ПНИПУ

Федеральное государственное автономное образовательное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Лабораторная работа №10.

Сохранение данных в файле с использованием потоков.

Выполнил студент группы РИС-23-3Б

Буковский Денис Владимирович

Проверила доцент кафедры ИТАС О.А. Полякова

2024

1. Постановка задачи
2. Создать пользовательский класс с минимальной функциональностью.
3. Написать функцию для создания объектов пользовательского класса (ввод исходной информации с клавиатуры) и сохранения их в потоке (файле).

3. Написать функцию для чтения и просмотра объектов из потока.

4. Написать функцию для удаления объектов из потока в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

5. Написать функцию для добавления объектов в поток в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

6. Написать функцию для изменения объектов в потоке в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

7. Для вызова функций B основной программе предусмотреть меню.

Создать класс Pair (пара чисел).

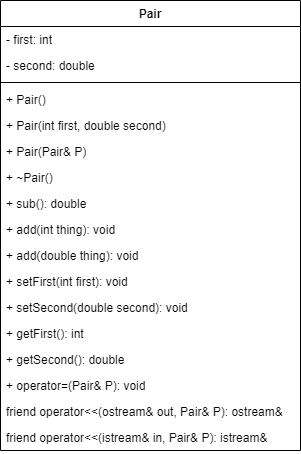
Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго.

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). Задание: — Удалить все записи меньшие заданного значения. — Увеличить все записи с заданным значением на число L. — Добавить K записей после элемента с заданным номером.

1. Анализ задачи

Возьмём реализацию класса Pair из прошлых лабораторных работ. Напишем функции для работы с файлами. Протестируем программу в функции main().

1. Блок-схема



1. Код

Pair.h

#pragma once

#include <iostream>

class Pair

{

private:

int first;

double second;

public:

void setFirst(int);

void setSecond(double);

int getFirst();

double getSecond();

Pair();

Pair(int, double);

~Pair();

friend std::istream& operator>>(std::istream&, Pair&);

friend std::ostream& operator<<(std::ostream&, const Pair&);

friend std::fstream& operator>>(std::fstream&, Pair&);

friend std::fstream& operator<<(std::fstream&, const Pair&);

Pair\* subPairs(Pair&, Pair&);

void add(int);

void add(double);

};

Pair.cpp

#include "Pair.h"

#include <fstream>

void Pair::setFirst(int first) {

this->first = first;

}

void Pair::setSecond(double second) {

this->second = second;

}

int Pair::getFirst() {

return this->first;

}

double Pair::getSecond() {

return this->second;

}

Pair::Pair() {

this->first = 0;

this->second = 0.0;

}

Pair::Pair(int first, double second) {

this->first = first;

this->second = second;

}

Pair::~Pair(){}

std::istream& operator>>(std::istream& in, Pair& pair) {

in >> pair.first >> pair.second;

return in;

}

std::ostream& operator<<(std::ostream& out, const Pair& pair) {

out << pair.first << " : " << pair.second << std::endl;

return out;

}

std::fstream& operator>>(std::fstream& fin, Pair& pair) {

fin >> pair.first;

fin >> pair.second;

return fin;

}

std::fstream& operator<<(std::fstream& fout, const Pair& pair) {

fout << pair.first << '\n' << pair.second << '\n';

return fout;

}

Pair\* Pair::subPairs(Pair& p1, Pair& p2) {

Pair\* new\_pair = new Pair;

new\_pair->first = p1.first - p2.first;

new\_pair->second = p1.second - p2.second;

return new\_pair;

}

void Pair::add(int first) {

this->first += first;

}

void Pair::add(double second) {

this->second += second;

}

File\_work.h

#pragma once

#include <iostream>

#include <fstream>

#include "Pair.h"

int make\_file(const char\* f\_name) {

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

if (!file) return -1;

int n;

Pair pair;

std::cout << "count: ";

std::cin >> n;

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

std::cin >> pair;

file << pair << "\n";

}

file.close();

return n;

}

int print\_file(const char\* f\_name) {

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0;

while (file >> pair) {

std::cout << pair << std::endl;

i++;

}

file.close();

return i;

}

int del\_by\_position(const char\* f\_name, int k) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

int i = 0;

Pair pair;

while (file >> pair) {

if (file.eof()) break;

i++;

if (i != k) tmp\_file << pair;

}

tmp\_file.close();

file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return i;

