Лабораторна робота №1

Теоретичні відомості

Ранні версії мов програмування містили тільки прості вбудовані типи даних - цілі, дійсні, логічні і т. п (так звані прості або атомарні типи). Ці дані можна було організовувати в масиви. Для обчислювальних задач цього було достатньо, однак поступово комп'ютери стали використовувати для обробки текстів, графічних зображень, ведення баз даних і т. п.

Таким чином, різноманітність оброблюваної інформації привело до необхідності створення програмістом власних складових типів даних - структур. Структури дозволяють поєднувати різнорідні дані - числові дані, масиви, рядки, самі структури і т. і. З структур можна утворювати масиви.

Структура - це складовий тип даних, побудований з використанням інших типів. Структура складається з полів. Поля (елементи структури) - змінні або масиви стандартного типу (int, char і т.п) або інші, раніше описані структури. Оголошення структури здійснюється за допомогою ключового слова struct, за яким йде її ім'я і далі список елементів, укладених у фігурні дужки. Наприклад структура, що описує дату може бути записана наступним чином:

```
struct date {
 int day;
 int month;
 int year;
};
```

Синтаксис оголошення змінних-структур такий самий, як і змінних інших типів. Наприклад,

```
date days;
```

Для звернення до полів структури використовується оператор **.** (точка). Наприклад, що б записати дату 20 листопада 2019 року в змінну days слід використовувати такий підхід:

```
days.day = 20;
days.month = 11;
days.year = 2019;
```

або можливо задати початкові значення відразу при оголошенні змінної:

```
date days = {20, 11, 2019};
```

Створення масиву структур має такий самий синтаксис, як і масиву елементів атомарного типу:

```
date days_array[20]; // Масив типу date з 20 елементів
date* days_dyn_array = new date[20]; // Динамічний масив
```

Робота з покажчиком на структуру має деякі особливості: формально, для того, що б звернутися до поля структури через покажчик, його необхідно розіменувати за допомогою операції *****, а потім, скористатися оператором **.** (точка). Наприклад:

```
date days;
date* pdays = &days; // Покажчик на структуру
(*pdays).day = 6;
```

Звернення до поля структури шляхом використання оператора розіменування (*****) і точки можна спростити, за рахунок застосування оператора -> (стрілка, вона складається з символу мінус і більше). Оператор стрілка включає в себе операції розіменування і точку. Таким чином, останній приклад може бути перетворений:

```
date days;
date* pdays = &days; // Покажчик на структуру
pdays->day = 6;
```

Функції бібліотеки fstream.h для роботи з бінарними файлами

Для роботи з бінарними файлами в бібліотеці fstream.h реалізовані дві функції: для read і write:

```
ostream& write( const char * s, streamsize n );
istream& read( char * s, streamsize n );
```

Перший параметр char * s є покажчиком на масив даних, який необхідно записати / зчитати з файлу. Другий параметр streamsize n - розмір масиву в байтах, який необхідно зчитати / записати.

Так само слід зазначити, що для коректної роботи цих функцій слід відкривати файл в режимі ios :: binary.

Приклад, написати програму (рис. 1), в якій:

