Лабораторная работа №5

“Исследование перегрузки операторов”

1 ЦЕЛЬ РАБОТЫ

Исследование назначения и средств создания перегруженных операторов при написании объектно-ориентированных программ.

2 ПОСТАНОВКА ЗАДАЧИ

Для заданного по варианту класса выполнить следующие действия:

- описать конструкторы и деструктор (по необходимости);

- переопределить оператор вывода в поток <<;

- переопределить оператор ввода из потока >>;

- переопределить заданные по варианту операторы;

- предусмотреть обработку ошибок.

Создать два объекта заданного по варианту класса и на их примере продемонстрировать корректную работу всех перегруженных операторов.

Вариант 5

Создать класс координат Coord (содержит поля х, у и delta). Перегрузить операторы:

1) ++ как унарный метод класса, увеличивающий значение координат на шаг (delta);

2) ! как унарную дружественную функцию, проверяющую нулевое значение координат;

3) + как бинарный метод класса, складывающий два объекта (только координаты);

4) >= как бинарную дружественную функцию, сравнивающую среднее квадратичное значение координат двух объектов.

3 ТЕКСТ ПРОГРАММЫ

#include <iostream>

#include<cmath>

using namespace std;

class Coord

{

int x, y, delta;

public:

Coord (int \_x, int \_y, int \_delta)

{

x=\_x; y=\_y; delta=\_delta;

}

Coord & operator ++()

{

x+=delta;

y+=delta;

return \*this;

}

friend Coord & operator !(Coord &ob1);

Coord operator + (const Coord &ob2)

{

x=x+ob2.x;

y=y+ob2.y;

return \*this;

}

friend bool operator >=(const Coord &ob1, const Coord &ob2);

void Vivod ()

{

cout<<"Координата x: "<<x<<endl;

cout<<"Координата y: "<<y<<endl;

}

friend ostream & operator << (ostream& os, const Coord & ob);

friend istream & operator >> (istream& is, Coord &ob);

};

Coord & operator !(Coord &ob1)

{

if (!ob1.x && !ob1.y) cout<<"Координаты x1 и y1 равны 0!"<<endl;

else cout<<"Координаты x1 и y1 не равны 0!"<<endl;

return ob1;

}

bool operator >=(const Coord &ob1, const Coord &ob2)

{

if (sqrt(pow(ob1.x, 2)+pow(ob1.y, 2)/2)>=sqrt(pow(ob2.x, 2)+pow(ob2.y, 2)/2)) return true;

return false;

}

ostream & operator << (ostream& os, const Coord &ob)

{

os << ob.x << '|' << ob.y << '|' << ob.delta<<endl;

return os;

}

istream & operator >> (istream& is, Coord &ob)

{

is >> ob.x >> ob.y >> ob.delta;

return is;

}

int main ()

{

Coord ob1(0,0,0);

Coord ob2(0,0,0);

cout<<"Введите координаты 1 объекта: ";

cin>>ob1;

cout<<"Введите координаты 2 объекта: ";

cin>>ob2;

cout<<"Координаты 1 объекта: "<<ob1;

cout<<"Координаты 2 объекта: "<<ob2;

cout<<"---------------------"<<endl;

cout<<"Оператор !:"<<endl;

!ob1;

cout<<"---------------------"<<endl;

cout<<"Оператор ++:"<<endl;

++ob1;

ob1.Vivod();

cout<<"---------------------"<<endl;

cout<<"Оператор >=:"<<endl;

if (ob1>=ob2) cout<<"> или ="<<endl;

else cout<<"<"<<endl;

cout<<"---------------------"<<endl;

cout<<"Оператор +:"<<endl;

ob2=ob2+ob1;

ob2.Vivod();

return 0;

}

4 ТЕСТИРОВАНИЕ

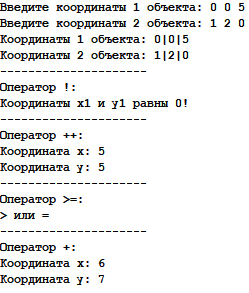


Рисунок 1 – Тест №1

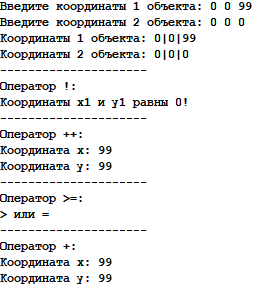


Рисунок 2 – Тест №2

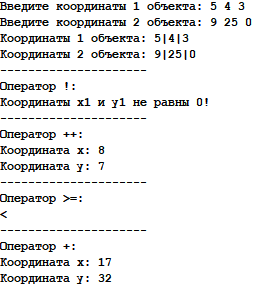


Рисунок 3 – Тест №3

ВЫВОД

В ходе выполнения лабораторной работы были получены навыки работы с перегрузкой операторов. Изучена перегрузка унарных и бинарных операторов. Была написана программа, описывающая два объекта заданного по варианту класса и на их примере продемонстрирована корректная работа всех перегруженных операторов.