}

int add\_file(const char\* f\_name, int k, Pair new\_pair) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0, l = 0;

while (file >> pair) {

if (file.eof()) break;

i++;

if (i == k) {

tmp\_file << new\_pair;

l++;

}

tmp\_file << pair;

}

if (++i == k) {

tmp\_file << new\_pair;

l++;

}

file.close();

tmp\_file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return l;

}

int add\_end(const char\* f\_name, Pair new\_pair) {

std::fstream file(f\_name, std::ios::app);

if (!file) return -1;

file << new\_pair;

return 1;

}

int change\_file(const char\* f\_name, int k, Pair new\_pair) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0, l = 0;

char ch;

while (file >> pair) {

if (file.eof()) break;

i++;

if (i == k) {

std::cout << pair << " is being changed. Continue[y/n]?" << std::endl;

std::cin >> ch;

if (ch == 'n' || ch == 'N') break;

tmp\_file << new\_pair;

l++;

}

else tmp\_file << pair;

}

file.close();

tmp\_file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return l;

}

int delete\_lt(const char\* f\_name, int num) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0;

while (file >> pair) {

if (file.eof()) break;

if (pair.getFirst() >= num) {

tmp\_file << pair;

i++;

}

}

file.close();

tmp\_file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return i;

}

int increase\_l(const char\* f\_name, int l) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0;

while (file >> pair) {

if (file.eof()) break;

pair.setFirst(pair.getFirst() + l);

tmp\_file << pair;

i++;

}

file.close();

tmp\_file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return i;

}

int increase\_l(const char\* f\_name, double l) {

std::fstream tmp\_file("temp", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair;

int i = 0;

while (file >> pair) {

if (file.eof()) break;

pair.setFirst(pair.getFirst() + l);

tmp\_file << pair;

i++;

}

file.close();

tmp\_file.close();

std::remove(f\_name);

std::rename("temp", f\_name);

return i;

}

int add\_afterK(const char \* f\_name, int n, int k) {

std::fstream tmp1\_file("temp1", std::ios::out);

std::fstream tmp2\_file("temp2", std::ios::out);

std::fstream file(f\_name, std::ios::in);

if (!file) return -1;

Pair pair, tmp\_pair;

int i = 0;

while (file >> pair) {

if (file.eof()) break;

if (i < k) tmp1\_file << pair;

else tmp2\_file << pair;

i++;

}

file.close();

tmp2\_file.close();

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

std::cin >> pair;

tmp1\_file << pair;

i++;

}

tmp2\_file.open("temp2", std::ios::in);

while (tmp2\_file >> pair) {

if (file.eof()) break;

tmp1\_file << pair;

i++;

}

tmp2\_file.close();

tmp1\_file.close();

std::remove(f\_name);

std::remove("temp2");

std::rename("temp1", f\_name);

return i;

}

Main.cpp

#include <iostream>

#include "File\_work.h"

#include "Pair.h"

using namespace std;

int main() {

Pair pair, pair1, pair2;

int k, choice, num;

double tmp;

char file\_name[30];

do {

cout << endl << "1. Make file" << endl << "2. Print file" << endl << "3. Delete record from file" << endl << "4. Add record to file" << endl << "5. Change record in file" << endl << "6. Complete task" << endl << "0. Exit" << endl;

cin >> choice;

switch (choice) {

case 1: cout << "Enter file name: ";

cin >> file\_name;

k = make\_file(file\_name);

if (k < 0) cout << "couldn't make file!";

break;

case 2: cout << "Enter file name: ";

cin >> file\_name;

k = print\_file(file\_name);

if (k == 0) cout << "Empty file!" << endl;

if (k < 0) cout << "Couldn't read file!" << endl;

break;

case 3: cout << "Enter file name: ";

cin >> file\_name;

cout << "Enter number: ";

cin >> num;

k = del\_by\_position(file\_name, num);

if (k < 0) cout << "Couldn't read file!" << endl;

break;

case 4: cout << "Enter file name: ";

cin >> file\_name;

cout << "Enter number: ";

cin >> num;

cout << "New pair: ";

cin >> pair1;

k = add\_file(file\_name, num, pair1);

if (k < 0) cout << endl << "Couldn't read file!" << endl;

if (k == 0) cout << endl << "No such record!" << endl;

break;

case 5: cout << "Enter file name: ";

cin >> file\_name;

cout << "Enter number: ";

cin >> num;

cout << "New pair: ";

cin >> pair2;

k = change\_file(file\_name, num, pair2);

if(k<0) cout << endl << "Couldn't read file!" << endl;

