**Лабораторна робота №17. Розробка програм із використанням класів потоків введення-виведення**

**Мета:** Набуття навичок в розробці програм, де використовується множинне наслідування.

**На період** карантину в дистанційній формі навчання результати надсилати на електронну адресу викладача [**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)у вигляді cpp- та h- файлів з іменем у форматі

**<Номер групи><Номер лабораторної>[-<Номер завдання>] <Прізвище англійською>**

та викладати на платформу коледжу у власний репозиторій, під’єднаний до репозиторію викладача.

Приклад іменування файлів, IPZ-31LAB11-1KRASNOPOROV.cpp, IPZ-31LAB11-1KRASNOPOROV.h.

При відсутності можливості доступу до Інтернету текст програми набрати в Блокноті або WordPad Windows та надіслати на електронну адресу викладача.

**Строк виконання цієї роботи 21.05.2020**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача.

**Завдання**

1. Опрацювати теоретичні відомості та переглянути лекцію Тема16.Класи потоків введення-виведення
2. Вся ця робота виконується з програмою, яка розроблювалась на ЛР №№ 5 – 15. Перелік завдань, які мають бути реалізовані в програмі надані в ЛР№№ 5 – 15, а також вимоги до оформлення програми надані в завданні до ЛР№11.
3. Створити нову позицію меню для запису до формування бінарного файлу та його читання "71 Test write – read binary ". Ці дії оформити функцією, яка приймає ім’я файлу , і викликається з цієї позиції меню. Ця функція повинна викликати наявну функцію введення інформації по студенту та записувати її, використовуючи write(), після завершення введення прочитати всі дані за допомогою read() і вивести їх на консоль.

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

Функції write() і read() зручно використовувати для організації прямого доступу до даних у файлі. Але для цього потрібні також функції члени, які дозволяють переміщати покажчик потоку в будь-яке місце файлу. Ці функції застосовуються при запису даних у файл та при їх читанні з файлу і мають відповідно вигляд:

in.seekp(n, dir);

in.seekg(n, dir);

де **in** — ім’я відповідного потоку (введення або виведення);

**n** — параметр, що вказує кількість байт, на яку треба перемістити покажчик потоку;

**dir** — необов’язковий параметр, що вказує на спосіб переміщення покажчика і приймає одне зі значень:

**ios::beg** — переміщення від початку файлу;

**ios::cur** — переміщення від поточної позиції;

**ios::end** — переміщення від кінця файлу.

Якщо параметр **dir** відсутній, то переміщення покажчика здійснюється з початку файлу.

Розглянемо програму, в яких використовуються методи прямого доступу до даних у файлі.

**Приклад**. Читання/запис масиву об’єктів у файл. Функції write(), read()

У прикладі використовуються функції write(), read() для роботи зі об’єктами класу BOOK, а саме:

* запис масиву об’єктів класу BOOK у файл;
* читання масиву об’єктів класу BOOK з файлу.

**#include <iostream>**

**#include <fstream>**

**#include <string.h>**

**#include <Windows.h>**

**using namespace std;**

**// Клас BOOK**

**class BOOK**

**{public:**

**char title[100]; // Назва книги**

**char author[70]; // Автор**

**int year; // Рік видання**

**float price; // вартість книги**

**};**

**// Запис масиву об’єктів в файл з допомогою функції write()**

**// читання масиву об’єктів з файлу з допомогою функції read()**

**bool Example7(const char \* filename)**

**{**

**// створити масив об’єктів**

**BOOK B[3] =**

**{**

**{ "Title-01", "Author-01", 2005, 100.95 },**

**{ "Title-02", "Author-02", 2008, 90.25 },**

**{ "Title-03", "Author-03", 2002, 180.50 }}**

**;**

**int n = 3; // кількість елементів у масиві**

**BOOK C[3]; // інший масив, з якого буде виконуватись читання**

**int n2; // кількість елементів у масиві C**

**int i;**

**// 1. Запис масиву об’єктів в файл**

**// outF - екземпляр файлу, в який здійснюється запис**

**ofstream outF(filename, ios::out | ios::binary);**

**if (!outF) return false;**

**// записати значення n**

**outF.write((char\*)&n, sizeof(int));**

**// запис масиву B[] в файл wf**

**for (i = 0; i < n; i++)**

**{**

**outF.write((char\*)&(B[i]), sizeof(BOOK));**

**}**

**cout << "Array is written\n" << endl;**

**// після закінчення роботи з файлом його потрібно закрити (обов'язково)**

**outF.close();**

**// 2. Читання масиву структур з файлу**

**// inF - екземпляр файлу, з якого здійснюється читання**

**ifstream inF(filename, ios::in | ios::binary);**

**if (!inF) return false;**

**cout << "Read the array...\n";**

**// Спочатку прочитати кількість записаних структур**

**inF.read((char\*)&n2, sizeof(int));**

**// цикл читання масиву об’єктів в змінну C**

**for (i = 0; i < n2; i++)**

**inF.read((char\*)&(C[i]), sizeof(BOOK));**

**inF.close(); // закрити файл**

**//**  **вивід масиву C на екран**

**cout << "Array C:" << endl;**

**for (i = 0; i < n2; i++)**

**{**

**cout << "Title = " << C[i].title << ", ";**

**cout << "Author = " << C[i].author << ", ";**

**cout << "Year = " << C[i].year << ", ";**

**cout << "Price = " << C[i].price << endl;**

**}**

**}**

**int main ()**

**{system("color F0");**

**string a="book.txt";**

**char c1[50]=" ";**

**strncpy(c1,a.c\_str(),a.length()+1);**

**if (Example7(c1))**

**{cout << "OK" << endl;**

**return 0;**

**}else return 1;**

**}**

Результат роботи

