Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра систем управления

Отчет по лабораторной работе № 8

Тема: «Программирование с использованием структур»

Вариант 7

Выполнил:

студент группы 322441 Калинин В.А.

Проверил:

зам.каф ИТАС Навроцкий А.А.

Минск

2023

Цель работы

Цель работы: изучить правила создания и обработки данных структурного типа, написать и отладить программу с созданием структур.

**Краткие теоретические сведения**

**Объявление и использование структур**

Структура – составной тип данных, в котором под одним именем объединены данные различных типов. Отдельные данные структуры называются полями.

Объявление структуры:

struct им\_структуры

{

тип\_элемента\_1 имя\_элемента\_1; тип\_элемента\_2 имя\_элемента\_2;

...

тип\_элемента\_n имя\_элемента\_n;

} ;

Например:

struct struc1

{

int m1;

double m2, m3;

};

Поля структуры могут быть любого типа, в том числе массивами и структурами.

После фигурной скобки допустимо указывать переменные соответствующего структурного типа:

struct struc1

{

int m1;

double m2, m3;

} a, b ,c;

Объявление переменной структурного типа:

struc1 x;

К отдельным частям структуры можно обращаться через составное имя.

Формат обращения:

имя\_структуры.имя\_поля

или

указатель\_на\_структуру -> имя\_поля

Например, если структура объявлена следующим образом:

struct struc1

{

int m1;

double m2, m3;

} x, \*y;

то обратиться к полю m1 можно (после выделения памяти для y):

x.m1 = 35;

или

(&x)->m1 = 35;

или

y->m1 = 35;

или

(\*y).m1 = 35;

Правила работы с полями структуры идентичны работе с переменными соответствующих типов. Инициализировать переменные-структуры можно путем помещения за объявлением списка начальных значений.

struct struc1

{

int m1;

double m2, m3;

} a = {5, 2.6, 34.2};

В качестве полей могут быть использованы другие структуры.

struct struc1

{

int m1;

double m2, m3;

struct

{

int mm1;

} m4;

} s;

Обращение к полю mm1 в этом случае будет следующим:

s.m4.mm1 = 3;

Если имя структуры не указывается, то такое определение называется

анонимным.

Допустимо использовать операцию присваивания для структур одного типа. Например:

struc1 x, y;

… x = y;

В этом случае все значения полей структуры y копируются в соответствующие поля структуры x.

Из структур, как правило, организуют массивы:

struct struc1

{ int m1;

double m2, m3;

};

…

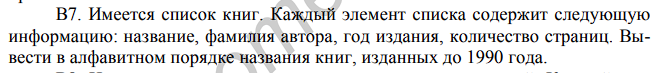
struc1 ms[100]; // Объявление массива структур

…

ms[99].m1 = 56; // Обращение к полю массива структур

Задание

Задание: объявить структуру с заданными полями. Динамически выделить память для хранения списка. Ввести данные. Выполнить задание, результат вывести на экран.



Листинг программы

#include <iostream>

#include <cstring>

#include <algorithm>

#include <iomanip>

using namespace std;

struct Book {

char title[50];

char author[30];

int year;

int pages;

};

bool compareBooks(const Book& book1, const Book& book2) {

return strcmp(book1.title, book2.title) < 0;

}

int main() {

int amount;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Lab 8 var 7 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Enter amount of books: " << endl;

cin >> amount;

Book\* books = new Book[amount];

for (int i = 0; i < amount; ++i) {

cout << "Enter info about book#" << i + 1 << ":\n";

cout << "Title: ";

cin.ignore();

cin.getline(books[i].title, sizeof(books[i].title));

cout << "Author: ";

cin.getline(books[i].author, sizeof(books[i].author));

cout << "Year: ";

cin >> books[i].year;

cout << "Number of pages: ";

cin >> books[i].pages;

}

sort(books, books + amount, compareBooks);

cout << "\n Books that have been published before 1990, ASC" << endl;

for (int i = 0; i < amount; ++i) {

if (books[i].year < 1990) {

cout << setw(20) << books[i].title << setw(20) << books[i].author << setw(7) << books[i].year << setw(7) << books[i].pages << endl;

}

}

delete[] books;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

return 0;

}

Результат работы программы

