|  |  |
| --- | --- |
| для прик эмбл | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования"Московский технологический университет"МИРЭА | |
| Факультет информационных технологий (ИТ) | |
| Кафедра практической и прикладной информатики | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ** | |
| **по дисциплине** | |
| **«**Объектно-ориентированное программирование**»** | |
|  | |
| Выполнил студент группы ИНБО-04-18 | Корнеев.Д.С. |
| Принял ассистент кафедры | Хлебникова В.Л. |

|  |  |  |
| --- | --- | --- |
| Лабораторные работы выполнены | «13» мая 2019 г. |  |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_201\_\_ г. |  |

Москва 2019

**Содержание**

[Таблица успеваемости 3](#_Toc7282756)

[Практическая работа №1 10](#_Toc7282757)

[Практическая работа №2 13](#_Toc7282758)

[Практическая работа №3 19](#_Toc7282759)

[Практическая работа №4 31](#_Toc7282760)

[Практическая работа №5 37](#_Toc7282761)

[Практическая работа №6 47](#_Toc7282762)

[Практическая работа №7 57](#_Toc7282763)

[Практическая работа №8 67](#_Toc7282764)

[Лабораторная работа №1 70](#_Toc7282765)

[Лабораторная работа №2 77](#_Toc7282766)

[Лабораторная работа №3 82](#_Toc7282767)

[Лабораторная работа №4 89](#_Toc7282768)

[Лабораторная работа №5 93](#_Toc7282769)

[Лабораторная работа №6 97](#_Toc7282770)

[Лабораторная работа №7 102](#_Toc7282771)

[Лабораторная работа №8 105](#_Toc7282772)

## Таблица успеваемости

**Лабораторные занятия**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №/р. | страница в отчете. | Студент выполнил:  (подпись) | Преподаватель принял:  (подпись) | Балл  (максимально возможный) | Балл (Фактический) |
| 1. |  |  |  | 2 |  |
| 2. |  |  |  | 2 |  |
| 3. |  |  |  | 4 |  |
| 4. |  |  |  | 6 |  |
| 5. |  |  |  | 6 |  |
| 6. |  |  |  | 8 |  |
| 7. |  |  |  | 8 |  |
| 8. |  |  |  | 8 |  |
|  |  |  | Итоговый балл: | 44 |  |

**Практические занятия**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №/р. | страница в отчете. | Студент выполнил:  (подпись) | Преподаватель принял:  (подпись) | Балл  (максимально возможный) | Балл (Фактический) |
| 1. |  |  |  | 2 | … |
| 2. |  |  |  | 2 |  |
| 3. |  |  |  | 4 |  |
| 4. |  |  |  | 6 |  |
| 5. |  |  |  | 6 |  |
| 6. |  |  |  | 8 |  |
| 7. |  |  |  | 8 |  |
|  |  |  | Итоговый балл: | 36 |  |

**Файлы используемые в практических работах:**

1. OOP.CPP:

#include "Start.h"

//--------------------------------------------\/-START-\/-----------------------

int main()

{

bool nachalo=true;

while(nachalo!=false){

cout << "\x1B[2J\x1B[H"; // console clear for \*nix version

int Numb = 0;

int SubNumb = 0;

cout << "Введите номер практической: ";

cin >> Numb;

cout << "Введите номер варианта задания: ";

cin >> SubNumb;

cout << "\n";

switch (Numb)

{

//-----------------------------------------\/-ZADANIE\_#1.\*-\/-------------------

case 1:

switch (SubNumb)

{

case 1:

Programma1\_1();

break;

case 2:

Programma1\_2();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#2.\*-\/-------------------

case 2:

switch (SubNumb)

{

case 1:

Programma2\_1();

break;

case 2:

Programma2\_2();

break;

case 3:

Programma2\_3();

break;

case 4:

Programma2\_4();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#3.\*-\/-------------------

case 3:

switch (SubNumb)

{

case 1:

Programma3\_1();

break;

case 2:

Programma3\_2();

break;

case 3:

Programma3\_3();

break;

case 4:

Programma3\_4();

break;

case 5:

Programma3\_5();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#4.\*-\/-------------------

case 4:

switch (SubNumb)

{

case 1:

Programma4\_1();

break;

case 2:

Programma4\_2();

break;

case 3:

Programma4\_3();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#5.\*-\/-------------------

case 5:

switch (SubNumb)

{

case 1:

Programma5\_1();

break;

case 2:

Programma5\_2();

break;

case 3:

Programma5\_3();

break;

case 4:

Programma5\_4();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#6.\*-\/-------------------

case 6:

switch (SubNumb)

{

case 1:

Programma6\_1();

break;

case 2:

Programma6\_2();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#7.\*-\/-------------------

case 7:

switch (SubNumb)

{

case 1:

Programma7\_1();

break;

case 2:

Programma7\_2();

break;

case 3:

Programma7\_3();

break;

case 4:

Programma7\_4();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

//-----------------------------------------\/-ZADANIE\_#8.\*-\/-------------------

case 8:

switch (SubNumb)

{

case 1:

Programma8\_1();

break;

case 2:

Programma8\_2();

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

break;

default:

cout << "Нет такого задания" << "\n";

break;

}

//---------------------------------------------\/-END-\/------------------------

cout <<"\n" << "Задание выполнено." << "\n";

cout << "Закончить выполнение программы или запустить её заново?";

cout << "(1 = запустить заново ИЛИ |остальное| = ЗАКОНЧИТЬ): ";

string EndZnach = "";

cin >> EndZnach;

if (EndZnach == "1"){

cout << "\x1B[2J\x1B[H"; // console clear for \*nix version

nachalo=true;

}else{

nachalo=false;

cout << "\x1B[2J\x1B[H"; // console clear for \*nix version

cout << "Програма завершена. \n \n Программу выполнил: \n Студент: Корнеев Дмитрий \n Группы: ИНБО-04-18"<< "\n";

}

}

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Start.h:

#ifndef Start

#define Start

#include <iostream>

#include <string.h>

#include <string>

#include <stdlib.h>

#include <cmath>

#include <vector>

using namespace std;

#include "h/Programms.h"

#endif

1. Programms.h:

#ifndef Programms

#define Programms

#include "Programm1\_1.h"

#include "Programm1\_2.h"

#include "Programm2\_1.h"

#include "Programm2\_2.h"

#include "Programm2\_3.h"

#include "Programm2\_4.h"

#include "Programm3\_1.h"

#include "Programm3\_2.h"

#include "Programm3\_3.h"

#include "Programm3\_4.h"

#include "Programm3\_5.h"

#include "Programm4\_1.h"

#include "Programm4\_2.h"

#include "Programm4\_3.h"

#include "Programm5\_1.h"

#include "Programm5\_2.h"

#include "Programm5\_3.h"

#include "Programm5\_4.h"

#include "Programm6\_1.h"

#include "Programm6\_2.h"

#include "Programm7\_1.h"

#include "Programm7\_2.h"

#include "Programm7\_3.h"

#include "Programm7\_4.h"

#include "Programm8\_1.h"

#include "Programm8\_2.h"

#endif

## Практическая работа №1

**Изучение синтаксиса языка C++**

**Цель практической работы**

Целью данной практической работы является знакомство с синтаксисом, основными алгоритмическими конструкциями языка С++, а также приобретение практических навыков разработки объектно-ориентированных программ на языке программирования С++.

**Задачи**

1. Реализовать программу, которая считывает имя и здоровается
2. Реализовать программу, производящую операции над числами. (Сложение/вычитание, целые числа/числа с плавающей запятой)

**Ход работы**

1. Реализовать программу, которая считывает имя и здоровается

ФАЙЛ: Programm1\_1.h

#ifndef Programm1\_1

#define Programm1\_1

void Programma1\_1(){

string name;

cout << "Введите имя: ";

cin >> name;

cout << "Приветствую, " << name << "!" << endl;

}

#endif

1. Реализовать программу, производящую операции над числами. (Сложение/вычитание, целые числа/числа с плавающей запятой)

ФАЙЛ: Programm1\_2.h

#ifndef Programm1\_2

#define Programm1\_2

void Programma1\_2(){

double a = 0;

double b = 0;

cout << "Введите два числа: ";

cin >> a;

cin >> b;

cout << "\n a+b= " << a+b << endl;

cout << " a-b= " << a-b << endl;

}

#endif

**Вывод**

В ходе выполнения практической работы была использована библиотека стандартного ввода и вывода, операторы ввода, вывода, сложения и вычитания. Были достигнуты результаты, изображённые на рисунках (рис. 1 и 2). Также исходный Файл: Programm\_.h программ был помещён на GitHub и доступен по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm1_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm1_2.h>

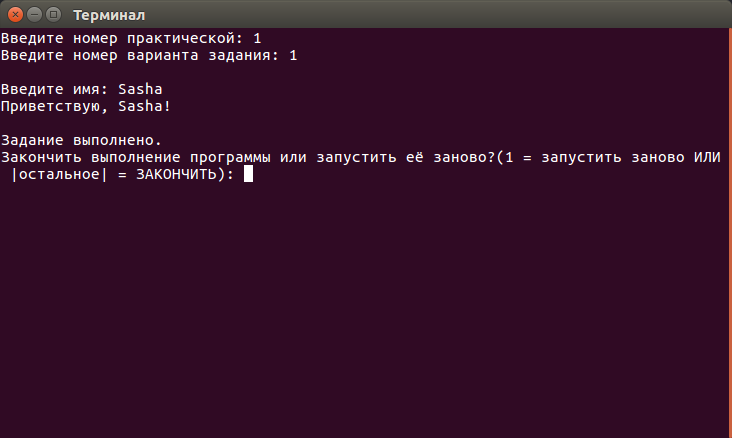


Рис. 1 Результат работы первой программы

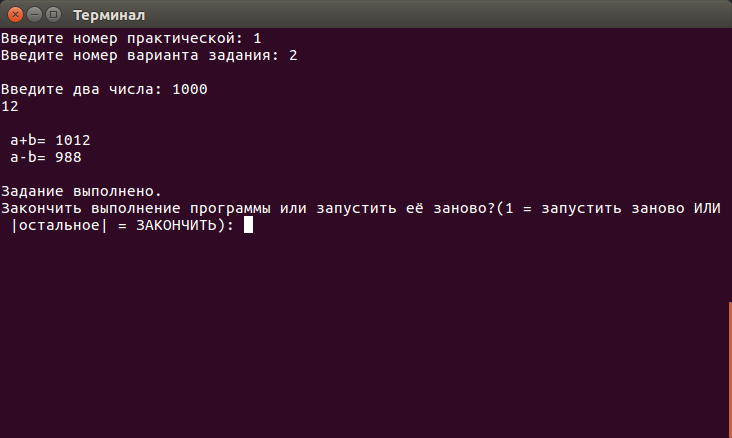


Рис. 2 Результат работы второй программы

## Практическая работа №2

**Решение задач на нелинейные алгоритмические конструкции языка С++**

**Цель практической работы**

Целью данной практической работы является получение практических навыков по разработке нелинейных алгоритмов с использованием конструкции «if-else».

**Ход работы**

1. Вариант 1

ФАЙЛ: Programm2\_1.h

#ifndef Programm2\_1

#define Programm2\_1

void Programma2\_1(){

double a, b, m, n;

cout << "Введите a и b: ";

cin >> a >> b;

if (a >= b) {

n = pow(a - b, 1/3);

} else {

n = a \* a + (a - b) / (sin(a \* b));

}

if (n < b) {

m = (n + a) / (-b) + sqrt(sin(a) \* sin(a) - cos(n));

} else {

if (n == b) {

m = b \* b + tan(n \* a);

} else {

m = b \* b \* b + n \* a \* a;

}

}

// проверка на ошибки при счете

if (isnan(m) || isnan(n) || isinf(m) || isinf(n)) {

cout << "Нельзя посчитать" << endl;

} else {

cout << "a = " << a << endl;

cout << "b = " << b << endl;

cout << "m = " << m << endl;

cout << "n = " << n << endl;

}

}

#endif

1. Вариант 2

ФАЙЛ: Programm2\_2.h

#ifndef Programm2\_2

#define Programm2\_2

void Programma2\_2(){

double a, b, z, t;

cout << "Введите a и b: ";

cin >> a >> b;

if (a < b) {

z = sqrt(abs(a \* a - b \* b));

} else {

z = 1 - 2 \* cos(a) \* sin(b);

}

if (z < b) {

t = pow(z + a \* a \* b, 1/3);

} else {

if (z == b) {

t = 1 - log10(z) + cos(a \* a \*b);

} else {

t = 1 / cos(z \* a);

}

}

// проверка на ошибки при счете

if (isnan(t) || isnan(z) || isinf(z) || isinf(t)) {

cout << "Нельзя посчитать" << endl;

} else {

cout << "a = " << a << endl;

cout << "b = " << b << endl;

cout << "t = " << t << endl;

cout << "z = " << z << endl;

}

}

#endif

1. Вариант 3

ФАЙЛ: Programm2\_3.h

#ifndef Programm2\_3

#define Programm2\_3

void Programma2\_3(){

double a, b, y, t;

cout << "Введите a и b: ";

cin >> a >> b;

if (a <= b) {

y = (a - b) \* (a + b) / ((a + b) \* (a \* a - a \* b + b \* b));

} else {

y = a + pow(log(b), 2);

}

if (y == b) {

t = (2 \* y + sqrt(y \* y - a)) / (2 \* b - sqrt(a \* a - y));

} else {

if (y < b) {

t = sin(y) \* sin(y) + 1 / tan(a - b);

} else {

t = pow(y \* sin(a), 1/3) + 1 / sqrt(y \* cos(b));

}

}

// проверка на ошибки при счете

if (isnan(y) || isnan(t) || isinf(y) || isinf(t)) {

cout << "Нельзя посчитать" << endl;

} else {

cout << "a = " << a << endl;

cout << "b = " << b << endl;

cout << "y = " << y << endl;

cout << "t = " << t << endl;

}

}

#endif

1. Вариант 4

ФАЙЛ: Programm2\_4.h

#ifndef Programm2\_4

#define Programm2\_4

void Programma2\_4(){

double a, x, y, t;

cout << "Enter a and x: ";

cin >> a >> x;

if (a <= x) {

y = a + log(x + a);

} else {

y = sqrt(sin(a \* x));

}

if (a > y) {

t = y / (a - x);

} else {

if (a == y) {

t = y / (a - x) + (a + x) / (y \* y);

} else {

t = tan(a \* x) + cos(2 \* a \* y);

}

}

// проверка на ошибки при счете

if (isnan(y) || isnan(t) || isinf(y) || isinf(t)) {

cout << "Нельзя посчитать" << endl;

} else {

cout << "a = " << a << endl;

cout << "x = " << x << endl;

cout << "y = " << y << endl;

cout << "t = " << t << endl;

}

}

#endif

**Вывод**

В ходе выполнения практической работы была использована библиотека cmath, которая позволила вычислять сложные математические функции, а также быстро совершить проверку на ошибки, и конструкция ветвления «if-else». Были достигнуты результаты, изображённые на рисунках (рис. 3-6). Также Файл: Programm\_.h программ доступен по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm2_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm2_2.h>
3. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm2_3.h>
4. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm2_4.h>

