Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Кафедра **«**Информационные технологии и автоматизированные системы**»**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

Дисциплина: «Основы алгоритмизации и программирования»

Семестр 2

Тема: Перегрузка операций

Вариант 12.

Выполнил работу

Студент группы РИС-22-1Б

Головин М.А.

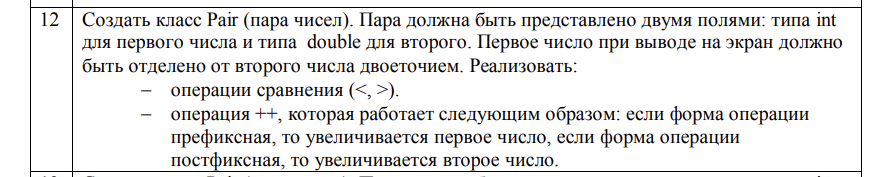
Проверил

Доцент кафедры ИТАС

Полякова О.А.

г. Пермь-2023

**Постановка задачи**



**Код программы**

**#include** <iostream>

#include "Header.h"

using namespace std;

int main() {

int a;

Pair kek, kekus;

cin >> kek >> kekus >> a;

if (a == 1)

kek > kekus;

else

kek < kekus;

cout << kek << " " << kekus << endl;

kek++;

cout << kek << endl;

++kek;

cout << kek << " " <<kekus;

return 0;

}

**#include** <iostream>

using namespace std;

class Pair {

int pairl;

double pairr;

public:

Pair() { pairl = 10; pairr = 100; }

Pair(const Pair& p) {

pairl = p.pairl;

pairr = p.pairr;

}

Pair(int a, double b) {

pairl = a;

pairr = b;

}

~Pair() {}

Pair& operator>(const Pair&);

Pair& operator<(const Pair&);

Pair& operator++();

Pair& operator ++(int);

friend istream& operator>>(istream&in, Pair&p);

friend ostream& operator<<(ostream&out, const Pair&p);

};

#include <iostream>

#include "Header.h"

using namespace std;

istream&operator>>(istream&in, Pair&p) {

cout << "left "; in >> p.pairl;

cout << "right "; in >> p.pairr;

return in;

}

ostream& operator<<(ostream& out, const Pair&p) {

return (out << p.pairl << " : " << p.pairr);

}

Pair& Pair::operator>(const Pair& p) {

if ((pairr > p.pairr) && (pairl > p.pairl)) return \*this;

else

{

pairr = p.pairr;

pairl = p.pairl;

return \*this;

}

}

Pair& Pair::operator<(const Pair&p) {

if ((pairr < p.pairr) && (pairl < p.pairl)) return \*this;

else

{

pairr = p.pairr;

pairl = p.pairl;

return \*this;

}

}

Pair& Pair::operator++() {

pairl++;

return \*this;

}

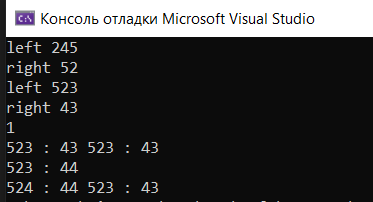
Pair&Pair::operator ++(int) {

pairr++;

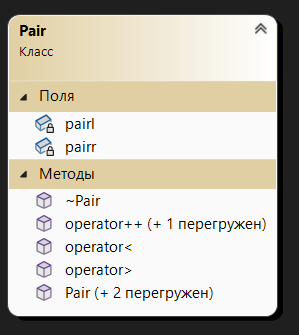
return \*this;

}

**Вывод программы**

****

**UML диаграмма**

****