**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

### **Федеральное государственное автономное образовательное учреждение высшего образования**

**«Санкт-Петербургский государственный университет  
аэрокосмического приборостроения»**

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  |  |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ |
| «Множественное наследование в языке С++» |
| по дисциплине: ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4131 |  |  |  | В.А. Алексеев |
|  |  |  | подпись, дата |  | инициалы, фамилия |

1. Задание

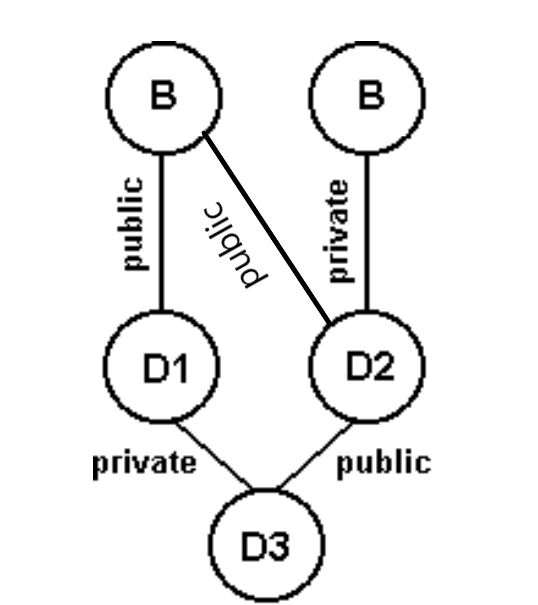
В работе необходимо построить иерархию классов согласно схеме наследования. Каждый класс должен содержать:

- инициализирующий конструктор с выводом значения, полученного в качестве формального параметра

- функцию **show** для вывода значений.

- деструктор.

При определении производного класса нужно указывать базовые классы, которые являются непосредственно родительскими для данного класса. Функция main должна иллюстрировать иерархию наследования. В функции main создать объекты производных классов только «самого нижнего» уровня в иерархии наследования.(Переделанный 1 вариант)



1. Листинг

Header.h

#pragma once

#include<iostream>

using namespace std;

class B1 {

int data;

public:

B1();

B1(int);

void setb1(int);

void show\_B1();

~B1();

};

class B2 {

int data;

public:

B2(int);

void show\_B2();

~B2();

};

class D1 :virtual public B1 {

int data;

public:

D1(int , int );

void show\_D1();

~D1();

};

class D2 :virtual public B1, private B2 {

int data;

public:

D2(int, int, int);

void show\_D2();

~D2();

};

class D3 :private D1, public D2 {

int data;

public:

D3(int , int , int , int , int) ;

void show\_D3() ;

~D3() ;

};

Source.cpp

#include "Header.h"

#include<iostream>

using namespace std;

B1::B1() {

data = 99;

cout << "Конструктор по умолчанию В1" << " = " << data << endl;

}

B1::B1(int a) {

data = a;

cout << "Работает конструктор В1" << " = " << data << endl;

}

void B1::setb1(int a) {

data = a;

cout << "Setter B1 = " << data << endl;

}

void B1::show\_B1() {

cout << "B1 = " << data << endl;

}

B1::~B1() {

cout << "Деструктор В1\n";

}

B2::B2(int a) {

data = a;

cout << "Работает конструктор В2" << " = " << data << endl;

}

void B2::show\_B2() {

cout << "B2 = " << data << endl;

}

B2::~B2() {

cout << "Деструктор В2\n";

}

D1::D1(int a,int b1) : B1(b1)

{

data = a;

cout << "Работает конструктор D1" << " = " << data << endl;

}

void D1::show\_D1() {

cout << "D1 = " << data << endl;

show\_B1();

};

D1::~D1() {

cout << "Деструктор D1\n";

};

D2::D2(int a,int b1,int b2) : B1(b1), B2(b2)

{

data = a;

cout << "Работает конструктор D2" << " = " << data << endl;

setb1(b1);

}

void D2::show\_D2() {

cout << "D2 = " << data << endl;

show\_B1();

show\_B2();

}

D2::~D2() {

cout << "Деструктор D2\n";

}

D3::D3(int a, int b1, int b2, int d1, int d2) : D1(d1, b1), D2(d2, b1, b2)