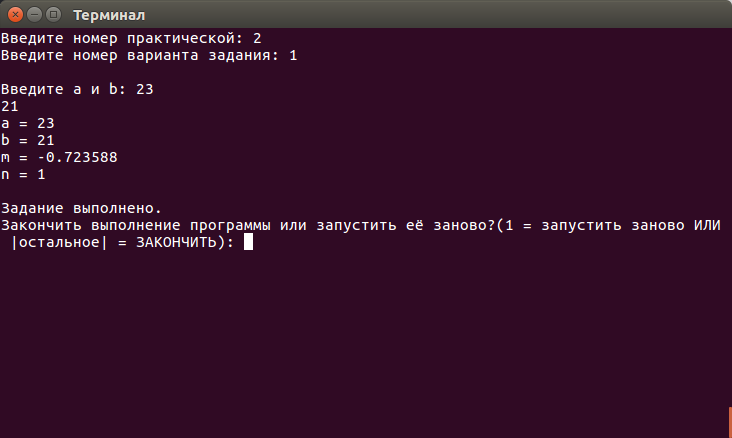


Рис. 3 Результат практической работы №2 вариант 1

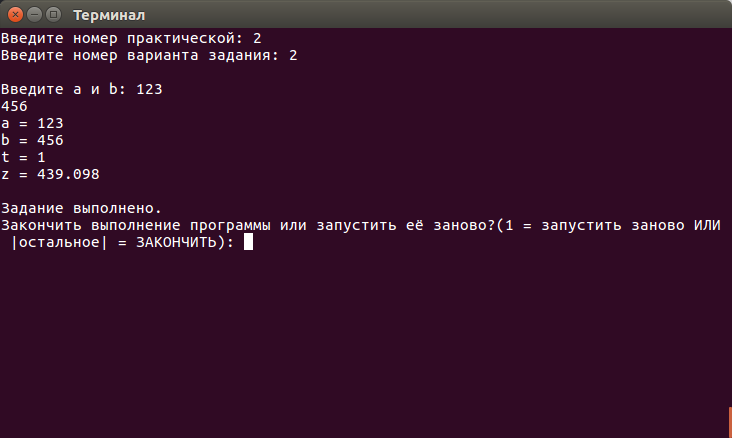


Рис. 4 Результат практической работы №2 вариант 2

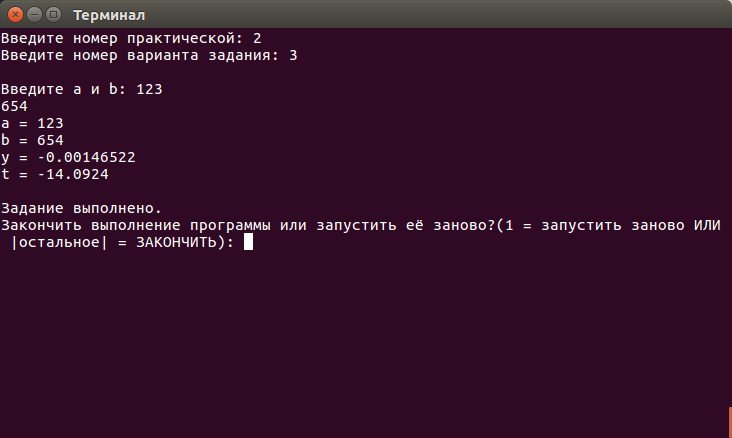


Рис. 5 Результат практической работы №2 вариант 3

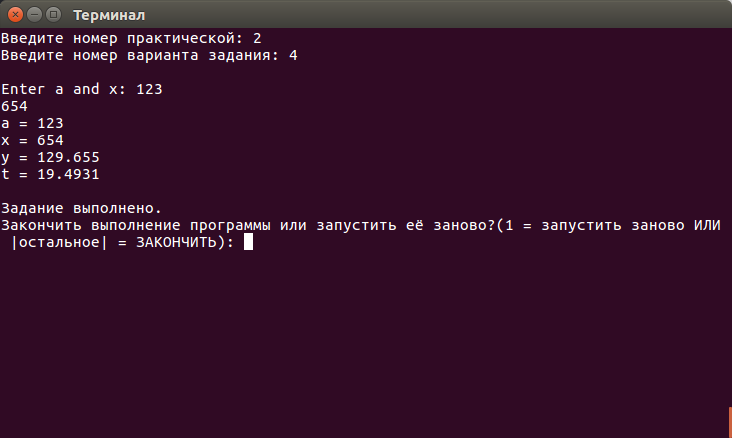


Рис. 6 Результат практической работы №2 вариант 4

## Практическая работа №3

**Классы в языке С++**

**Цель практической работы**

Целью данной практической работы является приобретение практических навыков при разработке классов на языке программирования С++.

**Задачи**

1. Реализовать класс «Книга»
2. Реализовать класс «Группа студентов»
3. Реализовать класс «Автомобиль»
4. Реализовать класс «Вектор»
5. Реализовать класс «Библиотека»

**Ход работы**

1. Реализовать класс «Книга»

ФАЙЛ: Programm3\_1.h

#ifndef Programm3\_1

#define Programm3\_1

class Book

{

private:

string name;

string author;

string publisher;

public:

void define(string na, string au, string pub)//Изменяющий значения

{

name = na;

author = au;

publisher = pub;

}

void display(void)//Отображающий значения

{

cout << "Название: " << name << "; Автор: " << author << "; Издательсво: " << publisher << endl;

}

};

Book book;

string na;

string au;

string pub;

void Programma3\_1(){

cout << "Введите название" << endl;

cin >> na;

cout << "Введите автора" << endl;

cin >> au;

cout << "Введите издательство" << endl;

cin >> pub;

book.define(na,au,pub);

book.display();

}

#endif

1. Реализовать класс «Группа студентов»

ФАЙЛ: Programm3\_2.h

#ifndef Programm3\_2

#define Programm3\_2

class Student

{

private:

string name;//поле имени

string surname;//поле фамилии

string midname;//поле отчества

int mark[5];//поле оценки

public:

void define(string n, string s, string m, int ma[5])//Изменяющий значения

{

name = n;

surname = s;

midname = m;

for (int i = 0; i < 5; i++)

mark[i] = ma[i];

}

void display(void)//Отображающий значения

{

cout << "Имя: " << name << "; Фамилия: " << surname << "; Отчество: " << midname << "; Оценки: " << endl;

for (int i = 0; i < 5; i++) cout << mark[i] << " ";

cout << endl;

}

};

class Group\_S

{

private:

string name;//название группы

int number;//число студентов

Student \*st;//студенты

public:

void define(string n, int num)

{

name = n;

number = num;

st = new Student[num];

string na, s, m;

int ma[5];

for(int i = 0; i<num;i++)

{

cin.ignore();

cout << "Введите имя студента" << endl;

getline(cin,na);

cout << "Введите фамилию студента" << endl;

getline(cin, s);

cout << "Введите отчество студента" << endl;

getline(cin, m);

cout << "Введите оценки" << endl;

for (int j = 0; j < 5; j++) cin >> ma[j];

st[i].define(na,s,m,ma);

}

}

void display(void)//Отображающий значения

{

cout << "Название группы: " << name << "; Количество студентов: " << number <<";" << endl;

for (int i = 0; i < number; i++)

{

st[i].display();

cout << endl;

}

}

void delete\_st(void)//Удаление массива

{

delete []st;

}

};

Group\_S Group;

string n;

int num;

void Programma3\_2(){

cout << "Введите название группы" << endl;

cin >> n;

cout << "Введите количество студентов" << endl;

cin >> num;

Group.define(n,num);

Group.display();

Group.delete\_st();

}

#endif

1. Реализовать класс «Автомобиль»

ФАЙЛ: Programm3\_3.h

#ifndef Programm3\_3

#define Programm3\_3

class Auto//определение класса

{

private:

string brand;//поле класса для марки

string model;//поле класса для модели

string color;//поле класса для цвета

string number;//поле класса для номера

public:

void define(string b, string m, string c, string nn)//метод класса, изменяющий значения полей

{

brand = b;

model = m;

color = c;

number = nn;

}

void display(void)//метод класса, отображающий значения полей

{

cout << "Марка: " << brand << "; Модель: " << model << "; Цвет: " << color << "; Номер: " << number<< endl;

}

};

Auto a;//определение объекта класса Auto

string b,m,c,nn;

void Programma3\_3(){

cout << "Введите марку автомобиля" << endl;

cin >> b;

cout << "Введите модель автомобиля" << endl;

cin >> m;

cout << "Введите цвет автомобиля" << endl;

cin >> c;

cout << "Введите номер автомобиля" << endl;

cin >> nn;

a.define(b,m,c,nn);//вызов метода define()

a.display();//вызов метода display

}

#endif

1. Реализовать класс «Вектор»

ФАЙЛ: Programm3\_4.h

#ifndef Programm3\_4

#define Programm3\_4

class Point//определение класса точки

{

private:

double x;//поле класса x

double y;//поле класса y

double z;//поле класса z

public:

void set(double n\_x, double n\_y, double n\_z)//метод, изменяющий значения

{

x = n\_x;

y = n\_y;

z = n\_z;

}

double get\_x(void)//метод, отображающий x

{

return x;

}

double get\_y(void)//метод, отображающий y

{

return y;

}

double get\_z(void)//метод, отображающий z

{

return z;

}

void display(void)//метод, отображающий значения

{

cout <<"x = " << x << "; y = " << y << "; z = " << z << endl;

}

};

class Vector

{

private:

Point begin;//координаты начала

Point end;//координаты конца

public:

void define(Point bb, Point e)//метод класса, изменяющий значения полей

{

begin.set(bb.get\_x(),bb.get\_y(),bb.get\_z());

end.set(e.get\_x(), e.get\_y(), e.get\_z());

}

void display(void)//значения полей

{

cout << "Координата начала: " << endl;

begin.display();

cout << "Координата конца: " << endl;

end.display();

}

};

Vector v;//определение объекта Vector

int x, y, z;

Point bb,e;

void Programma3\_4(){

cout << "Введите координату начала (x,y,z)" << endl;

cin >> x >> y >> z;

bb.set(x, y, z);

cout << "Введите координату конца (x,y,z)" << endl;

cin >> x >> y >> z;

e.set(x, y, z);

v.define(bb,e);//вызов define()

v.display();//вызов display

}

#endif

Реализовать класс «Библиотека»

ФАЙЛ: Programm3\_5.h

#ifndef Programm3\_5

#define Programm3\_5

class Bookk//определение класса

{

private:

string name;//поле названия

string author;//поле aвтора

string publisher;//поле издательствa

double price;//поле цены

public:

void define(string nnn, string aa, string pu, double p)//метод, изменяющий значения

{

name = nnn;

author = aa;

publisher = pu;

price = p;

}

void display(void)//метод, отображающий значения

{

cout << "Название: " << name << "; Автор: " << author << "; Издательсво: " << publisher << "; Цена: " << price << endl;

}

};

class Library

{

private:

string name;//поле названия

string address;//поле адреса

int number;//поле колиства книг

Bookk \*bookks;//поле книг

public:

void define(string nnn, string aa, int numm)//метод, изменяющий значения

{

name = nnn;

address = aa;

number = numm;

bookks = new Bookk[numm];

string na, au, pu;

int e;

double p;

for (int i = 0; i < numm; i++)

{

cin.ignore();

cout << "Введите название" << endl;

cin >> na;

cout << "Введите автора" << endl;

cin >> au;

cout << "Введите издательство" << endl;

cin >> pu;

cout << "Введите цену" << endl;

cin >> p;

bookks[i].define(na, au, pu, p);

}

}

void display(void)//метод, отображающий значения

{

cout << "Название: " << name << "; Адрес: " << address << "; Количество книг: " << number << endl;

for (int i = 0; i < number; i++)

{

bookks[i].display();

cout << endl;

}

}

void delete\_b(void)//метод удаление массива

{

delete []bookks;

}

};

Library l;//определение объекта класса Library

string nnn, aa;

int numm;

void Programma3\_5(){

cout << "Введите название" << endl;

cin >> nnn;

cout << "Введите адрес" << endl;

cin >> aa;

cout << "Введите количество книг" << endl;

cin >> numm;

l.define(nnn,aa,numm);//вызов метода define()

l.display();//вызов метода display

l.delete\_b();

}

#endif

**Вывод**

Все поставленные задачи были выполнены. Для этого были использованы библиотека STL, перегрузка оператора равенства, шаблоны функций. Все задачи были выполнены в виде много файлового проекта с разбиением классов на файлы объявления и реализации класса. Результаты работы программ приведены в рисунках (рис. 7-11). Файл: Programm\_.h программ также доступен на GitHub по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm3_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm3_2.h>
3. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm3_3.h>
4. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm3_4.h>
5. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm3_5.h>

