Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №1

По дисциплине: «Языки программирования»

Тема: «Классы и объекты в С++»

Выполнила:

Савицкая А.Ю., ПО-7

Проверил:

Бойко Д.О.

Брест 2021

**Цель работы:** получить практические навыки реализации классов на С++. Написать программу, в которой создаются и разрушаются объекты, определенного пользователем класса. Выполнить исследование вызовов конструкторов и деструкторов.

**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.

АДРЕС

имя – char\*

улица – char\*

номер дома – int

Задание 1.

Определение пользовательского класса с комментариями:

#pragma once

#include <iostream>

using namespace std;

class Person { //по умолчанию private

char\* Person\_name;

char\* Street;

int House;

public:

Person(); //конструктор без параметров

Person(char\* name, char\* streeet, int number); //конструктор с параметрами

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

~Person(); //деструктор

void set\_name(char\* name); //функция для установки значений поля имя

void set\_streeet(char\* streeet); //функция для установки значений поля возраст

void set\_number(int number); //функция для установки значений поля пол

void set(char\* name, char\* streeet, int number); //функция для установки значений полей

void show(); //функция для просмотра данных

void view(Person a); //функция для просмотра данных (нужна, чтобы вызвать констурктор копирования, когда объект передается функции по значению

};

Задание 2.

Реализация конструкторов (без параметров, с параметрами, копирования):

Person::Person(char\* name, char\* streeet, int number) {

Person\_name = name;

Street = streeet;

House = number;

cout << " Конструктор с параметрами " << this << endl;

}

Person::Person() {

cout << " Конструктор без параметров " << this << endl;

}

Person::Person(const Person& copyPerson) {

Person\_name = copyPerson.Person\_name;

House = copyPerson.House;

Street = copyPerson.Street;

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

}

Задание 3.

Реализация деструктора:

Person::~Person() {

cout << " Деструктор " << this << endl;

}

Задание 4, 5, 6.

Определить в классе компоненты-функции для просмотра и установки полей данных, указатель на компоненту-функцию, указатель на экземпляр класса.

а) arr1[1].set(name, street, number);

Person\* p = &arr1[1];//указатель на экземпляр класса

p->show();

б) Person\* object1 = new Person(name, streeet, 12);

void(Person::\* pfunction)() = &Person::show; // создание указателя на компоненту-функцию

(object1->\*pfunction)(); //вывод данных через указатель

Задание 7, 8.

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).

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

Текст программы:

1. class1.h

#pragma once

#include <iostream>

using namespace std;

class Person { //по умолчанию private

char\* Person\_name;

char\* Street;

int House;

public:

Person(); //конструктор без параметров

Person(char\* name, char\* streeet, int number); //конструктор с параметрами

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

~Person(); //деструктор

void set\_name(char\* name); //функция для установки значений поля имя

void set\_streeet(char\* streeet); //функция для установки значений поля возраст

void set\_number(int number); //функция для установки значений поля пол

void set(char\* name, char\* streeet, int number); //функция для установки значений полей

void show(); //функция для просмотра данных

void view(Person a); //функция для просмотра данных (нужна, чтобы вызвать констурктор копирования, когда объект передается функции по значению

};

1. realization.cpp

#include "class1.h"

Person::Person(char\* name, char\* streeet, int number) {

Person\_name = name;

Street = streeet;

House = number;

cout << " Конструктор с параметрами " << this << endl;

}

Person::Person() {

cout << " Конструктор без параметров " << this << endl;

}

Person::Person(const Person& copyPerson) {

Person\_name = copyPerson.Person\_name;

House = copyPerson.House;

Street = copyPerson.Street;

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

}

Person::~Person() {

cout << " Деструктор " << this << endl;

}

void Person::set\_name(char\* name) {

Person\_name = name;

}

void Person::set\_streeet(char\* streeet) {

Street = streeet;

}

void Person::set\_number(int number) {

House = number;

}

void Person::set(char\* name, char\* streeet, int number) {

Person\_name = name;

Street = streeet;

House = number;

}

void Person::show() {

cout << "Person's name:" << Person\_name << endl;

cout << "Street:" << Street << endl;

cout << "Gender:" << House << endl;

}

void Person::view(Person a) {

a.show();

}

1. demonstration.cpp

#include "class1.h"

#include <iomanip>

#define \_CRT\_SECURE\_NO\_WARNINGS

//функция для вызова конструктора копирования при построении

//временного объекта как возвращаемого значения функции

Person function(Person& cpPerson) {

Person temp(cpPerson);

char\* name = new char[50];

char\* streeet = new char[60];

cin.getline(name, 50);

cin.getline(streeet, 60);

temp.set\_name(name);

temp.set\_name(streeet);

return temp;

delete[]name;

delete[]streeet;

