d студента\n", kGR);
     p=vvesti1(kGR);
     copy1(&p, &GR[kGR]);
     show1(GR, kGR, kGR);
     kGR++;
   };
   return;
};

void show (stud * GR, int kGR) {
  //показ всього списку з GR
  printf("\n--- список групи ---\n");
  for ( int i=0; i < kGR; i++) {
     printf ("%3d) робіт: %2d -> %20s %20s %20s %6s %13s\n", i,
              GR[i].k, GR[i].f, GR[i].i, GR[i].o, GR[i].grup, GR[i].tel);
  };
  printf ("\nУ списку всього %d зап.\n",kGR);
  return;
};

void show1(stud * GR, int k, int kGR) {
    // показ k-го елемента
   printf("\n--- %d елемент групи ---\n", k);
   if (k> = 0 | | k < kGR)
      printf ("раб: %2d -> %20s %20s %20s %6s %13s\n",
                GR[k].k, GR[k].f, GR[k].i, GR[k].o, GR[k].grup, GR[k].tel);
   else
     printf ("запису № %d немає у списку (всього %d зап. у списку)\n",k, kGR);
   return;
};

int search (stud * GR, int kGR, char * st) {
    // пошук у списку прізвища з рядка st
    // повернення - номер масиву або -1
    for ( int i=0; i < kGR; i++) {
         if (strcmp(GR[i].f, st) == 0) return i;
```

```
    };
    return -1;
};
//------------
bool del1(stud * GR, int & kGR) {
  // видалити всіх, хто має робіт < 5
  // !!! як це не зручно робити !!!!!!
  //
  // 1) zap from 0 to kGR if k_робіт >= ---> file
  // 2) load file to GR
  //
  FILE * fb;
  char tname[]="$temp.dat";
  int kz;
  char buf [512];
  fb = fopen (tname, "wb");
   if ( fb == NULL ) {
        sprintf(buf,"Помилка %d відкриття тимчасового бінарного файлу %s для видалення елементів масиву\n",
errno, tname);
        perror(buf);
        return false;
    };
   // 1) !!!!!
   for ( int i=0; i < kGR; i++) {
        if (GR[i].k >= 5) {
          kz=fwrite(&GR[i], sizeof(stud), 1, fb);
          if ( kz != 1) { printf("... не вдалося записати %d елемент у тимчасовий файл %s\n", i, tname);
             fclose (fb);
             return false;
          };
        };
   };
   fclose(fb);
   // 2) !!!!!
   fb = fopen (tname, "rb");
   if ( fb == NULL ) {
        sprintf(buf,"Помилка %d відкриття тимчасового бінарного файлу %s для видалення елементів масиву\n",
errno,tname);
        perror(buf);
```

```
        return false;
    };
    kGR=0;
    while ((kz = free (&GR [kGR], sizeof (stud), 1, fb))! = 0) { // NULL
       //printf(" ------------------kGR=%d--- k=%d\n", kGR, k);
       kGR++;
    };
    fclose(fb);
    printf("... студенти з кількістю робіт МЕНШЕ П'ЯТИ видалені зі списку\n");
    return true;
//----
};
void copy1 (stud *from, stud *to) {
   //Копіювання одного запису
  strcpy (to->f, from->f);
  strcpy (to->i, from->i);
  strcpy (to->o, from->o);
  strcpy (to->grup, from->grup);
  strcpy (to->tel, from->tel);
  to->k = from->k;
};
char * Str2Ansi (const char * ASourceString, char * buf){
  OemToCharBuff(ASourceString, buf, 512);
  return buf;
};
char * Str2OEM (const char * ASourceString, char * buf) {
  CharToOemBuff (ASourceString, buf, 512);
  return buf;
};
```

Нагору

"Рядкове введення-виведення"

Короткі теоретичні відомості

Відкриття та закриття потоку

Перш ніж розпочати працювати з файлом, його треба ініціювати, тобто відкрити. При цьому покажчик файлу пов'язується зі структурою певного типу FILE, визначення якої знаходиться у файлі бібліотеки <stdio.h>. У структурі знаходиться покажчик на буфер, покажчик на поточну позицію файлу і т. п. При відкритті потоку повертається покажчик на потік, тобто на об'єкт типу FILE.

```
#include <stdio.h>;
. . . . . .
FILE * fp;
. . . . . .
fp = fopen ("t.txt", "r");
```

де fopen(<ім'я_файлу>,<режим_відкриття>) - функція для ініціації файлу.

Існують такі режими для відкриття файлу:
”w” - відкрити файл для запису, якщо файл існує, він стирається;
”r” – відкрити файл для читання;
”a” - відкрити файл для додавання, якщо файл існує, він не стирається і можна писати в кінець файлу;
”w+” - відкрити файл для запису та виправлення, якщо файл існує, то він стирається, а далі можна і читати, і писати, розміри файлу можна збільшувати;
”r+” - відкрити файл для читання та запису, але збільшити розмір файлу не можна;
”a+” - відкрити файл для додавання, тобто можна читати і писати, в тому числі і в кінець файлу.

Потік можна відкрити у текстовому (t) або двійковому (b) режимі. За замовчуванням – текстовий режим. У певному вигляді режим вказується так: ”r+b”або ”rb” - двійковий (бінарний) режим.

Приклад:

