**2 ЛАБОРАТОРНАЯ РАБОТА №2**

**«ИССЛЕДОВАНИЕ МЕХАНИЗМА ВИРТУАЛЬНЫХ ФУНКЦИЙ»**

**2.1 Цель работы**

Приобретение практических навыков при написании объектно-ориентированных программ с использованием механизмов наследования и виртуальных функций. Освоение особенностей отладки объектно-ориентированных программ.

**2.2 Вариант задания – 8**

Требуется создать абстрактный базовый класс - число с виртуальной функцией изменения знака числа. Создать производные классы: целое (int), вещественное (double) и комплексное (float, float). Для заданной иерархии описать классы, конструкторы и деструктор, функции ввода и вывода информации на экран. Базовый класс определить как абстрактный, а заданную функцию — как чисто виртуальную в базовом классе и переопределить ее в остальных классах иерархии. Проиллюстрировать корректную работу виртуальных функций и механизма наследования.

**2.3 Ход работы**

2.3.1 В программе создан абстрактный класс Number в котором содержится виртуальная функция, меняющая знак перед числом. Далее созданы 3 класса – наследника в каждом из которых есть по 2 конструктора, деструктор, функции ввода и вывода числа а также виртуальная функция. В функции main проведены действия с объектами этих трёх классов.

2.3.2 Написана программа на С++ согласно вышеописанного алгоритма.

#include <iostream>

#include <windows.h>

using namespace std;

class Number {

public:

virtual void change\_the\_sign() = 0;

};

class Int\_Number : public Number {

int i;

public:

Int\_Number() { i = 0; cout << "Int\_Number - конструктор по умолчанию" << endl; }

Int\_Number(int num) { i = num; cout << "Int\_Number - конструктор c параметрами" << endl; }

~Int\_Number() { cout << "Int\_Number - деструктор" << endl; system("pause"); }

void input\_int(int num) { i = num; }

void show\_int() { cout << "Целое число: " << i << endl; }

void change\_the\_sign() override {

if (i == 0) return;

i = -i;

}

};

class Double\_Number : public Number {

double d;

public:

Double\_Number() { d = 0; cout << "Double\_Number - конструктор по умолчанию" << endl; }

Double\_Number(double num) { d = num; cout << "Double\_Number - конструктор c параметрами" << endl; }

~Double\_Number() { cout << "Double\_Number - деструктор" << endl; }

void input\_double(double num) { d = num; }

void show\_double() { cout << "Double число: " << d << endl; }

void change\_the\_sign() override {

if (d == 0) return;

d = -d;

}

};

//flout real, imaginary;

class Complex\_Number : public Number {

float real, imagin;

public:

Complex\_Number() { real = imagin = 0; cout << "Complex\_Number - конструктор по умолчанию" << endl; }

Complex\_Number(float num1, float num2) { real = num1; imagin = num2; cout << "Complex\_Number - конструктор c параметрами" << endl; }

~Complex\_Number() { cout << "Complex\_Number - деструктор" << endl; }

void input\_Complex(float num1, float num2) { real = num1; imagin = num2; }

void show\_Complex() { cout << "Complex число: " << real << " + " << imagin << "i" << endl; }

void change\_the\_sign() override {

if ((real == 0) && (imagin == 0)) return;

real = -real; imagin = -imagin;

}

};

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "Здравствуйте!" << endl;

Number \*obj;

Int\_Number obj1;

obj1.show\_int();

obj = &obj1;

int num;

cout << "Введите целое число и мы его сделаем противоположным знаком :)" << endl;

cout << " -> "; cin >> num;

obj1.input\_int(num);

obj1.show\_int();

obj->change\_the\_sign();

obj1.show\_int();

Double\_Number obj2;

obj2.show\_double();

obj = &obj2;

double num2;

cout << "Введите double число и мы его сделаем противоположным знаком :)" << endl;

cout << " -> "; cin >> num2;

obj2.input\_double(num2);

obj2.show\_double();

obj->change\_the\_sign(); //изменили знак

obj2.show\_double();

Complex\_Number obj3;

obj3.show\_Complex();

obj = &obj3;

float num3, num4;

cout << "Введите два числа типа flout; это будет комплекное число, где" << endl;

cout << "первое число - реальная часть, а второе число мнимая часть" << endl;

cout << " -> "; cin >> num3;

cout << " -> "; cin >> num4;

obj3.input\_Complex(num3, num4);

obj3.show\_Complex();

obj->change\_the\_sign(); //изменили знак

obj3.show\_Complex();

system("pause");

return 0;

}

2.3.3 Выполнена отладка программы.

Результаты тестирования отображены на рисунке 2.1. На изображении видно как мы вводим число, и оно выводится с обратным значением.

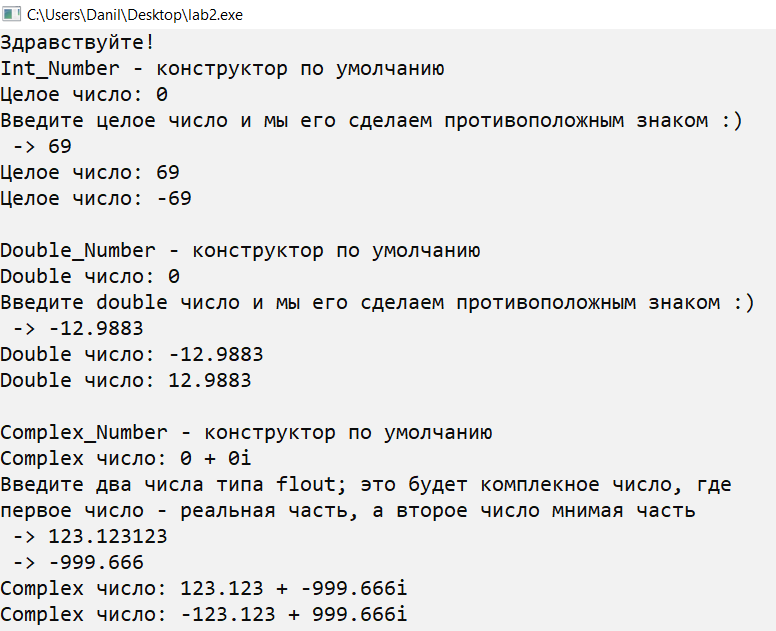


Рисунок 2.1 – Работа с программой

Результаты тестирования полностью соответствуют ожиданиям.

**Выводы**

В ходе выполнения данной лабораторной работы были получены навыки разработки программ, использующих виртуальные функции, абстрактные классы. Было изучено такое явление, как полиморфизм. Закреплены навыки разработки и отладки программ, использующих классы и объекты. Полученные во время разработки навыки помогут разрабатывать более сложные программы с использованием классов и объектов, более эффективные по времени выполнения алгоритмы.