{

data = a;

cout << "Работает конструктор D3 = " << data << endl;

}

void D3::show\_D3() {

cout << "D3 = " << data << endl;

show\_D1();

show\_D2();

}

D3::~D3() {

cout << "Деструктор D3\n";

}

void main() {

setlocale(LC\_ALL, "rus");

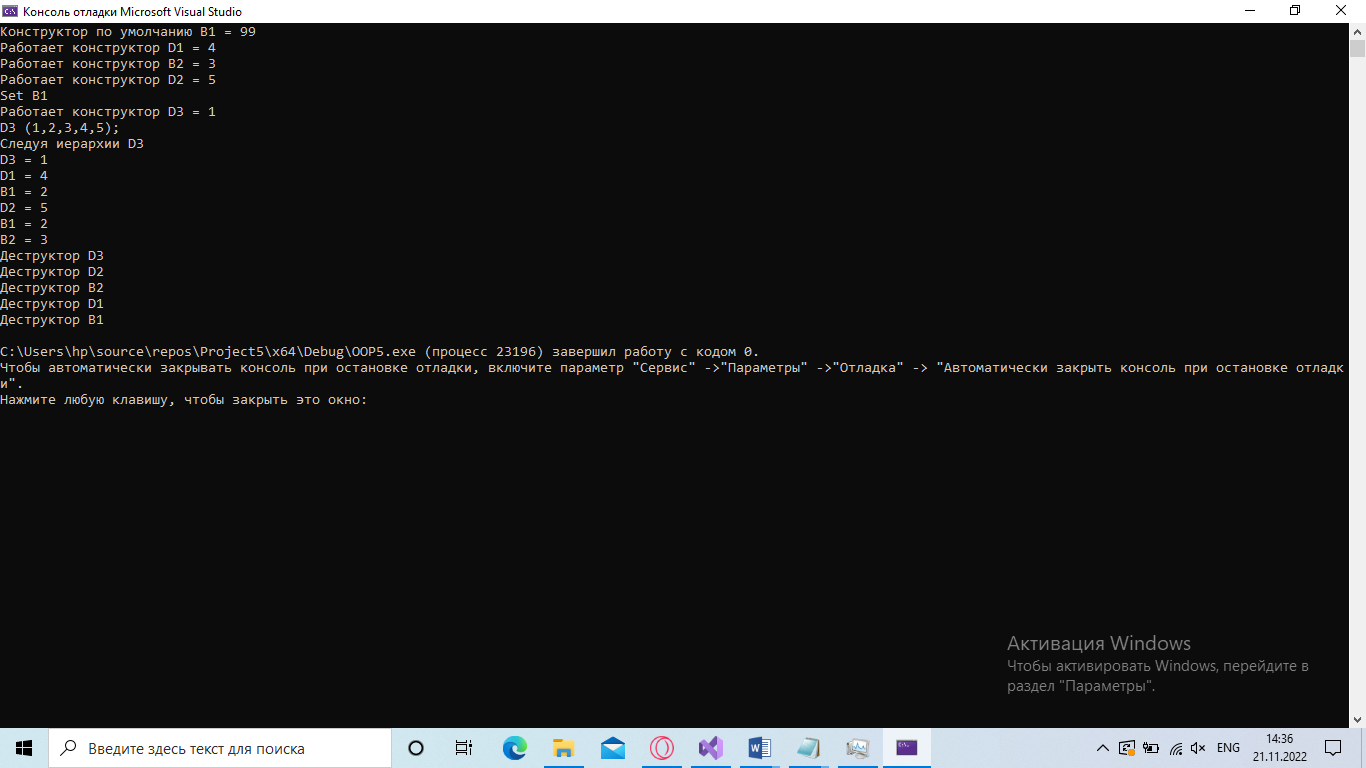
D3 x(1, 2, 3, 4, 5);

cout << "D3 (1,2,3,4,5);\nСледуя иерархии D3\n";

x.show\_D3();

}

1. Результат работы



1. Вывод

Я получил практические навыки использования множественного наследования в языке С++. Закрепил знания по теме: Классы, наследование классов, виртуальные функции, абстрактные классы.