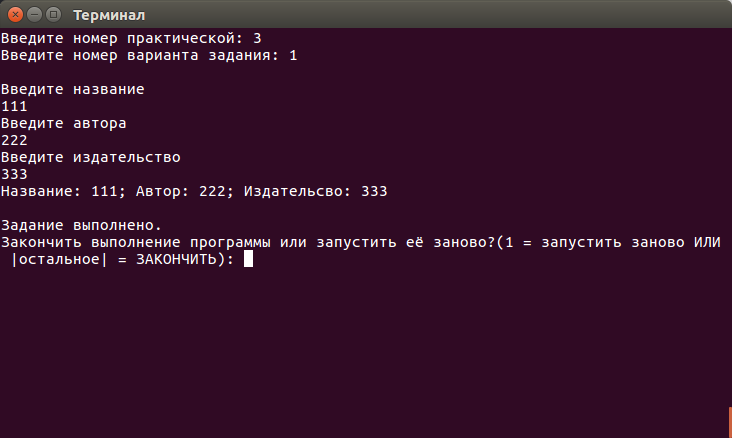


Рис. 7 Результат работы программы №1

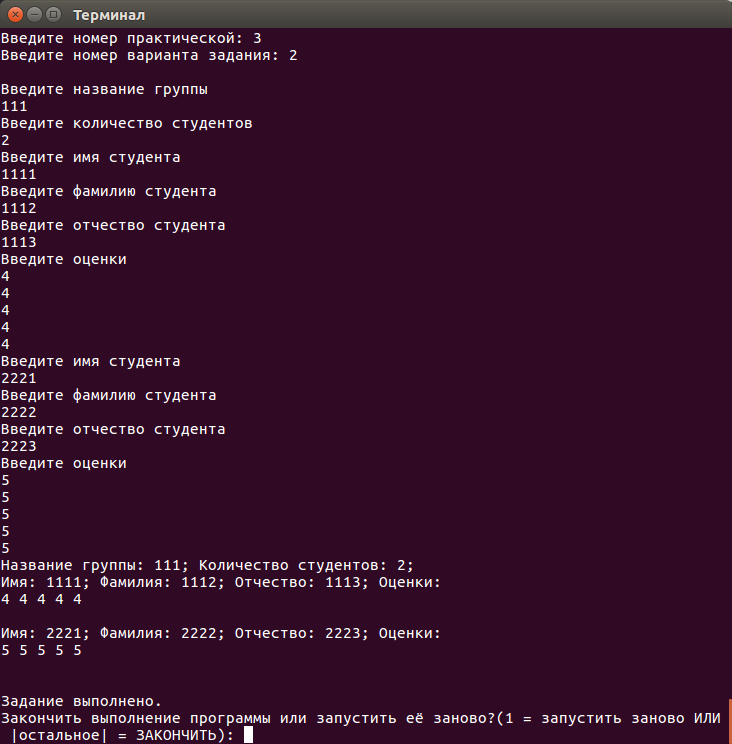


Рис. 8 Результат работы программы №2

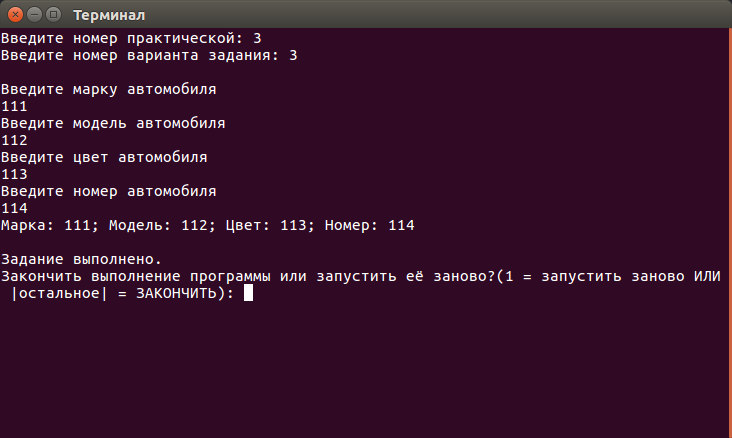


Рис. 9 Результат работы программы №4

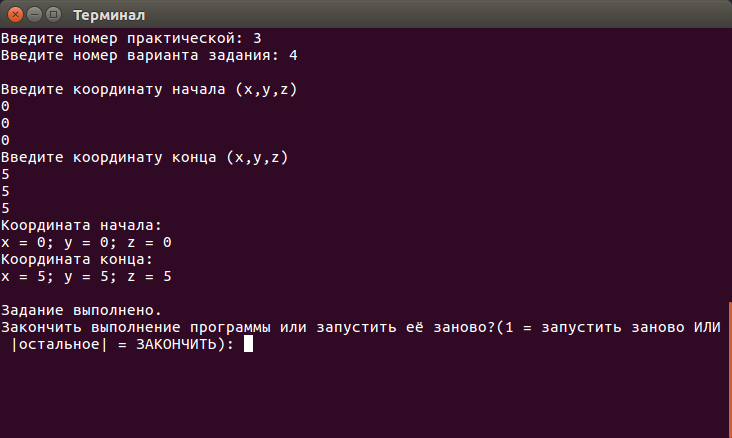


Рис 10 Результат работы программы №5

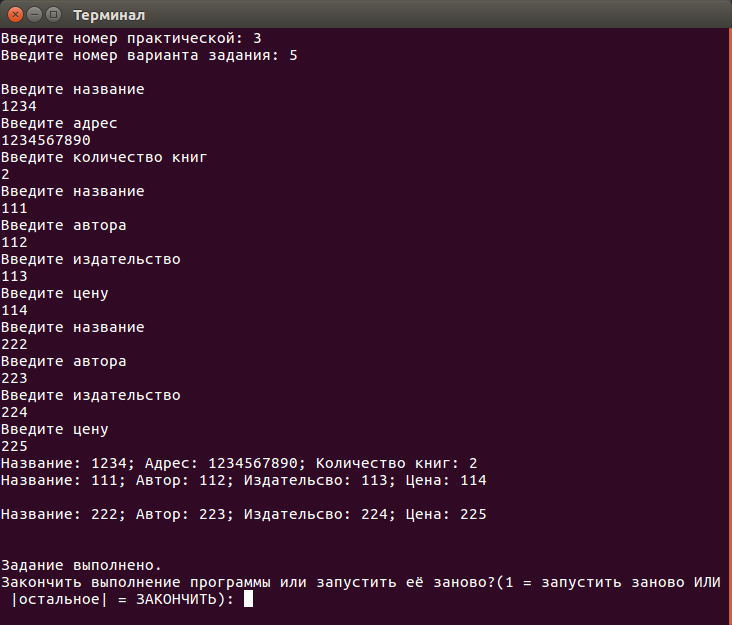


Рис. 11 Результат работы программы №5

## Практическая работа №4

**Динамическое выделение памяти для объектов в языке С++**

**Цель практической работы**

Целью данной практической работы является изучение динамического выделения и удаления памяти, а так же работа с указателями на языке С++.

**Задачи**

1. Объявите указатель на массив типа double и предложите пользователю выбрать его размер. Далее напишите четыре функции: первая должна выделить память для массива, вторая – заполнить ячейки данными, третья – показать данные на экран, четвёртая – освободить занимаемую память. Программа должна предлагать продолжать работу (создавать новые массивы) или выйти из программы.
2. Объявите указатель на массив типа int и выделите память для 12-ти элементов. Необходимо написать функцию, которая поменяет значения четных и нечетных ячеек массива.
3. Объявить и заполнить двумерный динамический массив случайными числами от 10 до 50. Показать его на экран. Для заполнения и показа на экран написать отдельные функции. Количество строк и столбцов выбирает пользователь

**Ход работы**

1. Программа №1

ФАЙЛ: Programm4\_1.h

#ifndef Programm4\_1

#define Programm4\_1

double \*createArray(int); // Создаёт массив

void fillArray(double \*, int); // Заполняет массив случайными числами

void printArray(double \*, int); // Выводит массив на экран

void deleteArray(double \*, int); // Удаляет массив

void Programma4\_1(){

srand(time(NULL));

int size;

cout << "Введите размер массива, чтобы выйти из программы введите число (< 1): ";

cin >> size;

while (size > 0) {

double \*arr = createArray(size);

fillArray(arr, size);

printArray(arr, size);

deleteArray(arr, size);

cout << "Введите размер массива, чтобы выйти из программы введите число (< 1): ";

cin >> size;

cout << endl;

}

}

double \*createArray(int size) {

return new double[size];

}

void fillArray(double \*arr, int size) {

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

arr[i] = rand() % 100;

}

}

void printArray(double \*arr, int size) {

cout << "Array: ";

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

cout << arr[i] << ' ';

}

cout << endl;

}

void deleteArray(double \*arr, int size) {

delete [] arr;

}

#endif

1. Программа №2

ФАЙЛ: Programm4\_2.h

#ifndef Programm4\_2

#define Programm4\_2

void swap(int &a, int &b)//функция для перестановки местами значений переменных

{

int buffer = 0;

buffer = a;

a = b;

b = buffer;

buffer = 0;

}

void Programma4\_2(){

double \*a = new double[12];//объявление указателя на массив и выделение памяти под массив

cout << "Введиет 12 чисел" << endl;

for (int i = 0; i < 12; i++) cin >> a[i];

for (int i = 0; i < 12; i+=2)

{

swap(a[i], a[i+1]);//перестановка элементов

}

cout << "Новый массив: " << endl;

for (int i = 0; i < 12; i++) cout << a[i] << " ";//вывод измененного массива

cout << endl;

delete[] a;//освобождение памяти

}

#endif

1. Программа №3

ФАЙЛ: Programm4\_3.h

#ifndef Programm4\_3

#define Programm4\_3

void fill(int \*\*a, int i, int j)//функция заполнения массива

{

srand(time(0));

for(int l=0;l<i;l++)

for (int c = 0; c < j; c++)

{

a[l][c] = 10+rand()%40;

}

}

void show(int \*\*a, int i, int j)//функция вывода массива

{

for (int l = 0; l < i; l++)

{

for (int c = 0; c < j; c++)

{

cout << a[l][c] << " ";

}

cout << endl;

}

}

void Programma4\_3(){

int i, j;

cout << "Введите кол-во строк: ";//ввод строк

cin >> i;

cout << endl << "Input the number of columns: ";//ввод столбцов

cin >> j;

cout << endl;

int \*\*a = new int\*[i];//создание динамического двумерного массива

for (int c = 0; c < i; c++)

{

a[c] = new int[j];

}

fill(a,i,j);

show(a, i, j);

}

#endif

**Вывод**

Все поставленные задачи были выполнены. При выполнении использовались массивы с динамическим выделением памяти, которые создавались с помощью генератора случайных чисел. Для генератора случайных чисел была подключена библиотека ctime. Результат работы программ приведён на рисунках (рис.12-14). Так же исходный Файл: Programm\_.h программ доступен на GitHub по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm4_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm4_2.h>
3. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm4_3.h>

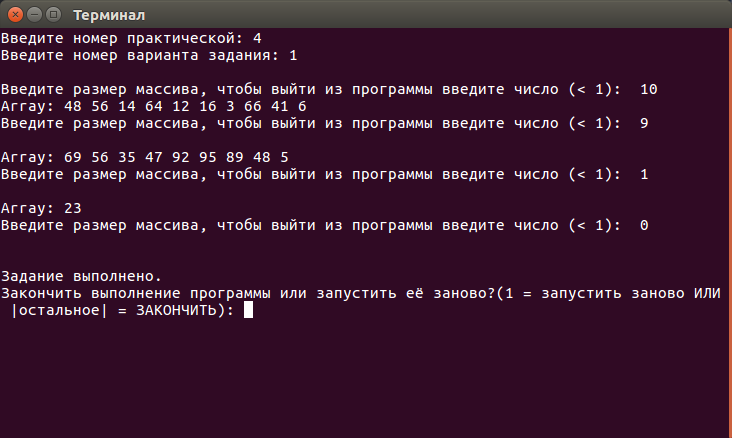


Рис. 12 Результат работы программы №1

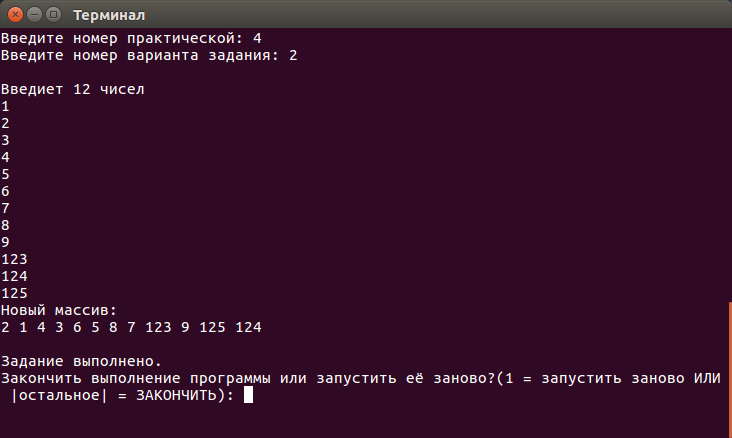


Рис. 13 Результат работы программы №2

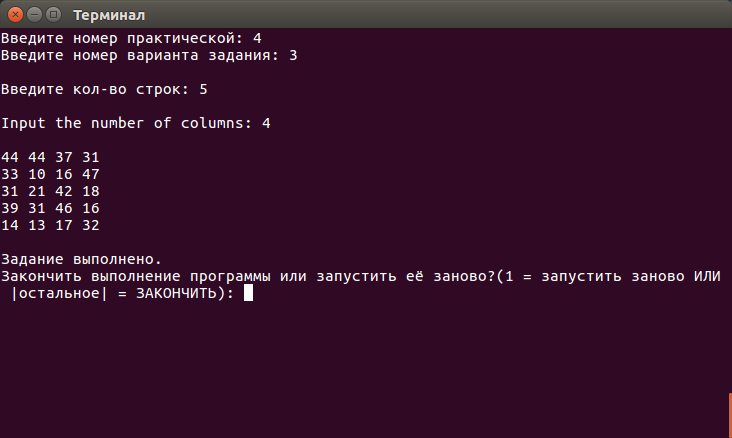


Рис. 14 Результат работы программы №3

## Практическая работа №5

**Перегрузка функций**

**Цель практической работы**

Целью данной практической работы является приобретение практических навыков по программированию перегрузки функций на языке C++.

**Задачи**

1. Реализовать сортировку пузырьком для целых чисел, а затем перегрузить её для дробных.
2. Реализовать сортировку выбором для целых чисел, а затем перегрузить её для дробных.
3. Реализовать сортировку вставками для целых чисел, а затем перегрузить её для дробных.
4. Реализовать программу-калькулятор, работающую с разными типами данных.

**Ход работы**

1. Программа №1

ФАЙЛ: Programm5\_1.h

#ifndef Programm5\_1

#define Programm5\_1

void bubleSort(int \*a, int size)//сортировка пузырьком для целых чисел

{

int buff;

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

for (int j = 0; j < size - 1; j++) {

if (a[j] > a[j + 1]) {

buff = a[j]; //

a[j] = a[j + 1]; // меняем местами

a[j + 1] = buff; // значения элементов

}

}

}

}

void bubleSort(double \*a, int size)//сортировка пузырьком для дробных чисел

{

double buff;

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

for (int j = 0; j < size - 1; j++) {

if (a[j] > a[j + 1]) {

buff = a[j];

a[j] = a[j + 1]; // меняем местами

a[j + 1] = buff; // значения элементов

}

}

}

}

void show1(int \*a, int size)//функция вывода массива для целых чисел

{

cout << "Int array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void show1(double \*a, int size)//функция вывода массива для дробных чисел

{

cout << "Double array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void Programma5\_1(){

int a[10];

double b[10];

srand(time(0));

for (int i = 0; i < 10; i++)//заполнение массивов

{

a[i]= 10 + rand() % 40;

b[i]= (double)(rand() % 1000 - 50) / 100;

}

show1(a, 10);

show1(b, 10);

bubleSort(a,10);

bubleSort(b, 10);

show1(a, 10);

show1(b, 10);

}

#endif

1. Программа №2

ФАЙЛ: Programm5\_2.h

#ifndef Programm5\_2

#define Programm5\_2

void choicesSort(int \*a, int size)//сортировка выбором для целых чисел

{

int buff;

for (int i = 0; i < size; i++)

{

buff = a[0];

for (int j = i + 1; j <size; j++)

{

if (a[i] > a[j])

{

buff = a[i];

a[i] = a[j];

a[j] = buff;

}

}

}

}

void choicesSort(double \*a, int size)//сортировка выбором для дробных чисел

{

double buff;

for (int i = 0; i < size; i++)

{

buff = a[0];

for (int j = i + 1; j <size; j++)

{

if (a[i] > a[j])

{

buff = a[i];

a[i] = a[j];

a[j] = buff;

}

}

}

}

void show2(int \*a, int size)//функция вывода массива для целых чисел

{

cout << "Int array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void show2(double \*a, int size)//функция вывода массива для дробных чисел

{

cout << "Double array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void Programma5\_2(){

int a[10];

double b[10];

srand(time(0));

for (int i = 0; i < 10; i++)//заполнение массивов

{

a[i]= 10 + rand() % 40;

b[i]= (double)(rand() % 1000 - 50) / 100;

}

show2(a, 10);

show2(b, 10);

choicesSort(a,10);

choicesSort(b, 10);

show2(a, 10);

show2(b, 10);

}

#endif

1. Программа №3

ФАЙЛ: Programm5\_3.h

#ifndef Programm5\_3

#define Programm5\_3

void insertSort(int \*a, int size)//сортировка вставками для целых чисел

{

int key, buff;

for (int i = 0; i < size - 1; i++)

{

key = i + 1;

buff = a[key];

for (int j = i + 1; j > 0; j--)

{

if (buff < a[j - 1])

{

a[j] = a[j - 1];

key = j - 1;

}

}

a[key] = buff;

}

}

void insertSort(double \*a, int size)//сортировка вставками для дробных чисел

{

int key;

double buff;

for (int i = 0; i < size - 1; i++)

{

key = i + 1;

buff = a[key];

for (int j = i + 1; j > 0; j--)

{

if (buff < a[j - 1])

{

a[j] = a[j - 1];

key = j - 1;

}

}

a[key] = buff;

}

}

void show3(int \*a, int size)//функция вывода массива для целых чисел

{

cout << "Int array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void show3(double \*a, int size)//функция вывода массива для дробных чисел

{

cout << "Double array:" << endl;

for (int i = 0; i < size; i++) cout << a[i] << " ";

cout << endl;

}

void Programma5\_3(){

int a[10];

double b[10];

srand(time(0));

for (int i = 0; i < 10; i++)//заполнение массивов

{

a[i]= 10 + rand() % 40;

b[i]= (double)(rand() % 1000 - 50) / 100;

}

show3(a, 10);

show3(b, 10);

insertSort(a,10);

insertSort(b, 10);

show3(a, 10);

show3(b, 10);

}

#endif

1. Программа №4

ФАЙЛ: Programm5\_4.h

#ifndef Programm5\_4

#define Programm5\_4

int add(int a,int b)//сложение целых чисел

{

return a + b;

}

double add(double a, double b)//сложение дробных чисел

{

return a + b;

}

int sub(int a, int b)//вычитание целых чисел

{

return (a - b);

}

double sub(double a, double b)//вычитание дробных чисел

{

return a - b;

}

int mul(int a, int b)//умножение целых чисел

{

return a \* b;

}

double mul(double a, double b)//умножение дробных чисел

{

return a \* b;

}

int divi(int a, int b)//деление целых чисел

{

if (b != 0)

return a / b;

else

{

cout << "Деление на 0" << endl;

return -1;