1. Користувач має можливість зчитувати і записувати з бінарного файлу інформацію:
    1. ПІБ співробітника
    2. рік народження
2. Користувач має можливість додавати та видаляти інформацію.

```
#include <iostream>
#include <fstream>
#include <cstring>
#include <windows.h>

using namespace std;

const int maxlen = 255;
#pragma pack(push, 1) //директиви компілятору для вирівнювання полів структур
struct sworker {
    char fio[maxlen];
    int age;
};
#pragma pack(pop) //директиви компілятору для вирівнювання полів структур

sworker arr[maxlen];
sworker arr_out[maxlen];
int worker_index = 0;

int menu();//прототипи функцій
void readFromFile(char* fileName);
void saveToFile(char* fileName);
void addNew();
void del();

int main()
{
    setlocale(LC_ALL, "Russian");//забезпечення використання кирилиці
    SetConsoleOutputCP(1251);
    SetConsoleCP(1251);
    while (1) {   //створення нескінченного циклу з меню вибору, виклик відповідних функцій
        switch (menu()) {
        case 1:
            readFromFile("file.dat");
            break;
        case 2:
            saveToFile("file.dat");
            break;
        case 3:
            addNew();
            break;
        case 4:
            del();
            break;
        case 5:
            return 0;
        default:
            cout << "Невірний вибір" << endl;
        }
    }
}
int menu()//функція показує пункти меню вибору
{
    cout << "\n";
    int ans;
```

```
    cout << "Оберіть\n";
    cout << "1-для зчитування з файла\n";
    cout << "2-для запису в файл\n";
    cout << "3-для додавання запису\n";
    cout << "4-для видалення запису\n";
    cout << "5-для виходу\n";
    cout << "\n";
    cout << "Ваш вибір  ";
    cin >> ans;
    return ans;
}
void saveToFile(char* fileName) //функція, що записує дані у бінарний файл
{
    ofstream f;
    f.open(fileName, ios::binary);
    f.write((char*)arr, sizeof(sworker) * worker_index);
    f.close();
    cout << "Введені дані збережено до файлу\n";
}
void readFromFile(char* fileName) //функція, що зчитує дані з бінарного файлу
{
    ifstream f;
    f.open(fileName, ios::binary);
    if (!f) {
        cout << "Файлу не існує";
    }
    else {
        sworker worker;
        worker_index = 0;
        while (1) {
            f.read((char*)&worker, sizeof(worker));
            if (f.eof())
                break;
            arr_out[worker_index] = worker;
            worker_index++;
        }
        f.close();
        cout << "Дані зчитано з файлу\n";
        for (int i = 0; i < worker_index; i++) {
            cout << i + 1 << "\t" << arr_out[i].fio << "\t" << arr[i].age << endl;
        }
    }
}
void addNew()//функція, що додає(створює) новий запис
{
    cout << "Додавання нового запису\n\n";
    cout << "Запис номер " << worker_index + 1 << "\n";
    cin.ignore();
    cout << "Введіть ПІБ  ";
    cin.getline(arr[worker_index].fio, maxlen);
    cout << "Введіть вік  ";
    cin >> arr[worker_index].age;
    worker_index++;
    cout << "\n";
    for (int i = 0; i < worker_index; i++) {
        cout << i + 1 << "\t" << arr[i].fio << "\t" << arr[i].age << endl;
    }
    cout << "\n";
}

void del()//функція, що видаляє запис
{
    int d;
    cout << "Оберіть номер запису, який необхідно видалити ";
    cin >> d;
    for (int i = d - 1; i < worker_index; i++)
```

```
        { arr[i] = arr[i + 1];}
    worker_index--;

    cout << "\n";
    for (int i = 0; i < worker_index; i++) {
        cout << i + 1 << "\t" << arr[i].fio << "\t" << arr[i].age << endl;
    }
    cout << "\n";
}
```

