Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**Тема:** Лабораторная работа №10 по ООП

Семестр: 2

Выполнил студент ИВТ-22-2б:

Мельников Глеб Владимирович

(дата, подпись)

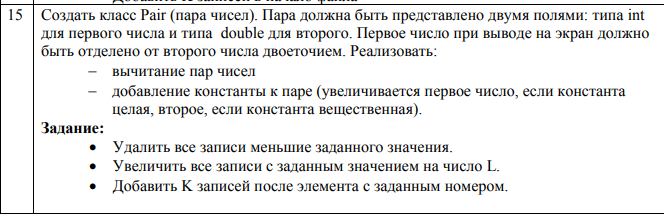
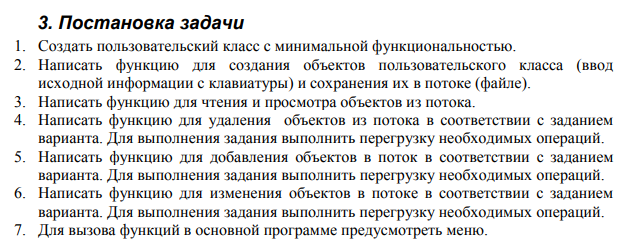
Проверила:

Полякова Ольга Андреевна

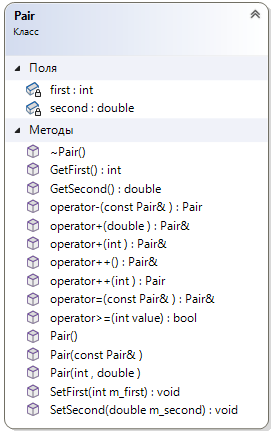
(дата, подпись)

Пермь 2023

**Задание**



**UML-Диаграмма**



**Программное решение**

**Pair.h**

#pragma once

#include <iostream>

using namespace std;

#include <fstream>

class Pair {

int first;

double second;

public:

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

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

Pair(int, double);

Pair();

Pair(const Pair&);

~Pair();

Pair& operator=(const Pair&);

Pair operator-(const Pair&);

Pair& operator+(int);

Pair& operator+(double);

Pair& operator++();

Pair operator++(int);

friend fstream& operator >> (fstream& fin, Pair& m\_Pair);

friend fstream& operator << (fstream& fout, const Pair& m\_Pair);

bool operator>=(int value);

void SetFirst(int m\_first);

void SetSecond(double m\_second);

int GetFirst() const;

double GetSecond() const;

};

**Pair.cpp**

#include "Pair.h"

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

this->first = first;

this->second = second;

}

istream& operator>>(istream& in, Pair& m\_pair) {//перегрузка оператора ввода

cout << "Введите данные в формате:\nцелое число вещественное число\n";

in >> m\_pair.first;

in >> m\_pair.second;

return in;

}

ostream& operator<<(ostream& out, const Pair& m\_pair) {//перегрузка оператора вывода

return (out << m\_pair.first << " : " << m\_pair.second);

}

fstream& operator>>(fstream& fin, Pair& m\_Pair) {

fin >> m\_Pair.first;

fin >> m\_Pair.second;

return fin;

}

fstream& operator<<(fstream& fout, const Pair& m\_Pair) {

fout << m\_Pair.first << ' ' << m\_Pair.second << '\n';

return fout;

}

Pair::Pair() {//конструктор без параметров

first = 0;

second = 0;

}

Pair::~Pair() {//деструктор

}

Pair::Pair(const Pair& m\_pair) {//конструктор копирования

first = m\_pair.first;

second = m\_pair.second;

}

Pair& Pair::operator=(const Pair& m\_pair) {//перегрузка оператора присваивания

if (&m\_pair != this) {

first = m\_pair.first;

second = m\_pair.second;

}

return \*this;

}

Pair Pair::operator-(const Pair& m\_pair) {//перегрузка оператора вычитания

Pair result(first - m\_pair.first, second - m\_pair.second);

return result;