}

}

double divi(double a, double b)//деление дробных чисел

{

if (b != 0)

return a / b;

else

{

cout << "Деление на 0" << endl;

return -1;

}

}

void Programma5\_4(){

int type;

double a, b;

int sw = 1;

cout << "Введите a,b" << endl;

cin >> a >> b;

cout << "1 - '+', 2 - '-', 3 - '\*', 4 - '/', другое - 'exit'" << endl;//выбор

cin >> sw;

while ((sw > 0)&&(sw < 5))

{

switch (sw)

{

case 1://сложение

{

cout << "INT: " << add((int)a, (int)b) << endl << "DOUBLE: " << add(a, b) << endl;

break;

}

case 2://вычитание

{

cout << "INT: " << sub((int)a, (int)b) << endl << "DOUBLE: " << sub(a, b) << endl;

break;

}

case 3://умножение

{

cout << "INT: " << mul((int)a, (int)b) << endl << "DOUBLE: " << mul(a, b) << endl;

break;

}

case 4://деление

{

cout << "INT: " << divi((int)a, (int)b) << endl << "DOUBLE: " << divi(a, b) << endl;

break;

}

default:

cout << "Выход" << endl;

break;

}

cout << "1 - '+', 2 - '-', 3 - '\*', 4 - '/', другое - 'exit'" << endl;//выбор

cin >> sw;

}

cout << "Выход" << endl;

}

#endif

**Вывод**

Все задачи были выполнены. В ходе выполнения была изучена техника перегрузки функций. Так же для заполнения массивов использовался генератор случайных чисел. Результат работы программы приведён на рисунках (рис. 15-18). Исходный Файл: Programm\_.h программ так же доступен на GitHub по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm5_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm5_2.h>
3. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm5_3.h>
4. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm5_4.h>

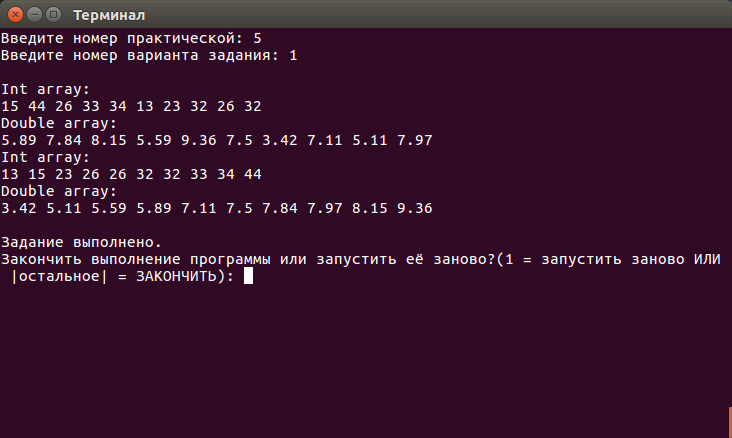


Рис. 15 Результат работы программы №1

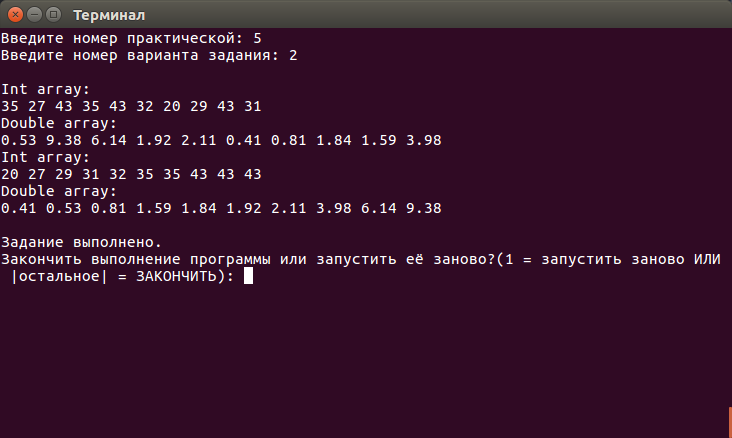


Рис. 16 Результат работы программы №2

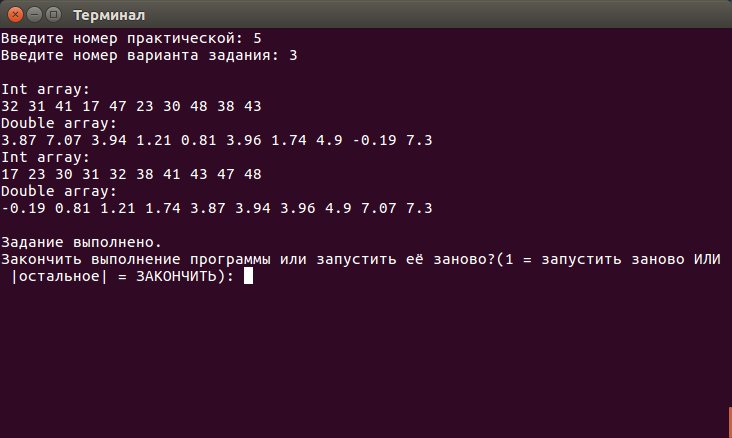


Рис. 17 Результат работы программы №3

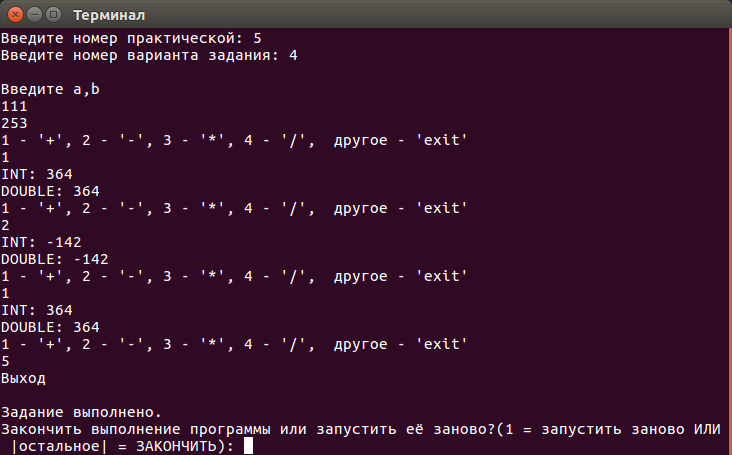


Рис. 18 Результат работы программы №4

## Практическая работа №6

**Решение задач на наследование на языке С++**

**Цель практической работы**

Целью данной практической работы является приобретение практических навыков использования принципа ООП – наследования для разработки программ на языке программирования C++.

**Задачи**

1. Написать иерархию классов, описывающих имущество налогоплательщиков. Она должна состоять из абстрактного базового класса Property и производных от него классов Appartment, Car и CountryHouse. Базовый класс должен иметь поле worth (стоимость), конструктор с одним параметром, заполняющий это поле, и чисто виртуальный метод рассчета налога, переопределенный в каждом из производных классов. Налог на квартиру вычисляется как 1/1000 ее стоимости, на машину – 1/200, на дачу – 1/500. Также, каждый производный класс должен иметь конструктор с одним параметром, передающий свой параметр конструктору базового класса. В функции main завести массив из 7 указателей на Property и заполнить его указателями на динамические объекты производных классов (первые 3 – Appartment, следующие 2 – Car и последние 2 – CountryHouse). Вывести на экран величину налога для всех 7 объектов. Не забудь также уничтожить динамические объекты перед завершением программы.
2. Написать набор классов, представляющий выражения. В этом наборе должен быть один абстрактный базовый тип, а также набор производных от него типов по видам выражений (константа, переменная, сумма, разность, произведение, частное, sin, cos, exp, ln). У каждого из классов должны быть следующие виртуальные функции: напечатать выражение (без параметров), вычислить выражение (параметр – значение переменной, результат – значение выражения), вернуть производную выражения (без параметров), создать копию выражения (тоже без параметров).

**Ход работы**

1. Программа №1

ФАЙЛ: Programm6\_1.h

#ifndef Programm6\_1

#define Programm6\_1

class Property//класс собственность

{

protected:

double worth;//стоимость

public:

Property(double cost): worth(cost) {};

virtual ~Property() {};

virtual double tax() = 0;//виртуальная функция налогов

};

class Appartment : public Property// класс квартира

{

public:

Appartment(double cost) : Property(cost) {};

double tax()

{

return worth/1000;

}

};

class Car : public Property// класс машина

{

public:

Car(double cost) : Property(cost) {};

double tax()

{

return worth / 200;

}

};

class CountryHouse : public Property// класс дача

{

public:

CountryHouse(double cost) : Property(cost) {};

double tax()

{

return worth / 500;

}

};

void Programma6\_1(){

Property \*properties[7];//массив указателей на Property

double cost;

for (int i = 0; i < 7; i++)

{

cout << "Введите стоимость" << endl;

cin >> cost;

if (i < 3)// первые 2 - класс "квартира"

properties[i] = new Appartment(cost);

else if (i == 3 || i == 4)// следующие 2 - класс машина

properties[i] = new Car(cost);

else//последние 2 - класс "дача"

properties[i] = new CountryHouse(cost);

}

cout << "Taxes:" << endl;

for (int i = 0; i < 7; i++) cout << (properties[i]->tax()) << endl;//вывод налогов

for (int i = 0; i < 7; i++) delete properties[i];//удаление массива

}

#endif

1. Программа №2

ФАЙЛ: Programm6\_2.h

#ifndef Programm6\_2

#define Programm6\_2

class Expression// абстрактный класс "выражение"

{

protected:

double a;

public:

Expression(double b = 0) : a(b) {};//конструктор

virtual void print() = 0;//виртуальная функция для печати выражения

virtual double result(double b) = 0;// виртуальная функция результата

virtual double derivative() = 0;// виртуальная функция производной

virtual void copy(double b) = 0;// виртуальная функция для копирования выражения

virtual ~Expression() {};

};

class Cons : Expression// класс "константа"

{

public:

Cons(double b = 0) : Expression(b) {};

void print()

{

cout << "Const: ";

}

double result(double b)

{

return a;

}

double derivative()

{

return 0;

}

void copy(double b)

{

}

};

class Var : Expression// класс "переменная"

{

public:

Var(double b = 0) : Expression(b) {};

void print()

{

cout << "Variable: ";

}

double result(double b)

{

return b;

}

double derivative()

{

return 1;

}

void copy(double b)

{

a = b;

}

};

class Sum : Expression// класс "сумма"

{

public:

Sum(double b = 0) : Expression(b) {};

void print()

{

cout << "Sum: ";

}

double result(double b)

{

return a+b;

}

double derivative()

{

return 1;

}

void copy(double b)

{

a = a + b;

}

};

class Dif : Expression// класс "разность"

{

public:

Dif(double b = 0) : Expression(b) {};

void print()

{

cout << "Difference: ";

}

double result(double b)

{

return a-b;

}

double derivative()

{

return 1;

}

void copy(double b)

{

a = a - b;

}

};

class Mult : Expression// класс "произведение"

{

public:

Mult(double b = 0) : Expression(b) {};

void print()

{

cout << "Multiply: ";

}

double result(double b)

{

return a \* b;

}

double derivative()

{

return 1;

}

void copy(double b)

{

a = a \* b;

}

};

class Pri : Expression// класс "частное"

{

public:

Pri(double b = 0) : Expression(b) {};

void print()

{

cout << "Quotient: ";

}

double result(double b)

{

if (b!=0)

return a / b;

else return 0;

}

double derivative()

{

return 1;

}

void copy(double b)

{

if (b != 0)

a = a / b;

else a = 0;

}

};

class Sin : Expression// класс "синус"

{

public:

Sin(double b = 0) : Expression(b) {};

void print()

{

cout << "Sin: ";

}

double result(double b)

{

return sin(a);

}

double derivative()

{

return cos(a);

}

void copy(double b)

{

a = b;

}

};

class Cos : Expression// класс "косинус"

{

public:

Cos(double b = 0) : Expression(b) {};

void print()

{

cout << "Cos: ";

}

double result(double b)

{

return cos(b);

}

double derivative()

{

return -sin(a);

}

void copy(double b)

{

a = b;

}

};

class Exp : Expression// класс "экспонента"

{

public:

Exp(double b = 0) : Expression(b) {};

void print()

{

cout << "Exp: ";

}

double result(double b)

{

return exp(b);

}

double derivative()

{

return exp(a);

}

void copy(double b)

{

a = b;

}

};

class Ln : Expression// класс "логарифм"

{

public:

Ln(double b = 0) : Expression(b) {};

void print()

{

cout << "Log: ";

}

double result(double b)

{

return log(b);

}

double derivative()

{

return 1/a;

}

void copy(double b)

{

a = b;

}

};

void Programma6\_2(){

double a, b;

cout << "Введите a, b" << endl;

cin >> a >> b;

Cons cnst = a;

cout << "Резульат----------l----------Производная" << endl;

cnst.print();

cout << cnst.result(b) << " ---------l--------- " << cnst.derivative() << endl;

Var v = a;

v.print();

cout << v.result(b) << " ---------l--------- " << v.derivative() << endl;

Sum s = a;

s.print();

cout << s.result(b) << " ---------l--------- " << s.derivative() << endl;

Dif d = a;

d.print();

cout << d.result(b) << " ---------l--------- " << d.derivative() << endl;

Mult m = a;

m.print();

cout << m.result(b) << " ---------l--------- " << m.derivative() << endl;

Pri p = a;

p.print();

cout << p.result(b) << " ---------l--------- " << p.derivative() << endl;

Sin sn = a;

sn.print();

cout << sn.result(b) << " ---------l--------- " << sn.derivative() << endl;

Cos cs = a;

cs.print();

cout << cs.result(b) << " ---------l--------- " << cs.derivative() << endl;

Exp e = a;

e.print();

cout << e.result(b) << " ---------l--------- " << e.derivative() << endl;

Ln l = a;

l.print();

cout << l.result(b) << " ---------l--------- " << l.derivative() << endl;

}

#endif

**Вывод**

Все поставленные задачи были выполнены. В ходе выполнения работы был изучен механизм наследования классов в C++. Также для выполнения работы были использованы абстрактные классы и чисто виртуальные методы базовых абстрактных классов. Результаты работы программ представлены на рисунках (рис 19, 20). Исходный Файл: Programm\_.h программ также доступен на GitHub по ссылкам:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm6_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm6_2.h>

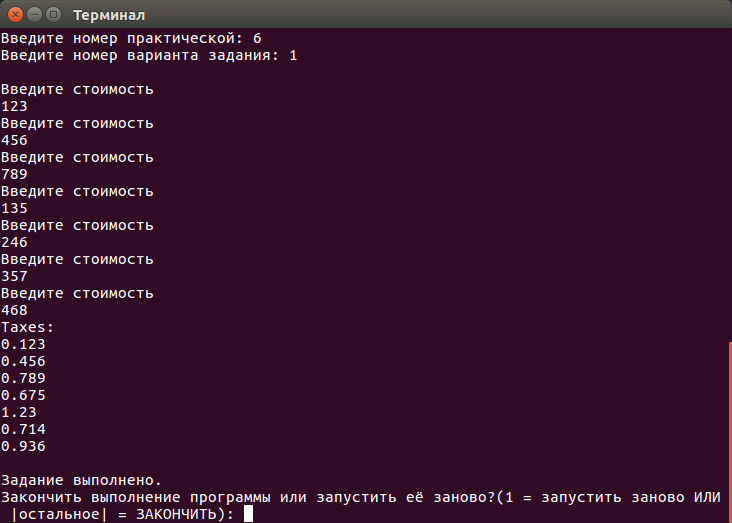


Рис. 19 Результат работы программы №1

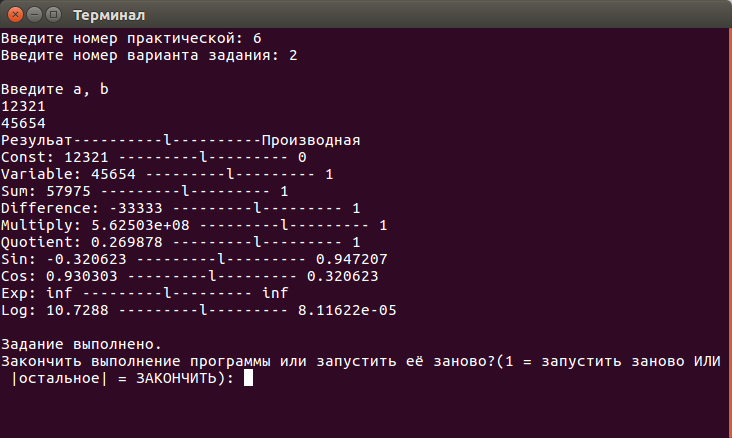


Рис. 20 Результат работы программы №2

## Практическая работа №7

**Абстрактные классы. Множественное наследование**

**Цель практической работы**

Целью данной практической работы является знакомство с абстрактными классами и механизмом виртуальных функций, а так же множественным наследованием на языке программирования C++.