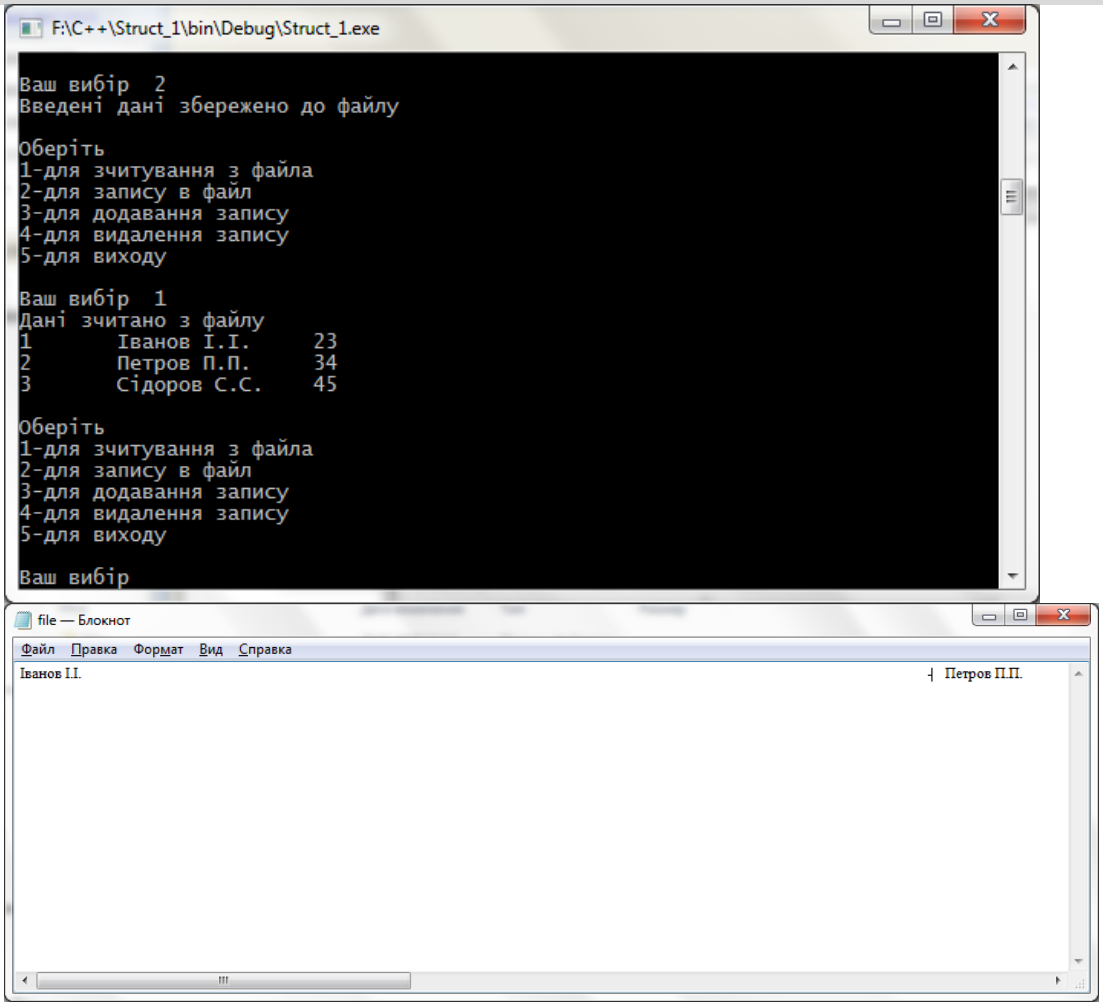

Рис.1 - Приклад роботи програми

Лабораторна робота №1

Варіант 1.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про службовця: а) ім'я, б) прізвище, в) зарплатня, г) зміна.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 2.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про потяг: а) номер, б) час відправлення з місця формування, в) час прибуття в пункт призначення, г) тип вагонів .
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 3.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про товар: а) найменування, б) ціна, в) кількість, г) термін поставки.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 4.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про процесор: а) виробник, б) тактова частота в гігагерцах, в) номінальне напруження, г) тип.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 5.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про книгу: а) автори, б) видавництво, в) кількість томів, г) бібліотечний шифр.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 6.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про деталь: а) вид деталі, б) матеріал, в) вартість, г) вага.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 7.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про автомобіль: а) виробник, б) марка, в) рік випуску, г) пробіг.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 8.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про багаж авіапасажира: а) прізвище власника, б) вага, в) кількість місць, г) рейс.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію.**
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.

Варіант 9.

Написати програму, в якій

1. Користувач має можливість записувати та зчитувати з бінарного файлу інформацію про місто: а) назва, б) країна, в) регіон (область), г) кількість мешканців.
2. Користувач має можливість додавати, видаляти, змінювати та **сортувати по п.1.а інформацію**.
3. Для виконання усіх дій, що описані вище, використовуються спеціально написані для цього функції.
4. Після виконання операції інформація о ній протоколюється (як на екран, так і в файл), таким чином, щоб користувач мав змогу бачити всю історію виконання операцій.