Институт информационных технологий и управления в технических системах

Кафедра информационных технологий и компьютерных систем

ОТЧЁТ

по лабораторной работе №6

«ИССЛЕДОВАНИЕ ПЕРЕГРУЗКИ ОПЕРАТОРОВ»

по дисциплине «Объектно-ориентированное программирование»

Выполнил студент группы ПИН/б-19-1-о

Мельник А.С.

Проверил ассистент

Тимофеев И.С.

Севастополь

2020

**1. ЦЕЛЬ РАБОТЫ**

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

**2. ОПИСАНИЕ ПРОЕКТА**

Разработать программу на языке С++, в которой необходимо создать класс и перегрузить операторы. Для класса описать конструктор с параметрами. Продемонстрировать работу перегруженных операторов.

Разработать тестовые примеры.

Выполнить отладку программы.

Сформулировать выводы.

Оформить отчет по проделанной работе.

**3. АНАЛИЗ ЗАДАЧИ**

Для варианта задания, полученного в лабораторной работе №6, необходимо выполнить следующее:

Описать заданный по варианту класс. Для класса описать конструктор с параметрами, а также деструктор (по необходимости). Перегрузить операторы и продемонстрировать их работу.

Вариант 3

Создать класс вектор FVector (float \*). Перегрузить операторы:

1) ++ (префиксный) как унарный метод класса, увеличивающий элементы вектора вдвое;

2) -- (префиксный) как унарную дружественную функцию, уменьшающую значение каждого элемента вектора на 6;

3) + как бинарный метод класса, суммирующий два вектора и сохраняющий результат в первом;

4) > как бинарную дружественную функцию, определяющую, существует ли в первом векторе такой элемент, который превосходит по величине все элементы второго вектора

**4. ТЕКСТ С++ ПРОГРАММЫ, ЗАДАННОЙ ВАРИАНТОМ ЗАДАНИЯ**

#include <iostream>

using std::cout;

using std::endl;

class FVector

{

public:

int var1[2];

int var2[2];

FVector& operator ++()

{

var1[0] += 1;

var1[1] += 1;

var2[0] += 1;

var2[1] += 1;

return \*this;

}

friend FVector& operator --(FVector& F);

FVector operator + (FVector& obj2)

{

this->var1[0] += obj2.var1[0];

this->var1[1] += obj2.var1[1];

this->var2[0] += obj2.var2[0];

this->var2[1] += obj2.var2[1];

return \*this;

}

friend bool operator > (FVector& F, FVector& B);

friend std::ostream& operator << (std::ostream& out, const FVector& F);

friend std::istream& operator >> (std::istream& in, FVector& F);

FVector(int obj1[2], int obj2[2])

{

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

{

var1[i] = obj1[i];

var2[i] = obj2[i];

}

}

};

FVector& operator --(FVector& F)

{

F.var1[0] -= 6;

F.var1[1] -= 6;

F.var2[0] -= 6;

F.var2[1] -= 6;

return F;

}

bool operator > (FVector& F, FVector& B)

{

if ((F.var1[0] > B.var1[0] and F.var1[0] > B.var2[0] and F.var1[0] > B.var1[1] and F.var1[0] > B.var2[1]) or

(F.var2[0] > B.var1[0] and F.var2[0] > B.var2[0] and F.var2[0] > B.var1[1] and F.var2[0] > B.var2[1]))

return true;

else return false;

}

std::ostream& operator << (std::ostream& out, const FVector& F)

{

out << F.var1[0] << "/" << F.var1[1] << endl << F.var2[0] << "/" << F.var2[1] << endl;

return out;

}

std::istream& operator >> (std::istream& in, FVector& F)

{

in >> F.var1[0];

in >> F.var1[1];

in >> F.var2[0];

in >> F.var2[1];

return in;

}

int main()

{

setlocale(LC\_ALL, "rus");

int l1[] = { 18,10 };

int l2[] = { 14,16 };

int l3[] = { 7,0 };

FVector\* fv1 = new FVector(l1, l2);

FVector\* fv2 = new FVector(l3, l2);

cout << "bool:";

bool bl = \*fv1 > \*fv2;

cout << bl << endl;

cout << endl << "начальное:" << endl << \*fv1;

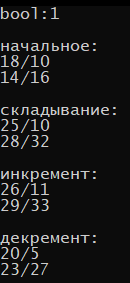
cout << endl << "складывание:" << endl << \*fv1 + \*fv2;

cout << endl << "инкремент:" << endl << fv1->operator++();

cout << endl << "декремент:" << endl << --(\*fv1);

}

**5. СВЕДЕНИЯ ОБ ОТЛАДКЕ ПРОГРАММЫ**

****

**6. ВЫВОД**

В ходе лабораторной работы исследованы основные механизмы перегрузки операторов.