**Задачи**

1. Реализовать абстрактный класс «Животное» и путём наследования от него получить классы «Кошка», «Собака», «Попугай».
2. Реализовать абстрактный класс «Фигура» и путём наследования от него получить абстрактный класс «Четырёхугольник», и затем путём наследования получить классы «Ромб», «Прямоугольник».
3. Реализовать абстрактный класс «Транспортное средство» и путём наследования от него получить классы «Автомобиль», «Автобус», «Велосипед».
4. Реализовать абстрактные классы «Экран» и «Клавиатура», путём наследования от них получить классы «Ноутбук», «Телефон», «Стационарный компьютер».

**Ход работы**

1. Программа №1

ФАЙЛ: Programm7\_1.h

#ifndef Programm7\_1

#define Programm7\_1

class Animal { //класс Животное

protected:

int legs;

string name;

public:

Animal(string&, int);

void setName(string &name) { this->name = name; }

string &getName() { return name; }

void setLegs(int legs) { this->legs = legs; }

int getLegs() { return legs; }

virtual void say() = 0; // Чисто виртуальная функция, выводящая на экран речь животного

};

class Cat : public Animal { // Производный класс Кошка

public:

Cat(string &name, int legs) : Animal(name, legs) {}

void say() { cout << "meow" << endl; }

};

class Dog : public Animal { // Производный класс Собака

public:

Dog(string &name, int legs) : Animal(name, legs) {}

void say() { cout << "woof" << endl; }

};

class Parrot : public Animal { // Производный класс Попугай

public:

Parrot(string &name, int legs) : Animal(name, legs) {}

void say() { cout << "tweet" << endl; }

};

void Programma7\_1(){

string s1 = "Mur", s2 = "Doggy", s3 = "Vasili";

Cat c(s1, 4);

Dog d(s2, 4);

Parrot p(s3, 2);

cout << "Dog goes ";

d.say();

cout << "Cat goes ";

c.say();

cout << "Bird goes ";

p.say();

cout << "But what does the fox say?" << endl;

}

Animal::Animal(string &name, int legs) {

this->name = name;

this->legs = legs;

}

#endif

1. Программа №2

ФАЙЛ: Programm7\_2.h

#ifndef Programm7\_2

#define Programm7\_2

class Shape// абстрактный класс фигура

{

protected:

double s;

public:

virtual double square() = 0;

};

class Quadrilateral : Shape // абстрактный класс четырехугольник

{

protected:

double a, b;

public:

Quadrilateral(double a1, double a2)

{

a = a1;

b = a2;

};

};

class Rhombus : Quadrilateral//подкласс ромб

{

private:

double sn;

public:

Rhombus(double a1, double a2, double s) : Quadrilateral(a1, a2) { sn = s; };

double square()

{

return a\*b\*sin(sn\*3.14 / 180);

}

};

class Rectangle : Quadrilateral//подкласс прямоугольник

{

public:

Rectangle(double a1, double a2) : Quadrilateral(a1, a2) {};

double square()

{

return a\*b;

}

};

void Programma7\_2(){

double a, b;

cout << "Введите сторону а затем угол ромба" << endl;

cin >> a >> b;

Rhombus rh(a, a, b);

cout << "Ввдите стороны прямоугольника" << endl;

cin >> a >> b;

Rectangle re(a, b);

cout << "Площадь ромба: " << rh.square() << endl;

cout << "Площадь рямоугольника: " << re.square() << endl;

}

#endif

1. Программа №3

ФАЙЛ: Programm7\_3.h

#ifndef programm7\_3

#define Programm7\_3

class Transport { //класс транспорт

protected:

int wheels;

public:

Transport(int);

void setwheels(int wheels) { this->wheels = wheels; }

int getwheels() { return wheels; }

virtual void beep() = 0;

};

class Carr : public Transport {

public:

Carr(int wheels) : Transport(wheels) {}

void beep() { cout << this->wheels; }

};

class Bus : public Transport {

public:

Bus(int wheels) : Transport(wheels) {}

void beep() { cout << this->wheels; }

};

class Bicycle : public Transport {

public:

Bicycle(int wheels) : Transport(wheels) {}

void beep() { cout << this->wheels; }

};

void Programma7\_3(){

Carr ca(4);

Bus bu(6);

Bicycle bi(2);

cout << "Bus has ";

bu.beep();

cout << " wheels" << endl;

cout << "Car has ";

ca.beep();

cout << " wheels" << endl;

cout << "Bicycle has ";

bi.beep();

cout << " wheels" << endl;

}

Transport::Transport(int wheels) {

this->wheels = wheels;

}

#endif

1. Программа №4

ФАЙЛ: Programm7\_4.h

#ifndef Programm7\_4

#define Programm7\_4

class Screen

{

protected:

string Resolution;

public:

Screen(string Resolution) { this->Resolution = Resolution; }

virtual string printResolution() = 0;

};

class Keyboard

{

protected:

string KeyboardLayout;

public:

Keyboard(string KeyboardLayout) { this->KeyboardLayout = KeyboardLayout; }

virtual string printKeyboardLayout() = 0;

};

class Notebook : public Screen, Keyboard

{

protected:

string name = "Notebook";

public:

Notebook(string Resolution, string KeyboardLayout) : Keyboard(KeyboardLayout), Screen(Resolution) {}

string printKeyboardLayout()

{

return KeyboardLayout;

};

string printResolution()

{

return Resolution;

};

};

class Phone : public Screen, Keyboard

{

protected:

string name = "Mobile Phone";

public:

Phone(string Resolution, string KeyboardLayout) : Keyboard(KeyboardLayout), Screen(Resolution) {}

string printKeyboardLayout()

{

return KeyboardLayout;

};

string printResolution()

{

return Resolution;

};

};

class PC : public Screen, Keyboard

{

protected:

string name = "PC";

public:

PC(string Resolution, string KeyboardLayout) : Keyboard(KeyboardLayout), Screen(Resolution) {}

string printKeyboardLayout()

{

return KeyboardLayout;

};

string printResolution()

{

return Resolution;

};

};

void Programma7\_4(){

string Resolution;

string KeyboardLayout;

cout << "ПК: " << endl;

cout << "Введите разрешение экрана (например: 1920х1080): ";

cin >> Resolution;

cout << "Введите раскладку клавиатуры (например QWERTY): ";

cin >> KeyboardLayout;

PC p(Resolution, KeyboardLayout);

cout << "Телефон: " << endl;

cout << "Введите разрешение экрана (например: 1920х1080): ";

cin >> Resolution;

cout << "Введите раскладку клавиатуры (например QWERTY): ";

cin >> KeyboardLayout;

Phone phon(Resolution, KeyboardLayout);

cout << "Ноутбук: " << endl;

cout << "Введите разрешение экрана (например: 1920х1080): ";

cin >> Resolution;

cout << "Введите раскладку клавиатуры (например QWERTY): ";

cin >> KeyboardLayout;

Notebook r(Resolution, KeyboardLayout);

cout << "ПК: " << endl << "разрешение экрана: " << p.printResolution() << endl << "Раскладка клавиатуры: " << p.printKeyboardLayout() << endl;

cout << "Телефон: " << endl << "разрешение экрана: " << phon.printResolution() << endl << "Раскладка клавиатуры: " << phon.printKeyboardLayout() << endl;

cout << "Ноутбук: " << endl << "разрешение экрана: " << r.printResolution() << endl << "Раскладка клавиатуры: " << r.printKeyboardLayout() << endl;

}

#endif

**Вывод**

Все задачи были выполнены. В ходе выполнения работы были более плотно изучены множественное наследование и абстрактные классы. Для достижения поставленной цели были использованы различные STL контейнеры и библиотека cmath. Результаты работы программ представлены на изображениях (рис. 21-24). Исходный Файл: Programm\_.hвыполненных програм также доступе но ссылка на GitHub:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm7_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm7_2.h>
3. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm7_3.h>
4. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm7_4.h>

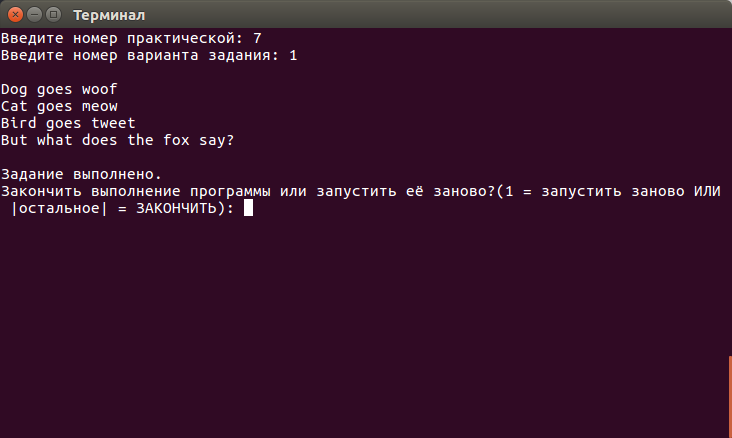


Рис. 21 Результат работы программы №1

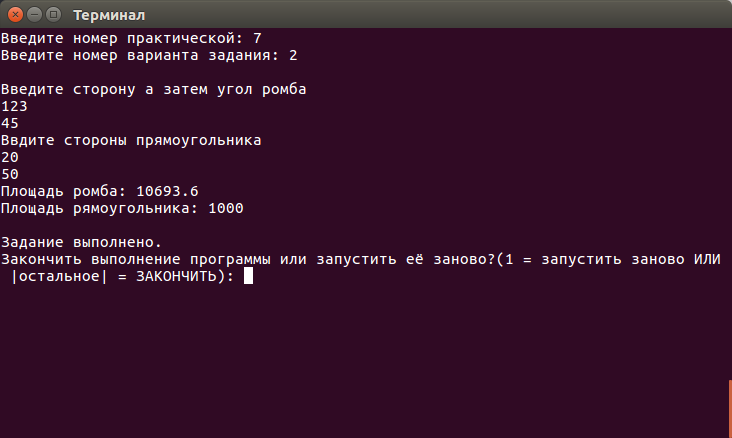


Рис. 22 Результат работы программы №2

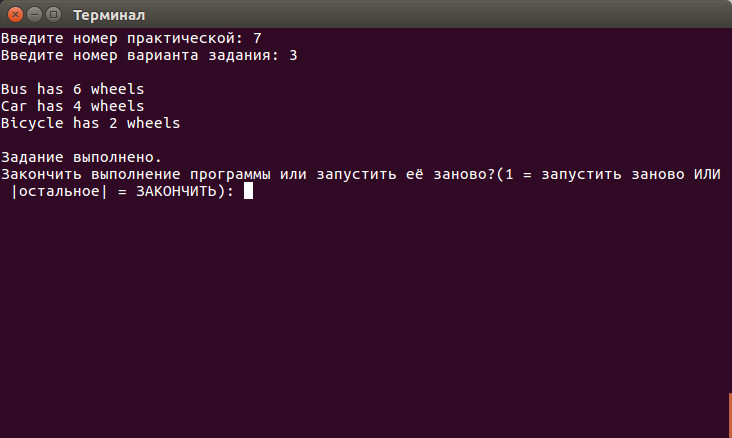


Рис. 23 Результат работы программы №3

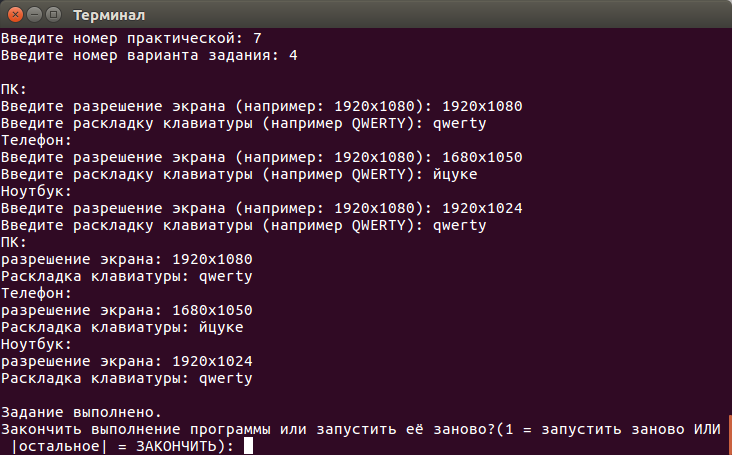


Рис. 24 Результат работы программы №4

## Практическая работа №8

**Работа с файлами в языке С++**

**Цель практической работы**

Целью данной практической работы является приобретение практических навыков по работе с файлами на языке программирования языке C++.

**Задачи**

1. Реализуйте программу, считывающую текст из файла и выводящую каждое слово с новой строки.
2. Реализуйте программу, считывающую текст с клавиатуры и записывающую его в файл.

**Ход работы**

1. Программа №1

ФАЙЛ: Programm8\_1.h

#include <fstream>

#ifndef Programm8\_1

#define Programm8\_1

void Programma8\_1(){

ifstream file("a.txt");

while (!file.eof()) {

string s;

file >> s;

cout << s << endl;

}

file.close();

}

#endif

1. Программа №1

ФАЙЛ: Programm8\_2.h

#ifndef Programm8\_2

#define Programm8\_2

void Programma8\_2(){

ofstream file("a.txt");

char c = 0;

cin >> c;

while (c != '\n') {

file << c;

cin.get(c);

}

file.close();

}

#endif

**Вывод**

Все задачи были выполнены успешно. В ходе работы была исследована работа с файлами. Для достижения результата использовалась библиотек fstream, позволяющая производить работу с файлами. Результаты работ программ приведены на рисунках (рис. 25, 26). Также исходный Файл: Programm\_.h програм доступенн по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm8_1.h>
2. <https://github.com/KorDima01/OOP/blob/master/Practicks/cpp/h/Programm8_2.h>

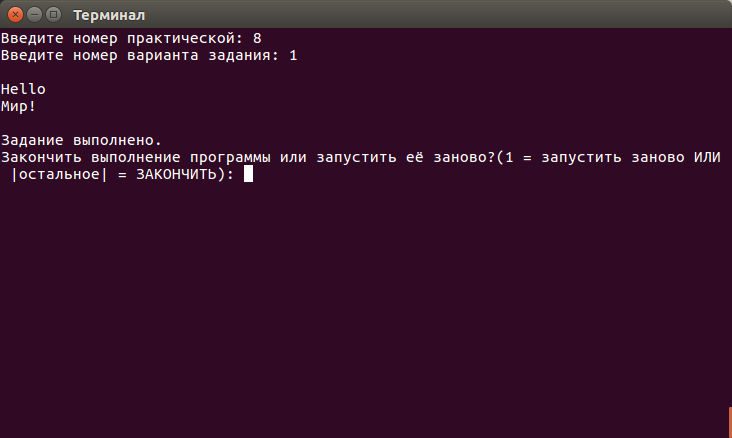
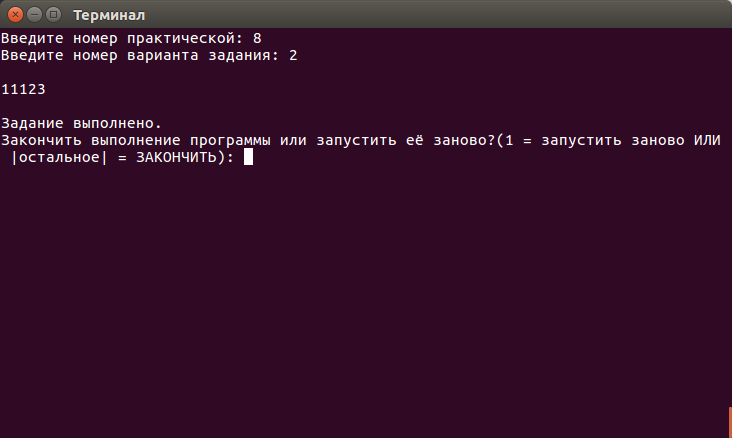


Рис. 25 Результат работы программы №1



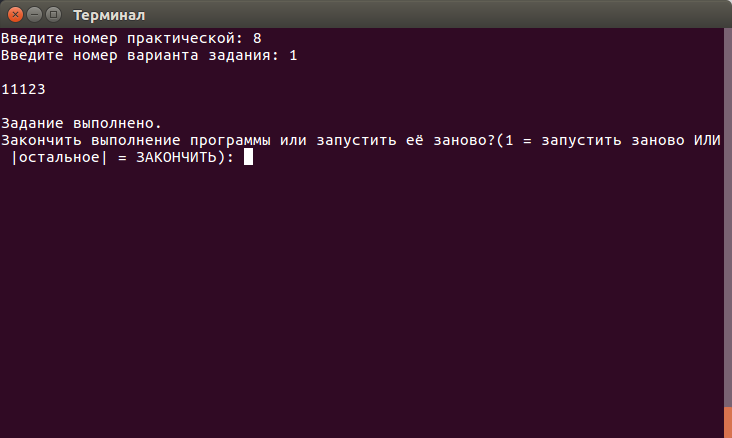


Рис. 26 Результат работы программы №2

## Лабораторная работа №1

**Создание многофайловых проектов**

**Цель практической работы**

Целью данной лабораторной работы освоить на практике создание многофайловых проектов а языке Си/Си++, познакомиться с директивами условной компиляции.