}

int main() {

setlocale(0, "");

system("color f0");

char\* name = new char[50];

char\* streeet = new char[50];

int number;

//использование статической памяти

Person arr1[2];

cout << setfill('\_') << setw(50) << "\_" << endl;

cout << "Enter the name: ";

cin.getline(name, 50);

cout << "Enter the street: ";

cin.getline(streeet, 60);

cout << "Enter the number of house : ";

cin >> number;

cout << endl;

cout << "Person 1" << endl;

arr1[0].set(name, streeet, number);

arr1[0].show();

//-------------------------

cin.ignore();

cout << setfill('\_') << setw(50) << "\_" << endl;

cout << "Enter the name: ";

cin.getline(name, 50);

cout << "Enter the street: ";

cin >> streeet;

cout << "Enter the number of house: ";

cin >> number;

cout << endl;

cout << endl;

cout << "Person 2" << endl;

arr1[1].set(name, streeet, number);

//через указатель доступ

Person\* p = &arr1[1]; //указатель на экземпляр класса

p->show();

cin.ignore();

cout << setfill('\_') << setw(50) << "\_" << endl;

//-----------------------

//использование динамической памяти

Person\* arr2 = new Person[2];

cout << "Enter the name: ";

cin.getline(name, 50);

cin.getline(streeet, 60);

arr2[0].set(name, streeet, 12);

arr2[0].show();

cout << setfill('\_') << setw(50) << "\_" << endl;

cout << "Enter the name: ";

cin.getline(name, 50);

(&arr1[0])->set(name, streeet, 15);

(&arr1[1])->view(arr1[1]); //вызов конструктора копирования при передаче функции по значению

delete[] arr2;

cout << setfill('\_') << setw(50) << "\_" << endl;

cout << "Enter the name: ";

cin.getline(name, 50);

cin.getline(streeet, 60);

Person\* object1 = new Person(name, streeet, 17);

cout << setfill('\_') << setw(50) << "\_" << endl;

void(Person:: \* pfunction)();

pfunction = &Person::show; //указатель на компоненту-функцию

(object1->\*pfunction)(); //вывод объекта //по указателю из объекта

cout << setfill('\_') << setw(50) << "\_" << endl;

//вызов конструктора копирования при использовании объекта для инициализации другого объекта

Person object2 = Person(\*object1);

object2.set\_number(12);

object2.show();

cout << setfill('\_') << setw(50) << "\_" << endl;

//вызов конструктора копирования при построении временного объекта как возвращаемого значения функции

Person object3 = function(object2);

object3.show();

cout << setfill('\_') << setw(50) << "\_" << endl;

delete[] name;

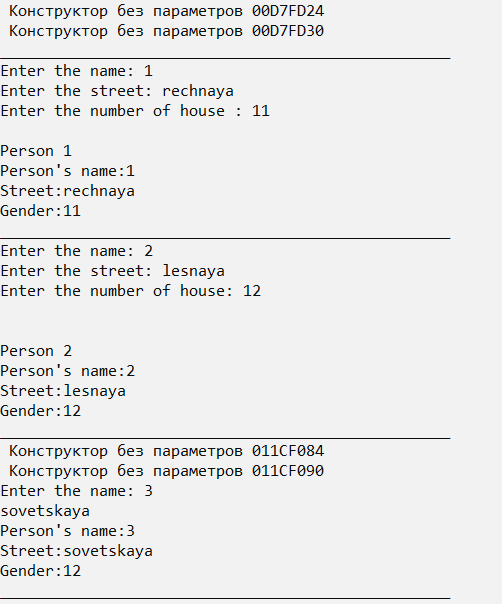
delete object1;

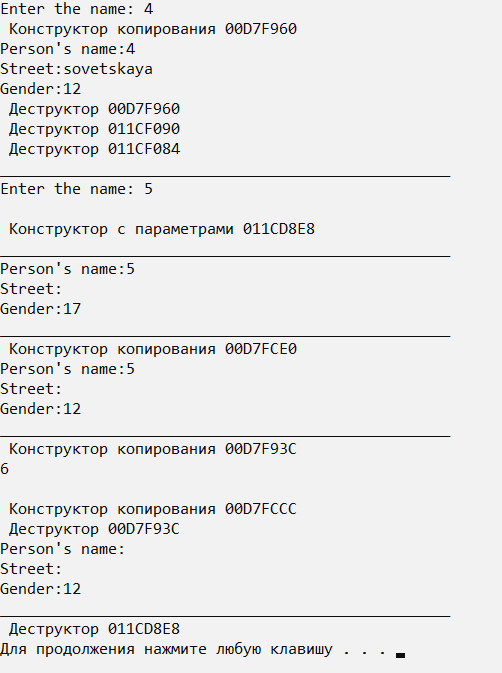
system("pause");

return 0;

}

Результаты тестирования программы:





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