}

Pair& Pair::operator+(int first) {//перегрузка оператора сложения - целый аргумент

this->first += first;

return \*this;

}

Pair& Pair::operator+(double second) {//перегрузка оператора сложения - дробный аргумент

this->second += second;

return \*this;

}

Pair& Pair::operator++() {//перегрузка оператора ++ префикс

++first;

++second;

return \*this;

}

Pair Pair::operator ++(int) {//перегрузка оператора ++ постфикс

Pair temp = \*this;

this->first++;

this->second++;

return temp;//сначала нужно вывести без изменений

}

bool Pair::operator>=(int value) {

return (this->first >= value && this->second >= value);

}

int Pair::GetFirst() const {

return first;

}

double Pair::GetSecond() const {

return second;

}

void Pair::SetFirst(int m\_first) {

first = m\_first;

}

void Pair::SetSecond(double m\_second) {

second = m\_second;

}

**File\_work.h**

#pragma once

#include "Pair.h"

#include <iostream>

using namespace std;

char f\_name[30];

int value;

void check\_k(int k) {

if (k < 0) {

cout << "Невозможно прочесть файл!\n";

}

}

int make\_file() {

cout << "Введите название файла: ";

cin >> f\_name;

fstream stream(f\_name, ios::out | ios::trunc);

if (!stream) {

return -1;

}

int n;

Pair m\_Pair;

cout << "Введите количество элиментов: ";

cin >> n;

for (int i = 0; i < n; i++) {

cin >> m\_Pair;

stream << m\_Pair << '\n';

}

stream.close();

return n;

}

int print\_file() {

cout << "Введите название файла: ";

cin >> f\_name;

fstream stream(f\_name, ios::in);

if (!stream) {

return -1;

}

Pair m\_Pair;

int i = 0;

while (stream >> m\_Pair) {

cout << m\_Pair << '\n';

i++;

}

if (i == 0) {

cout << "Файл пуст!\n";

}

stream.close();

return i;

}

int del\_file() {

cout << "Введите название файла: ";

cin >> f\_name;

cout << "Введите значение: ";

cin >> value;

fstream temp("temp", ios::out);

fstream stream(f\_name, ios::in);

if (!stream) {

return -1;

}

int i = 0;

Pair m\_Pair;

while (stream >> m\_Pair) {

if (stream.eof()) {

break;

}

i++;

if (m\_Pair >= value) {

temp << m\_Pair;

}

}

stream.close();

temp.close();

remove(f\_name);

rename("temp", f\_name);

return i;

}

int add\_file() {

int number;

cout << "Введите название файла: ";

cin >> f\_name;

cout << "Введите номер: ";

cin >> number;

cout << "Введите количество записей: ";

cin >> value;

fstream temp("temp", ios::out);

fstream stream(f\_name, ios::in);

int i = 1;

if (!stream) {

return -1;

}

Pair pair;

while (i != number && stream >> pair) {

if (stream.eof()) {

break;

}

i++;

temp << pair;

}

if (i == number) {

for (int j = 0; j < value; j++) {

Pair m\_Pair;

cin >> m\_Pair;

temp << m\_Pair;

}

}

while (stream >> pair) {

if (stream.eof()) {

break;

}

temp << pair;

}

temp.close();

stream.close();

remove(f\_name);

rename("temp", f\_name);

return value;

}

int change\_file() {

cout << "Введите название файла: ";

cin >> f\_name;

cout << "Введите значение: ";

cin >> value;

fstream temp("temp", ios::out);

fstream stream(f\_name, ios::in);

if (!stream) {

return -1;

}

Pair pair;

int i = 0;

while (stream >> pair) {

temp << pair + value;

}

temp.close();

stream.close();

remove(f\_name);

rename("temp", f\_name);

return i;

}

**OOPLab10.cpp**

#include <iostream>

using namespace std;

#include "file\_work.h"

using namespace std;

#include <string>

#include <functional>

int main() {

system("chcp 1251 > NULL");

int choose = 6;

function<int()> f[5] = {make\_file, print\_file, del\_file, change\_file, add\_file};

do {

if (choose != 6) {

check\_k(f[choose - 1]());