**Задачи**

1. Написать программу – калькулятор комплексных чисел.
2. На основе задания 1 написать программу, считывающую комплексные числа из файла complex.txt и находящуюю в них число с наибольшим модулем.

**Ход работы**

Файл заголовков Complex.h:

class Complex{

double Re\_,Im\_;

public:

Complex(double Re = 0,double Im = 0);

double printRe();

double printIm();

double getRe();

double getIm();

double module();

Complex operator+(Complex b);

Complex operator-(Complex b);

Complex operator\*(Complex b);

Complex operator/(Complex b);

Complex pow(int p);

};

Файл реализации класса Complex.cpp:

#include "Complex.h"

#include <cmath>

Complex::Complex(double Re,double Im){

Re\_ = Re;

Im\_ = Im;

}

double Complex::printRe(){

return Re\_;

}

double Complex::printIm(){

return Im\_;

}

double Complex::getRe(){

return this->Re\_;

}

double Complex::getIm(){

return this->Im\_;

}

double Complex::module(){

return sqrt(this->Re\_\*this->Re\_ + this->Im\_ \* this->Im\_);

}

Complex Complex::operator+(Complex b){

Complex res;

res.Re\_ = this->Re\_ + b.Re\_;

res.Im\_ = this->Im\_ + b.Im\_;

return res;

}

Complex Complex::operator-(Complex b){

Complex res;

res.Re\_ = this->Re\_ - b.Re\_;

res.Im\_ = this->Im\_ - b.Im\_;

return res;

}

Complex Complex::operator\*(Complex b){

Complex res;

res.Re\_ = this->Re\_ \* b.Re\_ - this->Im\_ \* b.Im\_;

res.Im\_ = this->Re\_ \* b.Im\_ + this->Im\_ \* b.Re\_;

return res;

}

Complex Complex::operator/(Complex b){

Complex res;

res.Re\_ = (this->Re\_ \* b.Re\_ + this->Im\_ \* b.Im\_)/(b.Re\_ \* b.Re\_ + b.Im\_ \* b.Im\_);

res.Im\_ = (b.Re\_ \* this->Im\_ - this->Re\_ \* b.Im\_)/(b.Re\_ \* b.Re\_ + b.Im\_ \* b.Im\_);

return res;

}

Complex Complex::pow(int p){

Complex res;

res = \*this;

while (p > 1) {

res = res \* \*this;

p--;

}

return res;

}

1. Программа №1

#include <iostream>

#include "Complex.cpp"

#include <vector>

#include <fstream>

using namespace std;

int main() {

ifstream file("Test.txt");

int n,r,m;

vector<Complex> p;

file >> n;

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

file >> r >> m;

Complex el(r,m);

p.push\_back(el);

}

file.close();

Complex max;

double md = p[0].module();

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

if (p[i].module() > md){

md = p[i].module();

max = p[i];

}

cout << max.printRe() << ' ' << max.printIm() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

return 0;

}

1. Программа №2

#include <iostream>

#include <vector>

#include <ctime>

#include <algorithm>

using namespace std;

int sum (int a,int b){

return a + b;

}

int multiply(int a,int b){

return a\*b;

}

int divis(int a,int b){

return a/b;

}

int diff(int a,int b){

return a-b;

}

void reverseArray(vector<int> &arr){

int t;

for (int i = 0;i < 5;i++){

t = arr[i];

arr[i] = arr[9-i];

arr[9-i] = t;

}

}

void neubArray(vector<int> &arr){

int t;

for (int i = 0;i < 10;i++)

for (int j = i;j < 10;j++)

if (arr[j] < arr[i]){

t = arr[j];

arr[j] = arr[i];

arr[i] = t;

}

}

void ubArray(vector<int> &arr){

int t;

for (int i = 0;i < 10;i++)

for (int j = i;j < 10;j++)

if (arr[j] > arr[i]){

t = arr[j];

arr[j] = arr[i];

arr[i] = t;

}

}

void (\*arrayFunc(vector<int> &arr))(vector<int> &arr){

int sum = 0;

for (int i = 0;i < 10;i++)

sum += arr[i];

if (sum == arr[0])

return reverseArray;

else if (sum > arr[0])

return neubArray;

else

return ubArray;

}

int (\*func)(int a,int b);

int main(int argc, const char \* argv[]) {

int a,b,action;

cout << "Enter a: ";

cin >> a;

cout << "Enter b: ";

cin >> b;

cout << "Enter 1 - sum of a and b \n 2 - different of a and b \n 3 - multiply of a and b \n 4 - division of a and b" << endl;

cin >> action;

switch (action) {

case 1:

func = sum;

break;

case 2:

func = diff;

break;

case 3:

func = multiply;

break;

case 4:

func = divis;

break;

}

cout << "Result is " << func(a,b) << endl;

//Вариант 2

vector<int> arr;

srand(time(0));

cout << endl << "Created array" << endl;

for (int i = 0; i < 10;i++){

arr.push\_back(rand()%20);

cout << arr[i] << ' ';

}

cout << endl;

arrayFunc(arr)(arr);

cout << "Array after function" << endl;

for (int i = 0;i < 10;i++)

cout << arr[i] << ' ';

cout << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. Для достижения требуемого результата был создан класс, разбитый на файлы с реализацией и объявлением. Также были использованы некоторые дерективы препрцессора для защиты от многократного подключения файлов и перегружены некоторые операторы. Результаты работы программ приведены на рисунках (рис. 27, 28). Исходный Файл также доступен по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/1/1_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/1/1_2>

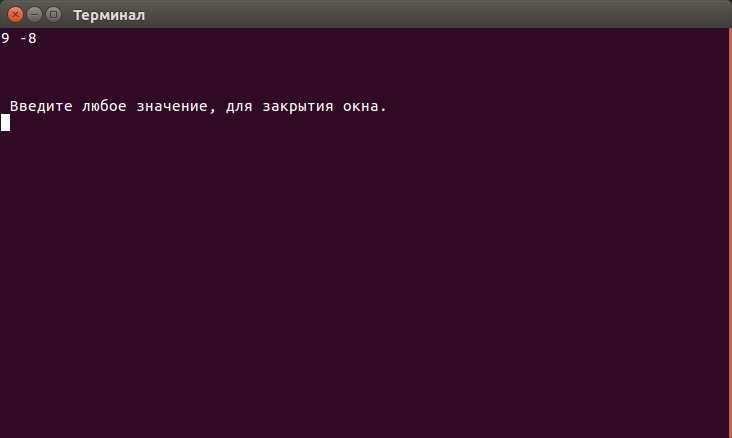


Рис. 27 Результат работы программы №1

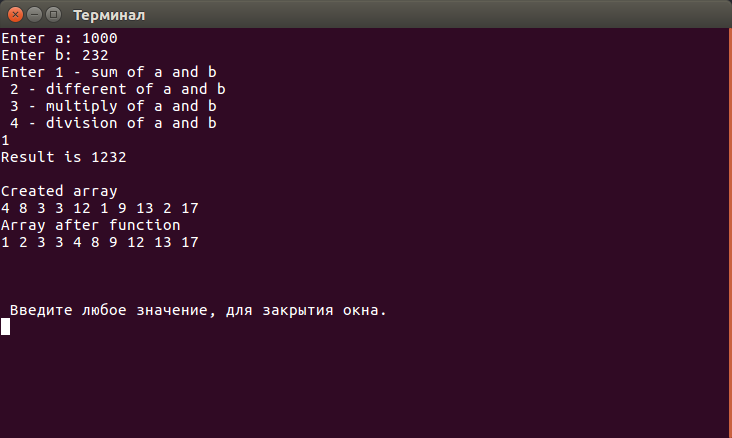


Рис. 28 Результат работы программы №2

## Лабораторная работа №2

**Указатели на функции**

**Цель практической работы**

Целью данной лабораторной работы освоить на практике вызов функции с использованием указателей.

**Задачи**

1. Напишите программу, которая вызывает различные виды функции в зависимости от заданного условия.
2. Напишите программу с использованием некой универсальной функции, которая в качестве возвращаемого значения возвращает указатель на функцию, зависящий от некоторого условия.v

**Ход работы**

1. Программа №1

#include <iostream>

using namespace std;

int sum(int a, int b) {

return a + b;

}

int diff(int a, int b) {

return a - b;

}

int prod(int a, int b) {

return a \* b;

}

int divis(int a, int b) {

return a / b;

}

int(\*res)(int a, int b);

int main()

{

char arr[4] = { '+', '-', '\*', '/' };

int a, b, ch;

cout << "Enter a and b:" << endl;

cin >> a >> b;

cout << "Enter:\nSum - 1\ndiff - 2\n prod - 3\ndiv - 4" << endl;

cin >> ch;

switch (ch)

{

case(1):

{

res = sum;

break;

}

case(2):

{

res = diff;

break;

}

case(3):

{

res = prod;

break;

}

case(4):

{

res = divis;

break;

}

default:

break;

}

cout << "a " << arr[ch-1] << " b = " << res(a, b) << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №2

#include <iostream>

#include <cstdlib>

#include <vector>

#include <ctime>

using namespace std;

void reverseArray(vector<int> &arr) {

int t;

for (int i = 0; i < 5; i++) {

t = arr[i];

arr[i] = arr[9 - i];

arr[9 - i] = t;

}

}

void neubArray(vector<int> &arr) {

int t;

for (int i = 0; i < 10; i++)

for (int j = i; j < 10; j++)

if (arr[j] < arr[i]) {

t = arr[j];

arr[j] = arr[i];

arr[i] = t;

}

}

void ubArray(vector<int> &arr) {

int t;

for (int i = 0; i < 10; i++)

for (int j = i; j < 10; j++)

if (arr[j] > arr[i]) {

t = arr[j];

arr[j] = arr[i];

arr[i] = t;

}

}

void(\*arrayFunc(vector<int> &arr))(vector<int> &arr) {

int sum = 0;

for (int i = 0; i < 10; i++)

sum += arr[i];

if (sum == arr[0])

return reverseArray;

else if (sum > arr[0])

return neubArray;

else

return ubArray;

}

int main()

{

vector<int> arr;

srand(time(NULL));

cout << "Created array" << endl;

for (int i = 0; i < 10; i++) {

arr.push\_back(rand() % 20);

cout << arr[i] << ' ';

}

cout << endl;

arrayFunc(arr)(arr);

cout << "Array after function" << endl;

for (int i = 0; i < 10; i++)

cout << arr[i] << ' ';

cout << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. Для достижения поставленных целей были использованы указатели на функции с различными возвращаемыми значениями. Результаты работы программ можно увидеть на рисунках (рис. 29, 30). Исходный Файл: Programm\_.h программ также доступен по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/2/2_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/2/2_2>

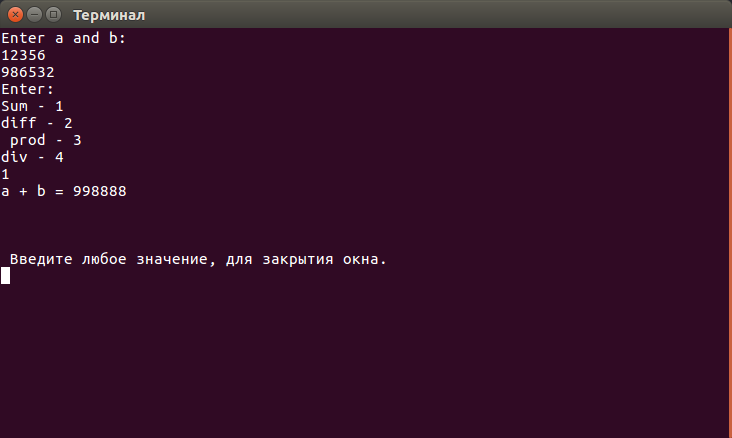


Рис. 29 Результат работы программы №1

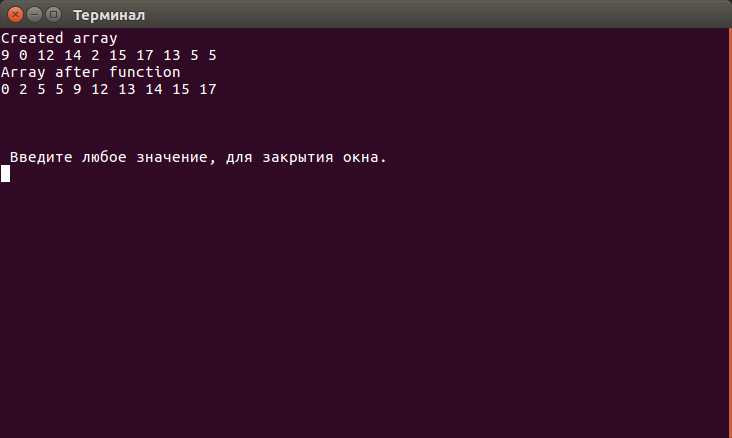


Рис. 30 Результат работы программы №2

## Лабораторная работа №3

**Классы**

**Цель практической работы**

Целью данной лабораторной работы является знакомство с классами в языке С++.

**Задачи**

1. Определить класс Child, который содержит такие поля (члены класса): закрытые — имя ребенка, фамилию и возраст, публичные — методы ввода данных и отображения их на экран. Объявить два объекта класса, внести данные и показать их.
2. Создать класс Tiles (кафель), который будет содержать поля с открытым доступом: brand, size\_h, size\_w, price и метод класса getData(). В главной функции объявить пару объектов класса и внести данные в поля. Затем отобразить их, вызвав метод getData().
3. Создать класс Complex, в котором реализовано комплексное число. В данном классе должны присутствовать методы, позволяющие рассчтать и вывеси модуль и аргументы данного числа.
4. Реализовать класс Vector, позволяющий хранить в себе математический вектор. В классе должно присутствовать метод позволяющей получить модуль вектора и методы, позволяющие складывать и вычитать разные векторы.

