Министерство образования Республики Беларусь

Учреждения образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

По дисциплине: «Языки программирования»

Тема: «Классы и объекты С++»

Выполнила:

Студентка 1 курса

Группа ПО-7

Фурсевич Д.С.

Проверил:

Бойко Д. О.

2021

Лабораторная работа №4

**Цель:** получить практические навыки реализации классов на С++.

**Ход работы**

**Задание:**

1. Определить пользовательский класс в соответствии с вариантом задания.

2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.

3. Определить в классе деструктор.

4. Определить в классе компоненты-функции для просмотра и установки полей данных.

5. Определить указатель на компоненту-функцию.

6. Определить указатель на экземпляр класса.

7. Написать демонстрационную программу, в которой создаются и разрушаются объекты пользовательского класса и каждый вызов конструктора и деструктора сопровождается выдачей соответствующего сообщения (какой объект, какой конструктор или деструктор вызвал).

8. Показать в программе использование указателя на объект и указателя на компоненту-функцию.

**Вариант 11:**

Персона

имя – char\*

возраст – int

пол – int(bool)

**Определение пользовательского класса:**

class Person {

private:

char\* m\_name; //имя (указатель на текстовую строку)

int m\_age; //возраст

int m\_floor; //пол

public:

Person(); //конструктор без параметров

Person(char\* name, int age, int floor); //конструктор с параметрами

Person(const Person &other); //конструктор копирования

~Person(); //деструктор

// для просмотра и установки полей данных

char\* GETName(); //для получения значений поля имя

int GETAge(); //для получения значений поля возраст

int GETFloor(); //для получения значений поля пол

void SETName(char\* name); //для установки значений поля имя

void SETAge(int age); //для установки значений поля возраст

void SETFloor(int floor); //для установки значений поля пол

void SET(char\* name, int age, int floor); // инициализация всех полей объекта

void Show(); //просомтр полей объекта

Person::Person(); //конструктор по умолчанию

Person::Person(char\* name, int age, int floor)

: m\_name(name), m\_age(age), m\_floor(floor); // прямая инициализация переменных-члена класса

Person::Person(const Person &other); //конструктор копирования

Person::~Person(); //деструткор

};

**Реализация конструкторов и деструктора:**

Person::Person() //конструктор по умолчанию

{

cout << "Вызвался конструктор без параметров для объекта " << this <<endl;

}

Person::Person(char\* name, int age, int floor) // прямая инициализация переменных-члена класса

{

m\_name = name;

m\_age = age;

m\_floor =floor ;

cout << "Вызвался конструктор с параметрами для объекта " << this<<endl;

}

Person::Person(const Person &other) { //конструктор копирования

m\_name = new char[100];

m\_name = other.m\_name;

m\_age = other.m\_age;

m\_floor = other.m\_floor;

cout << "Конструктор копирования " << this << endl;

}

Person:: ~Person() { //деструктор

cout << "Вызвался деструктор для объекта " << this<< endl;

}

**Использование указателя на объект и на функцию**

array[0].SET(name, 12, 1);

Person \*point = &array[0]; //указатель на объект класса

point->Show();// вывод данных через указатель

//использование указателя на компонентную функцию

void(Person::\*pf)();//определение указателя на компоненту-функцию

pf = &Person::Show; //настройка указателя

cin.getline(name, 100);

Person \*example = new Person(name, 23, 1);

(example->\*pf)(); //вывод объекта

**Текст программы:**

**class.h:**

#pragma once

#include <iostream>

using namespace sts;

class Person {

private:

char\* m\_name; //имя (указатель на текстовую строку)

int m\_age; //возраст

int m\_floor; //пол

public:

Person(); //конструктор без параметров

Person(char\* name, int age, int floor); //конструктор с параметрами

Person(const Person &other); //конструктор копирования

~Person(); //деструктор

// для просмотра и установки полей данных

char\* GETName(); //для получения значений поля имя

int GETAge(); //для получения значений поля возраст

int GETFloor(); //для получения значений поля пол

void SETName(char\* name); //для установки значений поля имя

void SETAge(int age); //для установки значений поля возраст

void SETFloor(int floor); //для установки значений поля пол

void SET(char\* name, int age, int floor); // инициализация всех полей объекта

void Show(); //просомтр полей объекта

Person::Person(); //конструктор по умолчанию

Person::Person(char\* name, int age, int floor)

: m\_name(name), m\_age(age), m\_floor(floor); // прямая инициализация переменных-члена класса

Person::Person(const Person &other); //конструктор копирования

Person::~Person(); //деструткор

};

**realization.cpp:**

#include "class.h"

Person::Person() //конструктор по умолчанию

{

cout << "Вызвался конструктор без параметров для объекта " << this <<endl;

}

Person::Person(char\* name, int age, int floor) // прямая инициализация переменных-члена класса

{

m\_name = name;

m\_age = age;

m\_floor =floor ;

cout << "Вызвался конструктор с параметрами для объекта " << this<<endl;

}

Person::Person(const Person &other) { //конструктор копирования

m\_name = new char[100];

m\_name = other.m\_name;

m\_age = other.m\_age;

m\_floor = other.m\_floor;

cout << "Конструктор копирования " << this << endl;