}

cout << "Введите номер команды:\n1. Создать файл\n2. Печать файла\n3. Удалить все записи меньше заданного значения\n4. Увеличить все записи с заданным значением на число L\n5. Добавить K записей перед элиментом с заданным номером\n6. Выход\n";

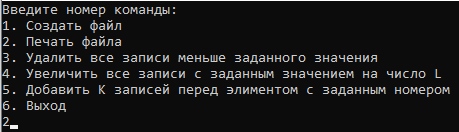
cin >> choose;

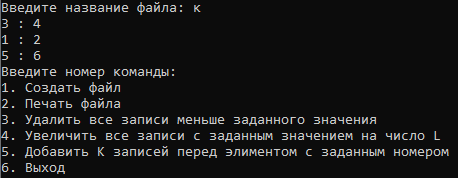
system("cls");

} while (choose != 6);

return 0;

}





**Контрольные вопросы**

1. Что такое поток?

Поток - последовательность байтов, не зависящая от конкретного устройства, с которым производится обмен

1. Какие типы потоков существуют?

* Стандартные
* Строковые
* Файловые

Все они делятся на однонаправленные и двунаправленные.

1. Какую библиотеку надо подключить при использовании стандартных потоков?

iostream

1. Какую библиотеку надо подключить при использовании файловых потоков?

fstream

1. Какую библиотеку надо подключить при использовании строковых потоков?

sstream

1. Какая операция используется при выводе в форматированный поток?

<<

1. Какая операция используется при вводе из форматированных потоков?

>>

1. Какие методы используются при выводе в форматированный поток?



1. Какие методы используется при вводе из форматированного потока?





1. Какие режимы для открытия файловых потоков существуют?



1. Какой режим используется для добавления записей в файл?

app

1. Какой режим (комбинация режимов) используется в конструкторе  
   ifstream file("f.txt")?

in

1. Какой режим (комбинация режимов) используется в конструкторе  
   fstream file("f.txt")?

in | out

1. Какой режим (комбинация режимов) используется в конструкторе  
   ofstream file("f.txt")?

out

1. Каким образом открывается поток в режиме ios::out|ios::app?

Открывается на запись в конец файла.

1. Каким образом открывается поток в режиме ios::out [ios::trunc?

Открывается для записи, предварительно очистив файл

1. Каким образом открывается поток в режиме ios::out ios::in/ios::trunk?

Открывается на ввод/ вывод данных с предварительной очисткой файла

1. Каким образом можно открыть файл для чтения?

Явно методом open и неявно при вызове конструктора

1. Каким образом можно открыть файл для записи?

Ответ: явно методом open и неявно при вызове конструктора

1. Привести примеры открытия файловых потоков в различных режимах.

ofstream output(filename);

ifstream input(filename);

ofstream base(filename, ios::trunc);

1. Привести примеры чтения объектов из потока.

* getline(file, s)
* file » s

1. Привести примеры записи объектов в поток.

Ответ: запись строки, чисел, символов

file << s

1. Сформулировать алгоритм удаления записей из файла.
2. Открыть 2 файла
3. Во время чтения данных из файла 1 проверять данные на соответствие условию
4. В случае соответствия записывать в файл 2
5. Закрыть файлы
6. Переименовать файл 2 в файл 1
7. Сформулировать алгоритм добавления записей в файл.
8. Открыть 2 файла.
9. Во время чтения данных из файла 1 и записи в файл 2 проверять данные на требование добавления записи после
10. Если так, то записать в файл 2 новые данные
11. Закрыть файлы.
12. Переименовать файл 2 в файл 1
13. Сформулировать алгоритм изменения записей в файле
14. Открыть 2 файла
15. Проверить каждый элемент в файле 1 на соответствие
16. Соответствующие элементы записать в файл 2 c измененными данными
17. Закрыть файлы
18. Переименовать файл 2 в файл 1