**Ход работы**

1. Программа №1

#include <iostream>

#include <cmath>

using namespace std;

class Complex

{

double re, im;

public:

Complex(double re = 0.0, double im = 0.0) {this->re = re, this->im = im;}

double getRe() {

return re;

}

double getIm() {

return im;

}

double module() {

return (sqrt(this->re \* this->re + this->im \* this->im));

}

};

int main()

{

Complex c(1.1, 2.0);

cout << "Re: " << c.getRe() << " Im: " << c.getIm() << endl;

cout << c.module() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программы №2

#include <iostream>

#include <string>

using namespace std;

class Tiles {

public:

string brand;

int size\_h, size\_w, price;

void getData() {

cout << "Brand: " << brand << endl;

cout << "Size\_h: " << size\_h << endl;

cout << "Size\_w: " << size\_w << endl;

cout << "Price: " << price << endl;

}

};

int main()

{

Tiles t1, t2;

t1.brand = "Ferm"; t1.size\_h = 15; t1.size\_w = 10; t1.price = 30;

t1.getData();

t2.brand = "Organic"; t2.size\_h = 5; t2.size\_w = 6; t2.price = 70;

t2.getData();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №3

#include <iostream>

#include <string>

using namespace std;

class Child {

string firstName;

string lastName;

int years;

public:

void setData() {

cout << "Enter firstName: ";

cin >> firstName;

cout << "Enter lastName: ";

cin >> lastName;

cout << "Enter years: ";

cin >> years;

}

void getData() {

cout << "firstName: " << firstName << endl;

cout << "lastName: " << lastName << endl;

cout << "years: " << years << endl;

}

};

int main(int argc, const char \* argv[]) {

Child ch1, ch2;

ch1.setData();

ch1.getData();

ch2.setData();

ch2.getData();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №4

#include <iostream>

#include <cmath>

using namespace std;

class Vector {

public:

int x, y;

Vector() {}

Vector(int x, int y) : x(x), y(y) {}

double module() { return sqrt(x\*x + y \* y); };

Vector sum(Vector b) {

Vector res;

res.x = this->x + b.x;

res.y = this->y + b.y;

return res;

}

Vector diff(Vector b) {

Vector res;

res.x = this->x - b.x;

res.y = this->y - b.y;

return res;

}

};

int main()

{

Vector a(1, 2), b(3, -4);

cout << "Sum: " << a.sum(b).x << ' ' << a.sum(b).y << endl;

cout << "module a: " << a.module() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были достигнуты. Для получения требуемого результата былииспользованы конструкции класса и перегрузка некоторых операторов. Результаты работы программ изображены на рисунках (рис. 31-34). Исходные Файл: Programm\_.hы программ также доступны по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/3/3_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/3/3_2>
3. <https://github.com/KorDima01/OOP/tree/master/Labs/3/3_3>
4. <https://github.com/KorDima01/OOP/tree/master/Labs/3/3_4>

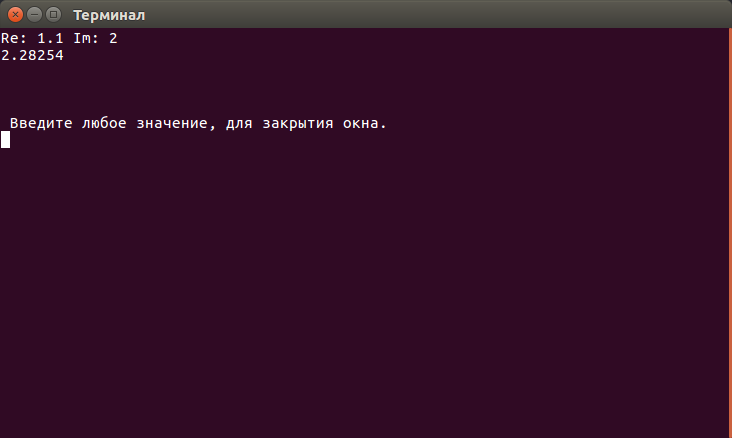


Рис. 31 Результат работы программы №1

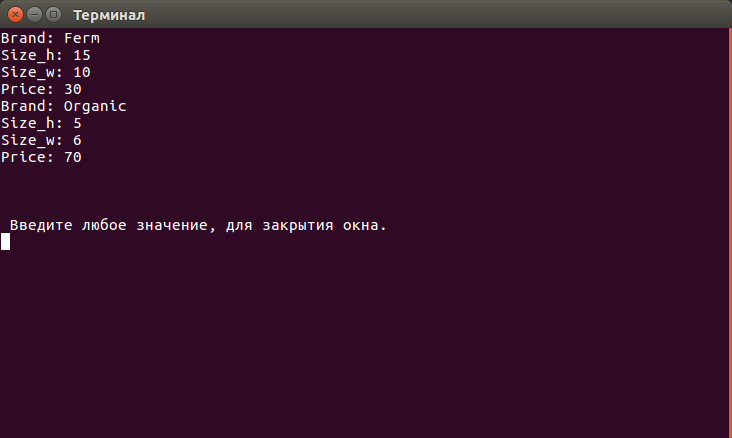


Рис. 32 Результат работы программы №2

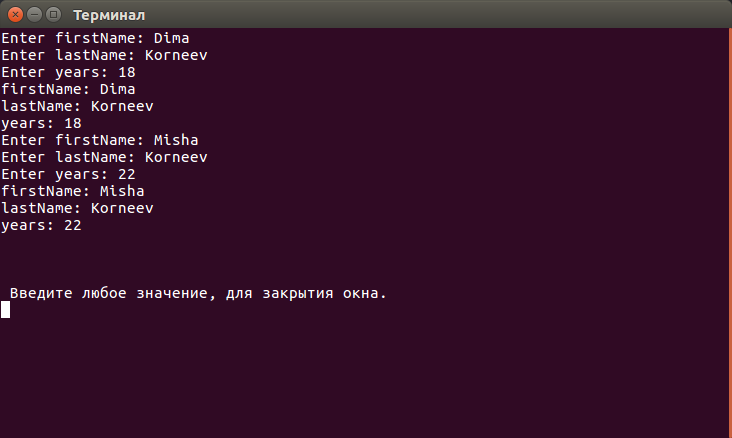


Рис. 33 Результат работы программы №3

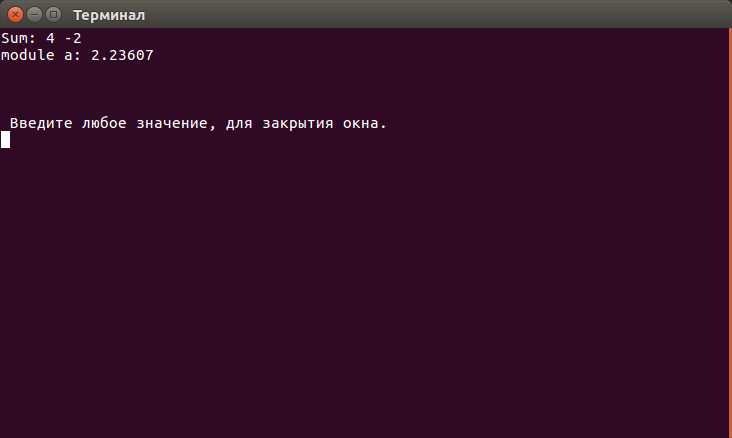


Рис. 34 Результат работы программы №4

## Лабораторная работа №4

**Конструкторы и деструкторы**

**Цель практической работы**

Целью данной лабораторной работы является знакомство с конструкторами и деструкторами в языке С++.

**Задачи**

Из прошлой лабораторной работы дополнить всем видами конструкторов и деструкторами классы:

1. Complex
2. Vector
3. Tiles
4. Child

**Ход работы**

1. Программы №1-4

#include <iostream>

#include <cmath>

#include <string>

using namespace std;

class Child {

string firstName;

string lastName;

int years;

public:

Child(string firstName, string lastName, int years) :

firstName(firstName), lastName(lastName), years(years) {}

~Child() { firstName = ""; lastName = ""; years = 0;

cout << "Object destroyed" << endl; }

void setData() {

cout << "Enter firstName: ";

cin >> firstName;

cout << "Enter lastName: ";

cin >> lastName;

cout << "Enter years: ";

cin >> years;

}

void getData() {

cout << "firstName: " << firstName << endl;

cout << "lastName: " << lastName << endl;

cout << "years: " << years << endl;

}

};

class Tiles {

public:

string brand;

int size\_h, size\_w, price;

Tiles(string br,

int s\_h, int s\_w, int pr) : size\_h(s\_h), size\_w(s\_w), brand(br),

price(pr) {}

~Tiles()

{

brand = "";

size\_h = 0;

size\_w = 0;

price = 0;

cout << "Object destroyed" << endl;

}

void getData() {

cout << "Brand: " << brand << endl;

cout << "Size\_h: " << size\_h << endl;

cout << "Size\_w: " << size\_w << endl;

cout << "Price: " << price << endl;

}

};

class Complex

{

double re, im;

public:

Complex(double re = 0.0, double im = 0.0) { this->re = re, this->im = im; }

double getRe() {

return re;

}

double getIm() {

return im;

}

double module() {

return (sqrt(this->re \* this->re + this->im \* this->im));

}

~Complex()

{

re = 0;

im = 0;

cout << "Object destroyed" << endl;

}

};

class Vector {

public:

int x, y;

Vector() {}

Vector(int x, int y) : x(x), y(y) {}

double module() { return sqrt(x\*x + y \* y); };

Vector sum(Vector b) {

Vector res;

res.x = this->x + b.x;

res.y = this->y + b.y;

return res;

}

Vector diff(Vector b) {

Vector res;

res.x = this->x - b.x;

res.y = this->y - b.y;

return res;

}

~Vector()

{

x = 0;

y = 0;

cout << "Object destroyed" << endl;

}

};

int main()

{

// 1

Child ch("123", "456", 1);

ch.getData();

// 2

Tiles t("789", 10, 11, 12);

t.getData();

// 3

Complex c(1, 2);

cout << c.module() << endl;

// 4

Vector v(3, 4);

cout << v.module() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Были выполнены все задачи. В процессе работы были более подробно изучены разные виды конструкторов и деструкторы. Для выполнения работы были использованы простые конструкторы, конструкторы копирования, а также перегруженный оператор присваивания. Результаты работы программ можно увидеть на рисунках (рис. 35-38). Исходный Файл: Programm\_.h программ также доступен по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/4>

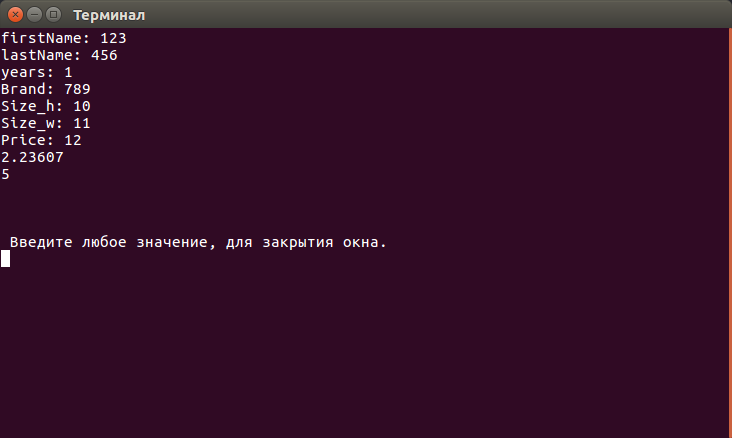


Рис. 35 Результат работы программм

## Лабораторная работа №5

**Перегрузка операторов в языке программирования C++**

**Цель практической работы**

Целью данной лабораторной работы является изучениеперегрузки операторов в языке С++ и использование перегруженныхоператоров на практике.

**Задачи**

1. Для класса Complex перегрузить операторы присваивания, инкремента, декремента, сравнения, ввода и вывода.
2. Для класса Vector перегрузить операторы присваивания, сравнения, ввода и вывода.

**Ход работы**

1. Программа №1

#include <iostream>

using namespace std;

class Complex

{

int re, im;

public:

Complex(double re = 0.0, double im = 0.0) {

this->re = re, this->im = im;

}

void operator= (Complex a) {

re = a.re;

im = a.im;

}

bool operator<(Complex a) {

return re \* re + im \* im < a.re\*a.re + a.im\*a.im;

}

bool operator>(Complex a) {

return re \* re + im \* im > a.re\*a.re + a.im\*a.im;

}

friend const Complex operator++ (Complex& c, int) {

Complex res(c);

c.re++;

c.im++;

return res;

}

friend const Complex operator-- (Complex& c, int) {

Complex res(c);

c.re--;

c.im--;

return res;

}

friend istream& operator>>(istream &in, Complex &a) {

in >> a.re;

in >> a.im;

return in;

}

friend ostream& operator<<(ostream &out, Complex &a) {

out << a.re << ' ' << a.im << endl;

return out;

}

};

int main()

{

Complex a, b(1, 2), c(3, 4);

a = b;

cout << (a > c) ? 1 : 0;

cout << endl;

cout << a << endl;

a++;

cout << a << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №2

#include <iostream>

using namespace std;

class Vector

{

int x, y;

public:

Vector(int x = 0, int y = 00) {

this->x = x, this->y = y;

}

void operator=(Vector a) {

x = a.x;

y = a.y;

}

bool operator<(Vector a) {

return x \* x + y \* y < a.x \* a.x + a.y \* a.y;

}

bool operator>(Vector a) {

return x \* x + y \* y > a.x \* a.x + a.y \* a.y;

}

friend istream& operator>>(istream &in, Vector &a) {

in >> a.x;

in >> a.y;

return in;

}

friend ostream& operator<<(ostream &out, Vector &a) {

out << a.x << ' ' << a.y << endl;

return out;

}

};

int main()

{

Vector a, b(1, 2), c(3, 4);

a = b;

cout << (a > c) ? 1 : 0;

cout << endl;

cout << a << endl;

cout << a << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. В процессе выполнения работы был более плотно изучен механизм перегрузки. Для выполнения поставленных задач была использована перегрузка некоторых бинарных и унарных операторов. Результаты работы программ можно увидеть на рисунках (рис. 39, 40). Исходный Файл: Programm\_.h программ также доступен по ссылкам на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/5/5_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/5/5_2>

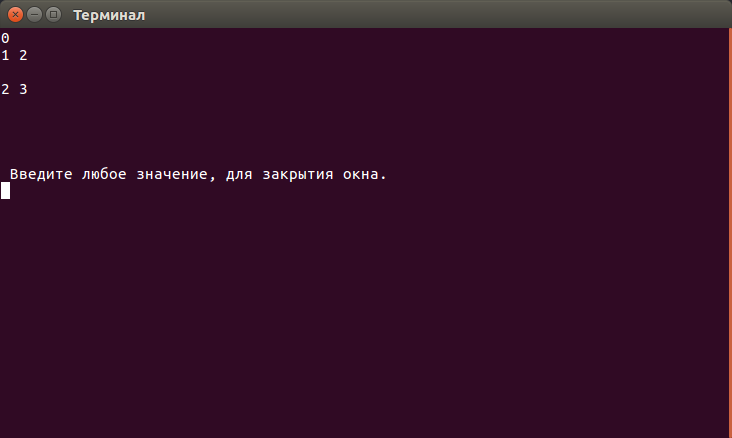


Рис. 36 Результат работы программы №1

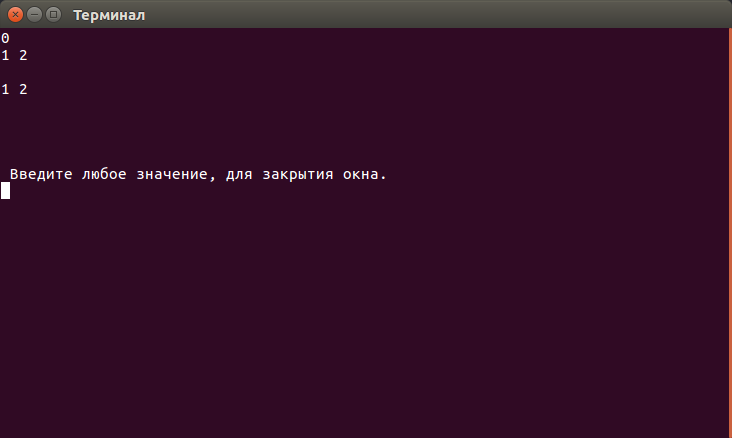


Рис. 37 Результат работы программы №2

## Лабораторная работа №6

**Наследование**

**Цель практической работы**

Целью данной лабораторной работы является изучение наследованная классов в языке С++.