}

Person:: ~Person() { //деструктор

cout << "Вызвался деструктор для объекта " << this<< endl;

}

char\* Person::GETName() { //для получения значений поля имя

return m\_name;

}

int Person::GETAge() { //для получения значений поля возраст

return m\_age;

}

int Person::GETFloor() { //для получения значений поля пол

return m\_floor;

}

void Person::SETName(char\* name) { //для установки значений поля имя

m\_name = name;

}

void Person::SETAge(int age) { //для установки значений поля возраст

m\_age = age;

}

void Person::SETFloor(int floor) { //для установки значений поля пол

m\_floor = floor;

}

void Person::SET(char\* name, int age, int floor) { // инициализация всех полей объекта

m\_name = name;

m\_age = age;

m\_floor = floor;

}

void Person::Show() { //просмотр полей объекта

cout << "Имя человека " << m\_name << endl;

cout << "Возраст " << m\_age << endl;

cout << "Пол " << m\_floor << endl << endl;

}

void Person::View(Person a) {

a.Show();

}

**demonstration.cpp:**

#include "class.h"

Person fun () {

//cout << "Вызвалась функция fun" << endl;

char\*name = new char[100];

cin.getline(name, 100);

Person p(name, 45, 0);

//cout << "Конец функции fun" << endl;

delete[]name;

return p;

}

int main()

{

setlocale(LC\_ALL, "ru");

char\* name = new char[100];

Person array[1];//использование статической памяти

cin.getline(name, 100);

array[0].SET(name, 12, 1);

array[0].Show();

cout << endl;

Person \*point = &array[0]; //указатель на объект класса

point->Show();// вывод данных через указатель

/\*

point->SETFloor(0);

cout << "указатель: " << endl;

point->Show();

cout << "объект класса: " << endl;

array[0].Show();\*/

Person \*array\_1 = new Person[2]; //использование динамической памяти

cin.getline(name, 100);

array\_1 ->SET(name, 34, 1);

array\_1[0].Show();

cout << endl;

cin.getline(name, 100);

(array\_1 + 1)->SET(name, 65, 0);

(array\_1 + 1)->View(array\_1[1]); //вызов конструктора копирования, когда объект передаётся функции по значению

cout << endl;

delete[] array\_1;

//использование указателя на компонентную функцию

void(Person::\*pf)();//определение указателя на компоненту-функцию

pf = &Person::Show; //настройка указателя

cin.getline(name, 100);

Person \*example = new Person(name, 23, 1);

(example->\*pf)(); //вывод объекта

Person example\_2 = Person(\*example);//вызов конструктора копирования при использовании объекта для инициализации другого объекта

example\_2.Show();

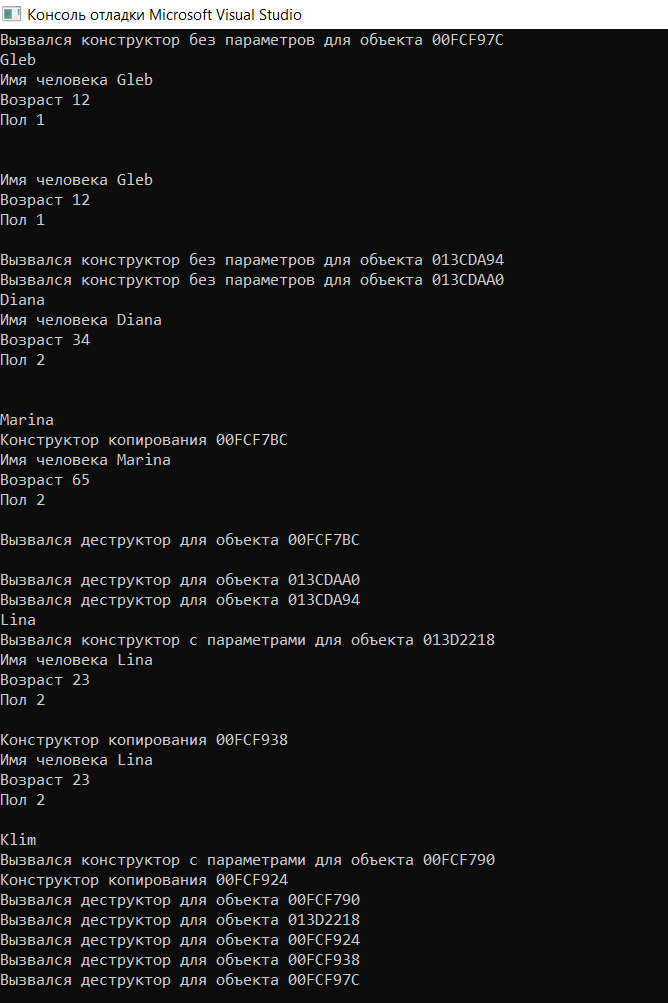
Person example\_3 = fun();//вызов конструктора копирования при построении временного объекта как возвращаемого значения функции

delete example;

delete[] name;

}

**Результаты:**



**Вывод:** получила практические навыки реализации классов на С++.