Министерство образования и науки Российской Федерации

Федеральное государственное образовательное автономное учреждение высшего образования

"Пермский национальный исследовательский политехнический университет"

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

Дисциплина: Основы алгоритмизации и программирования

Тема: Перегрузка операций

|  |  |
| --- | --- |
| Выполнил работу | |
| Студент группы РИС-22-1б | |
| Деревнин И.В. | |
|  | |
| Проверил работу | |
| Доцент кафедры ИТАС | |
| Полякова О.А. | |
|  | |

Пермь – 2023

**Анализ предметной области**

**Постановка задачи**

1. Определить пользовательский класс.
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.
4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы).
5. Перегрузить операции: присваивания, вывода и ввода объектов с помощью потоков, операции указанные в варианте
6. Написать демонстрационную программу, в которой продемонстрировать создание объектов и работу всех перегруженных операций.

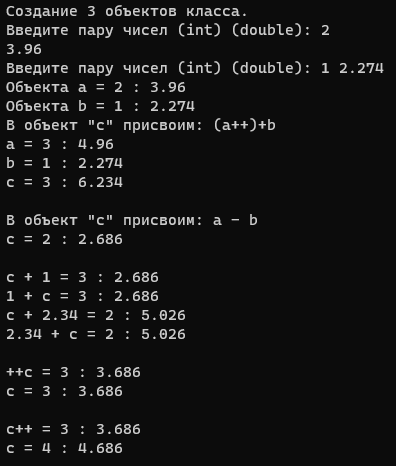
Вариант 15: создать класс Pair (пара чисел). Пара должна быть представлена двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием. Реализовать:

- вычитание пар чисел

- добавление константы к паре (увеличивается первое число, если константа целая, второе, если константа вещественная).

**Анализ задачи**

Класс Pair представляет из себя тип данных пара чисел, первое поле которого – имеет тип данных int, а второе double. Для данного типа данных перегрузить операторы: присваивания (метод), ввода и вывода (дружественные методы), вычитания пар чисел (метод), добавление константы к паре (дружественные методы). При добавлении константы необходимо учесть, что константу можно добавлять как справа, так и слева.

****Тестирование программы**

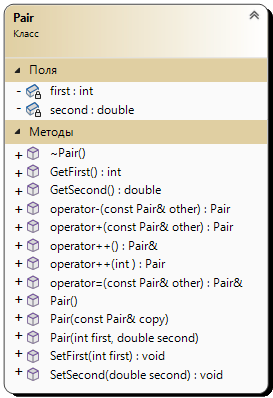
*Рис. 1 – Тестирование программы и класса Pair.*

**Заключение**

Была разработана программа, которая реализовывает собственный тип данных Pair. Для реализации данной программы был описан класс, который хранит в себе 2 поля, из которых составляется тип данных Pair, а также методы, благодаря которым можно управлять объектом класса и операторы, которые над типом данных позволяют производить такие же операции, как и с обычными типами данных (сложение, вычитание, присваивание).

**Приложения**

Приложение UML-диаграмма



Приложение Б – код программы

Main.cpp:

#include "Pair.h"

int main()

{

setlocale(LC\_ALL, "ru");

cout << "Создание 3 объектов класса." << endl;

Pair a, b, c;

cout << "Введите пару чисел (int) (double): ";

cin >> a;

cout << "Введите пару чисел (int) (double): ";

cin >> b;

cout << "Объекта а = " << a << endl;

cout << "Объекта b = " << b << endl;

cout << "В объект \"с\" присвоим: (a++)+b" << endl;

c = (a++) + b;

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

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

cout << "c = " << c << endl << endl;

cout << "В объект \"с\" присвоим: a - b" << endl;

c = a - b;

cout << "c = " << c << endl << endl;

cout << "c + 1 = " << c + 1 << endl;

cout << "1 + c = " << 1 + c << endl;

cout << "c + 2.34 = " << c + 2.34 << endl;

cout << "2.34 + c = " << 2.34 + c << endl << endl;

cout << "++c = " << ++c << endl;

cout << "c = " << c << endl << endl;

cout << "c++ = " << c++ << endl;

cout << "c = " << c << endl << endl;

return 0;

}

pair.h:

#pragma once

#include <iostream>

using namespace std;

class Pair

{

public:

Pair(); //конструктор

Pair(int first, double second); //конструктор с параметрами

Pair(const Pair& copy); //конструктор без параметров

~Pair() {} //деструктор

int GetFirst() { return this->first; } //геттеры и сеттеры для первого и второго поля.

double GetSecond() { return this->second; }

void SetFirst(int first) { this->first = first; }

void SetSecond(double second) { this->second = second; }

Pair& operator =(const Pair& other); //перегрузка оператора присваивания

Pair operator +(const Pair& other); //перегрузка оператора сложения 2-х пар чисел

Pair& operator ++(); //инкремент префиксный

Pair operator ++(int); //инкремент постфиксный

Pair operator -(const Pair& other); //оператор вычитания 2-х пар чисел

friend const Pair operator +(const Pair& left, const int& right); //добавление int константы справа и слева

friend const Pair operator +(const int& left, const Pair& right);

friend const Pair operator +(const Pair& left, const double& right);//добавление double константы справа и слева

friend const Pair operator +(const double& left, const Pair& right);

friend ostream& operator <<(ostream& os,const Pair& str); //перегрузка оператора вывода

friend istream& operator >> (istream& is, Pair& str); //перегрузка оператора ввода

private:

int first; //поле 1

double second; //поле 2

};

pair.cpp:

#include "Pair.h"

ostream& operator <<(ostream& os, const Pair& val)

{

os << val.first << " : " << val.second;

return os;

}

istream& operator>>(istream& is, Pair& val)

{

is >> val.first >> val.second;

return is;

}

Pair::Pair()

{

first = 0;

second = 0;

}

Pair::Pair(const Pair& copy)

{

this->first = copy.first;

this->second = copy.second;

}

Pair::Pair(int first, double second)

{

this->first = first;

this->second = second;

}

Pair& Pair::operator=(const Pair& other)

{

this->first = other.first;

this->second = other.second;

return \*this;

}

Pair Pair::operator+(const Pair& other)

{

Pair temp(this->first + other.first, this->second + other.second);

return temp;

}

Pair& Pair::operator++()

{

++this->first;

++this->second;

return \*this;

}

Pair Pair::operator++(int)

{

Pair temp(\*this);

++this->first;

++this->second;

return temp;

}

Pair Pair::operator-(const Pair& other)

{

Pair temp(this->first - other.first, this->second - other.second);

return temp;

}

const Pair operator+(const Pair& left, const int& right)

{

return Pair(left.first + right, left.second);

}

const Pair operator+(const int& left, const Pair& right)

{

return right + left;

}

const Pair operator+(const Pair& left, const double& right)

{

return Pair(left.first, left.second + right);

}

const Pair operator+(const double& left, const Pair& right)

{

return right + left;

}