**Задачи**

1. Создать класс «Староста», производный от класса «Студент». Новый класс должен содержать несколько дополнительных методов и полей.
2. Создать класс Alive и расширить его до Bird, Fish, Animal.
3. Создать класс Animal, и расширить его до Dog, Cat.

**Ход работы**

1. Программа №1

#include <iostream>

#include <string>

using namespace std;

class Student

{

string name, lastname;

public:

void setName(string name, string lastname) {

this->name = name;

this->lastname = lastname;

}

void getName() {

cout << "Name: " << name << "\nLastname: " << lastname << endl;

}

};

class Headman :public Student

{

string email, group;

public:

Headman(string email,string group) : email(email), group(group){}

void getIdOfGroup() {

cout <<"id: " << group << endl;

}

};

int main()

{

Headman h("@111", "INBO-04-18");

h.setName("Dmitriy", "Korneev");

cout << "Headman \n"; // староста

h.getName();

h.getIdOfGroup();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №2

#include <iostream>

#include <string>

using namespace std;

class Alive {

string name;

int years;

public:

Alive(string nm = "", int y = 0) :name(nm), years(y) {}

void setData(string name, int years) {

this->name = name;

this->years = years;

}

string getName() { return name; }

int getYears() { return years; }

};

class Bird :public Alive {

int countWings;

public:

Bird(int wg) :countWings(wg) {}

int getCountWings() { return countWings; }

};

class Fish :public Alive {

string type;

public:

Fish(string tp) :type(tp) {}

string getType() { return type; }

};

class Animal :public Alive {

string type\_animal;

public:

Animal(string tpAn) :type\_animal(tpAn) {}

string getType() { return type\_animal; }

};

int main()

{

Animal a("123");

Fish b("MoonFish");

Bird c(10);

a.setData("111", 2);

b.setData("222", 3);

c.setData("333", 2);

cout << "Animal\nType animal: " << a.getType() <<

"\nName of animal: " << a.getName() <<

"\nYears: " << a.getYears() << endl;

cout << "Fish\nType animal: " << b.getType() <<

"\nName of animal: " << b.getName() <<

"\nYears: " << b.getYears() << endl;

cout << "Bird\nCount wings: " << c.getCountWings() <<

"\nName of animal: " << c.getName() <<

"\nYears: " << c.getYears() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №3

#include <iostream>

#include <string>

#include <vector>

using namespace std;

class Animal3{

string name;

int years;

public:

Animal3(string nm = "Tom",int ys = 0):name(nm),years(ys){}

};

class Dog:Animal3{

string breed;

public:

Dog(string brd):breed(brd){}

};

class Cat:Animal3{

int childs;

public:

Cat(int chld):childs(chld){}

};

int main() {

Dog myDog("Buldog");

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. В процессе выполнения работы было изучено и использовано наследование. Результат работы программ изображены на рисунках (рис. 41-42). Исходный Файл: Programm\_.h программ также можно просмотреть на GitHub:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/6/6_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/6/6_2>
3. <https://github.com/KorDima01/OOP/tree/master/Labs/6/6_3>

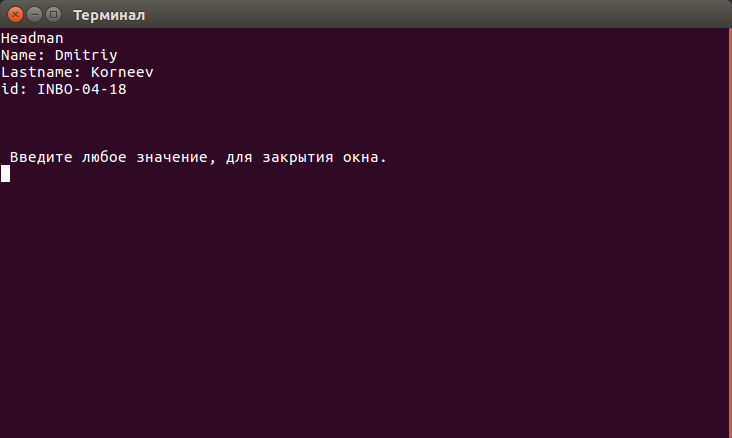


Рис. 41 Результат работы программы №1

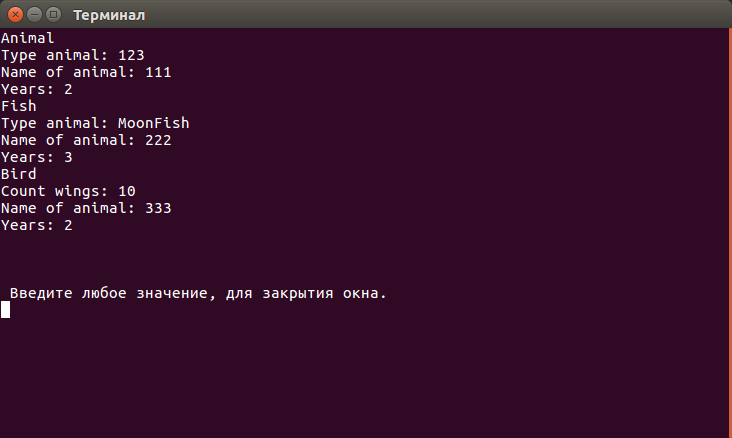


Рис. 42 Результат работы программы №2

## Лабораторная работа №7

**Создание абстрактных классов**

**Цель практической работы**

Целью данной лабораторной работы является изучение и создание абстрактных классов в языке С++.

**Задачи**

Реализовать систему из классов, изображённую на UML диаграмме.

**Ход работы**

#include <iostream>

#include <string>

using namespace std;

class Human {

string surname, name, midname;

int age;

public:

Human() {

surname = "123";

name = "456";

age = 18;

}

Human(string n, string sn, string mn, int ag) :name(n), surname(sn), midname(mn), age(ag) {}

~Human() {

midname = "";

surname = "";

name = "";

age = 0;

}

virtual void print() = 0;

};

class Student :Human {

bool on\_lesson;

public:

Student() :Human(), on\_lesson(0) {}

Student(string n, string sn, string mn, int ag, bool ol) :Human(n, sn, mn, ag), on\_lesson(ol) {}

void print() {

if (on\_lesson)

cout << "On lesson? True" << endl;

else

cout << "On lesson? False" << endl;

}

~Student() {

on\_lesson = false;

}

};

class Boss :Human {

int number\_of\_workers;

public:

Boss() :Human(), number\_of\_workers(0) {}

Boss(string n, string sn, string mn, int ag, int now) :Human(n, sn, mn, ag), number\_of\_workers(now) {}

void print() {

cout << "Number of workers: " << number\_of\_workers << endl;

}

~Boss() {

number\_of\_workers = 0;

}

};

int main()

{

Student s("123", "456", "789", 18, 1);

s.print();

Boss b("987", "654", "321", 54, 143);

b.print();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. При выплнении работы были изучены основные элементы UML диаграмм. Для достижения поставленной цели были использованы механизмы наследования, переопределения функций и абстрактные классы. Результат работы программы изображён на рисунке (рис. 44). Файл: Programm\_.h программы также доступен на GitHub по ссылке:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/7>

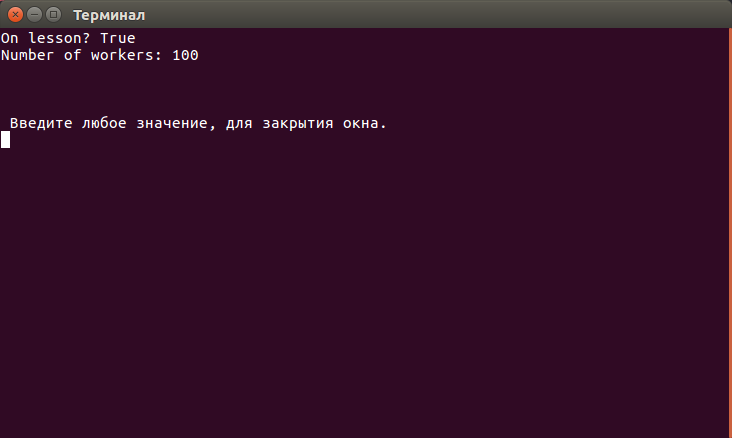


Рис. 44 Результат работы программы

## Лабораторная работа №8

**Бибилиотека STL**

**Цель практической работы**

Целью данной лабораторной работы является знакомство с библиотекой STL – стандартной библиотекой шаблонов - в языке С++, а также показать ее использование на примерах.

**Задачи**

1. Используйте шаблон vector для массива данных о студентах.
2. Используйте шаблон list для двусвязного списка данных класса Complex.
3. Используйте шаблон queue для очереди авто на мойке.

**Ход работы**

1. Программа №1

#include <iostream>

#include <vector>

#include <string>

using namespace std;

class group

{

string code;

vector<string> names;

public:

group();

int find(vector<string>, string);

void addName(string);

void removeName(string);

void getNames();

void setCode(string);

string getCode();

~group();

};

group::group()

{

}

int group::find(vector<string> v, string item) {

for (int i = 0; i < v.size(); i++)

{

if (v[i] == item) return i;

}

return -1;

}

void group::addName(string name) {

names.push\_back(name);

}

void group::removeName(string name) {

int status = find(names, name);

if (status >= 0)

names.erase(names.begin() + status);

else

cout << "Student name is missing.\n";

}

void group::getNames() {

for (int i = 0; i < names.size(); i++)

{

cout << i + 1 << ": " << names[i] << endl;

}

}

void group::setCode(string code) {

this->code = code;

}

string group::getCode() {

return code;

}

group::~group()

{

}

int main()

{

group group1;

string name, code;

cout << "Enter group code: " << endl;

getline(cin, code);

group1.setCode(code);

cout << "Enter name" << endl;

getline(cin, name);

group1.addName(name);

cout << "Enter second name" << endl;

getline(cin, name);

group1.addName(name);

cout << "Composition of the group:\n";

group1.getNames();

cout << "Enter name:" << endl;

getline(cin, name);

group1.removeName(name);

cout << "Composition of the group:\n";

group1.getNames();

cout << "Group code: " << group1.getCode() << endl;

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №2

#include <iostream>

#include <list>

using namespace std;

class Complex

{

int re, im;

public:

Complex(int re, int im) : re(re), im(im){}

void getData() {

cout << re << " + " << im << 'i' << endl;

}

};

int main()

{

list<Complex> d;

Complex a(1, 2), b(3, 4), c(5, 6);

d.push\_back(a);

d.push\_back(b);

d.push\_back(c);

d.back().getData();

d.pop\_back();

d.back().getData();

d.pop\_back();

d.back().getData();

d.pop\_back();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

1. Программа №3

#include <iostream>

#include <string>

#include <queue>

using namespace std;

class Car

{

string number, color, model;

public:

Car(string number, string color, string model):number(number),

color(color), model(model){}

void getSpecification() {

cout << "Model: " << model <<

"\nColor: " << color << "\nNumber: " << number << endl;

}

};

int main()

{

queue<Car> c;

Car c1("123", "blue", "123n");

Car c2("321", "red", "321n");

c.push(c1);

c.push(c2);

c.back().getSpecification();

cout << endl;

c.front().getSpecification();

cout << "\n \n \n Введите любое значение, для закрытия окна." << "\n";

cin.get();

cin.get();

return 0;

}

**Вывод**

Все поставленные задачи были выполнены. Была изучена библиотека STL. Для выполнения работы были использованы коллекции queue, string, list и vector. Результат работы программ изображён на рисунках (рис. 45-47). Исходный Файл: Programm\_.h программ так же доступен на GitHub по ссылкам:

1. <https://github.com/KorDima01/OOP/tree/master/Labs/8/8_1>
2. <https://github.com/KorDima01/OOP/tree/master/Labs/8/8_2>
3. <https://github.com/KorDima01/OOP/tree/master/Labs/8/8_3>

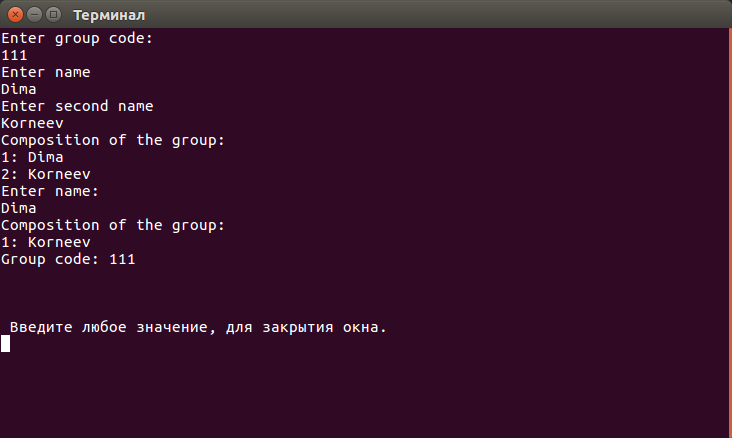


Рис. 45 Результат работы программы №1

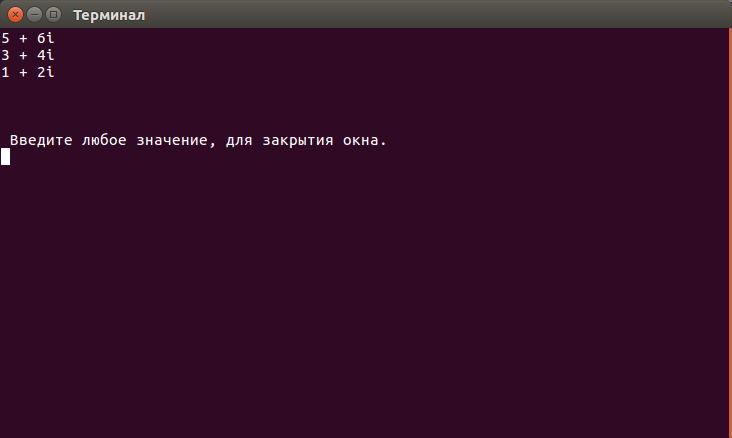


Рис. 46 Результат работы программы №2

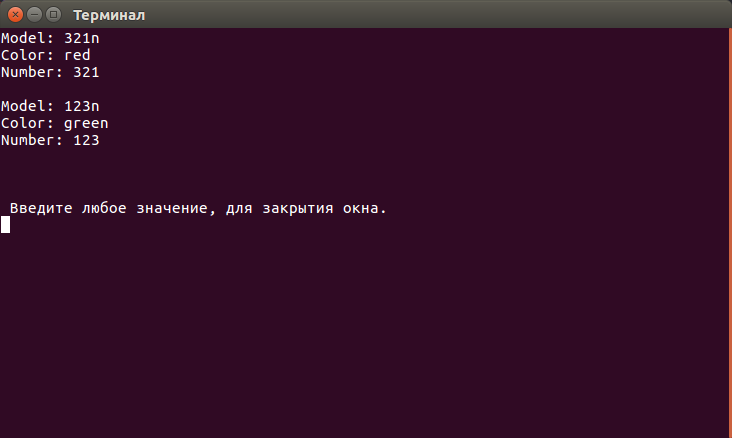


Рис. 47 Результат работы программы №3