```
if ((fp=fopen("t.txt", "w")==NULL)
{
perror("\nпомилка при відкритті файлу"); // виводить рядок символів із повідомленням про помилку
exit(0);
}
```

Після роботи з файлом його треба закрити

```
fclose(<покажчик_на_потік>);
```

Для рядкового введення-виведення використовуються такі функції:

1)char *fgets(char *s, int n, FILE *F), де

char *s - адреса, яким розміщуються лічені байти;
int n - кількість зчитуваних байтів;
FILE *fp - покажчик на файл, з якого виконується зчитування.

Прийом символів закінчується після передачі n байтів або при отриманні \n.
Керуючий символ "\n" теж передається в рядок, що приймає. У будь-якому випадку рядок закінчується "\0". При успішному завершенні зчитування функція повертає покажчик на прочитаний рядок, інакше повертає NULL.

2) char * fputs (char * s, FILE * F), де

char *s - адреса, з якої беруться байти, що записуються у файл;
FILE *fp - покажчик на файл, у якому виконується запис.

Приклад:

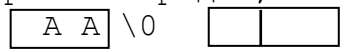
```
int MAXLINE=255; //максимальна довжина рядка
FILE *in // Вихідний файл
     *out; //Приймає файл
char buf [MAXLINE]; //Рядок, за допомогою якого виконується копіювання
if ( ( in=fopen("ім'я_вихідного_файлу","rt") == NULL )
{

      perror("\nпомилка при відкритті вихідного файлу"); exit(666);
};

if ( ( out=fopen("ім'я_результуючого_файлу","wt") == NULL )
{

      perror("\nпомилка при відкритті результуючого файлу"); exit(666);
};

//Копіювання рядків одного файлу в інший
while (fgets (buf, MAXLINE, in)! = NULL)
     fputs(buf,out);

fclose(in); fclose(out);
```

Нагору

"Рядки"

1. Короткі теоретичні відомості

Для надання символьної (текстової) інформації можна використовувати символи, символьні змінні та символьні константи.
Символьна константа є послідовністю знаків, що у лапки: “Початок рядка \n”. У Сі немає окремого типу для рядків. Масив символів – це і є рядок. Кількість елементів у такому масиві на один елемент більше, ніж зображення рядка, тому що в кінець рядка додано '\0' (нульовий байт або нуль-термінатор).



'A' “A”
символ(1 байт) рядок (2 байти)

Присвоїти значення масиву символів за допомогою звичайного оператора присвоєння не можна. Помістити рядок у масив можна або під час введення, або за допомогою ініціалізації:

```
char s [] = "ABCDEF";
```

Для роботи з рядками є спеціальна бібліотека string.h. Приклади функцій роботи з рядками з бібліотеки string.h:

Функція	Прототип та короткий опис функції
strcmp	int strcmp (const char * str1, const char * str2); Порівнює рядки str1 та str2. Якщо str1< str2, то результат негативний, якщо str1 = str2, то результат дорівнює 0, якщо str1> str2, то позитивний результат.
strcpy	char * strcpy (char * s1, const char * s2); Копіює байти з рядка s1 до рядка s2
strdup	char * strdup (const char * str); Виділяє пам'ять та перенесення в неї копію рядка str.
strlen	unsigned strlen (const char * str);

	Обчислює довжину рядка str.
strncat	char * strncat (char * s1, const char * s2, int kol); Приписує kol символів рядка s1 до рядка s2.
strncpy	char * strncpy (char * s1, const char * s2, int kol); Копіює kol символів рядка s1 у рядок s2.
strnset	char * strnset (char * str, int c, int kol); Замінює перші kol символів рядка s1 символом c.

Самостійно вивчіть функцію strtok

Рядки, під час передачі у функцію, як фактичних параметрів може бути визначено або як одномірні масиви типу char[], або як покажчики типу char*. На відміну від звичайних масивів, у цьому випадку немає необхідності явно вказувати довжину рядка.

Нагору

Введення та виведення в Сі

Особливістю Сі є відсутність у цій мові структурованих файлів. Усі файли розглядаються як структурована послідовність байтів. При такому підході поняття файлу поширюється і різні пристрої.
У Сі відсутні засоби введення-виведення. Усі операції вводу-виводу реалізуються за допомогою функцій, що знаходяться в бібліотеці Сі. Бібліотека Сі підтримує три рівні введення-виведення:

- потокове введення-виведення;
- введення-виведення нижнього рівня;
- введення-виведення для консолі та портів (залежить від ОС).

Потокове введення-виведення

На рівні потокового введення-виведення обмін даними проводиться побайтно, тобто за одне звернення до пристрою (файлу) проводиться зчитування або запис фіксованої порції даних (512 1024 байта). При введенні з диска або при зчитуванні файлу дані поміщаються в буфер ОС, потім побайтно або порціями передаються в програмі користувача. При виведенні файл дані накопичуються в буфері, а при заповненні буфера записуються у вигляді єдиного блоку на диск. Буфери ОС реалізуються як ділянок основний пам'яті . Функції бібліотеки Сі, що підтримують обмін, з даними на рівні потоку, дозволяють обробляти дані різних розмірів і форматів.
Потік – це файл разом із наданими засобами буферизації. При роботі з потоком можна:
1)Відкривати та закривати потоки (пов'язувати покажчики на потік із конкретними файлами);

2)вводить та виводити рядок, символ, форматовані дані, порцію даних довільної довжини;
3)аналізувати помилки введення-виведення та досягнення кінця файлу;
4)керувати буферизацією потоку та розміром буфера;
5)отримувати та встановлювати покажчик поточної позиції у файлі;
Функції бібліотеки вводу-виводу знаходяться в заголовку <stdio.h>.

Відкриття та закриття потоку

Перш ніж почати працювати з потоком, його треба ініціювати, тобто відкрити. При цьому потік зв'язується зі структурою певного типу FILE, визначення якої знаходиться у файлі бібліотеки <stdio.h>. У структурі знаходиться покажчик на буфер, покажчик на поточну позицію файлу і т. п. При відкритті потоку повертається покажчик на потік, тобто на об'єкт типу FILE.

```
#include <stdio.h>;
. . . . . .
FILE * fp;
. . . . . ..
fp = fopen ("t.txt", "r");
```

де fopen(<ім'я_файлу>,<режим_відкриття>) - функція для ініціації файлу.
Існують такі режими для відкриття файлу:
”w” - відкрити файл для запису, якщо файл існує, він стирається;
”r” – відкрити файл для читання;
”a” - відкрити файл для додавання, якщо файл існує, він не стирається і можна писати в кінець файлу;
”w+” - відкрити файл для запису та виправлення, якщо файл існує, то він стирається, а далі можна і читати, і писати, розміри файлу можна збільшувати;
”r+” - відкрити файл для читання та запису, але збільшити розмір файлу не можна;
”a+” - відкрити файл для додавання, тобто можна читати і писати, в тому числі і в кінець файлу.
Потік можна відкрити у текстовому (t) або двійковому (b) режимі. За замовчуванням – текстовий режим. У певному вигляді режим вказується так: ”r+b”або ”rb” - двійковий (бінарний) режим.
Приклад:

```
if ((fp=fopen("t.txt", "w")==NULL)
{
perror("\nпомилка при відкритті файлу"); // виводить рядок символів з повідомленням // про помилку
exit(0);
}
```

Після роботи з файлом його треба закрити

```
fclose(<покажчик_на_потік>);
```

Блокове введення-виведення

Для блокового введення та виведення використовуються функції:
1)int fread(void *ptr, int size, int n, FILE *fp) , де

void *ptr - покажчик на область пам'яті, в якій розміщуються дані, що зчитуються з файлу;
int size - розмір одного елемента, що зчитується;
int n - кількість елементів, що зчитуються;
FILE *fp - покажчик на файл, з якого виконується зчитування.
У разі успішного зчитування інформації, функція повертає кількість прочитаних елементів (а не байтів), інакше повертає EOF.
2) int fwrite(void *ptr, int size, int n, FILE *fp) , де
void *ptr - покажчик на область пам'яті, в якій розміщуються дані, що записуються у файл;
int size - розмір одного записуваного елемента;
int n - кількість записуваних елементів;
FILE *fp - покажчик на файл, у якому виконується запис.
У разі успішного запису інформації, функція повертає кількість записаних елементів, інакше повертає EOF.
Приклад:

```
. . . .. . . ..
STRUCT EMPLOYEE
{
char name [40];
char post [40];
float rate;
};
void main()
{
FILE *f; // покажчик пов'язаний із файлом
EMPLOYEE e; // змінна
EMPLOYEE mas[10] // масив
//відкриваємо файл
if ((f=fopen("f.dat", "wb")==NULL) exit(1); // якщо при відкритті файлу виникає
                          //помилка, виходимо з функції
int i;
for(i=1; i<=10;i++)
{
//формуємо запис e
printf("name="); scanf("%s",&e.name);
printf("post="); scanf("%s",&e.post);
printf("rate="); scanf("%f",e.rate);
// записуємо запис e файл
fwrite(&e, sizeof(EMPLOYEE),1,f);
if (ferror(f)==NULL) exit(2);
}
```

```
fclose(f);
//читання записів із файлу
if ((f=fopen("f.dat", "rb")==NULL) exit(3); // якщо при відкритті файлу виникає
                          //помилка, виходимо з функції
i=0;
while(!feof(f)&&i<=10)
{
fread(&mas[i], sizeof(EMPLOYEE),1,f);
i++;
}
fclose(f);
}
```

Нагору