if (k == 0) cout << endl << "No such record!" << endl;

break;

case 6: cout << "Enter file name: ";

cin >> file\_name;

cout << "Enter number to erase lesser numbers: ";

cin >> num;

k = delete\_lt(file\_name, num);

if (k < 0) cout << endl << "Couldn't read file!" << endl;

if (k == 0) cout << endl << "No such record!" << endl;

cout << "Enter L number to inclease existing pairs: " << endl << "1 - int, 2 - float: ";

cin >> choice;

cout << "Enter L: ";

switch (choice) {

case 1:

cin >> num;

k = increase\_l(file\_name, num);

break;

case 2:

cin >> tmp;

k = increase\_l(file\_name, tmp);

break;

}

if (k < 0) cout << endl << "Couldn't read file!" << endl;

if (k == 0) cout << endl << "No such record!" << endl;

cout << "Enter >k to enter new pairs after >n'th element: ";

cin >> k >> num;

k = add\_afterK(file\_name, num, k);

if (k < 0) cout << endl << "Couldn't read file!" << endl;

if (k == 0) cout << endl << "No such record!" << endl;

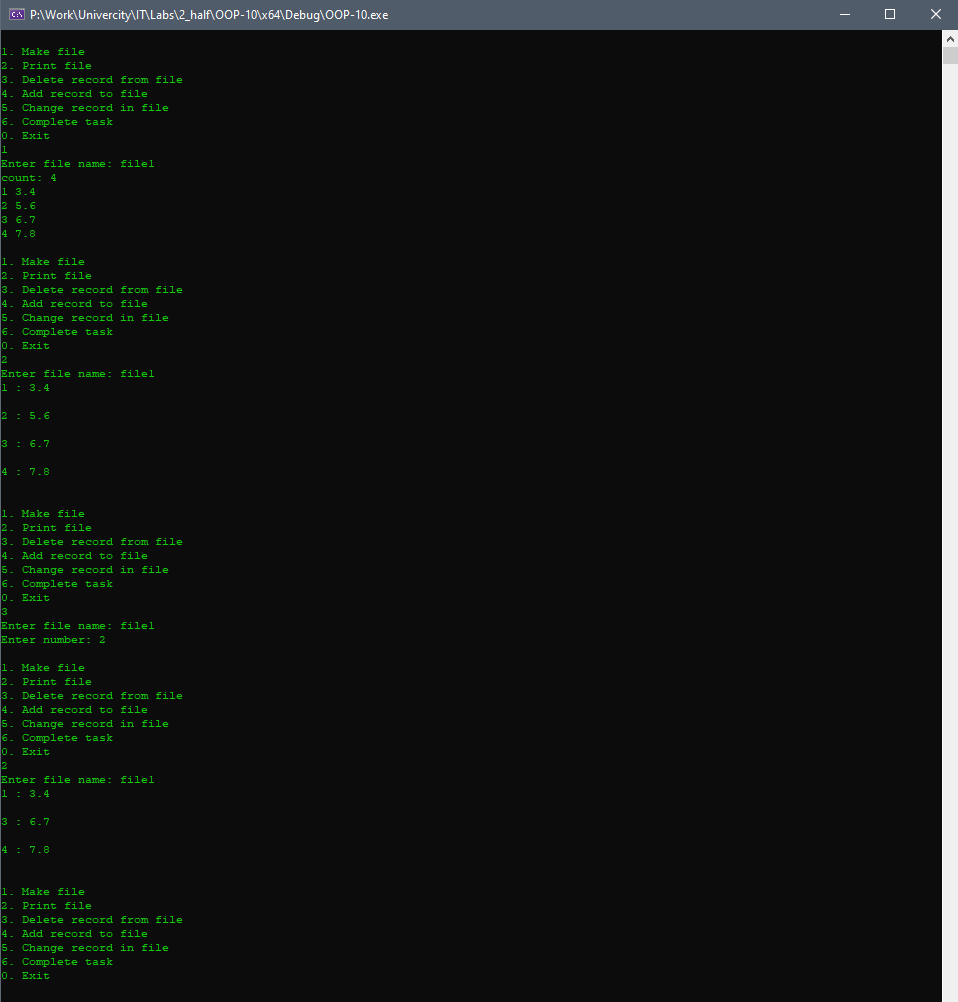
}

} while (choice != 0);

return 0;

}

1. Вывод программы



1. Ответы на контрольные вопросы
2. Поток - последовательность байтов
3. Стандартные, строковые, файловые
4. <iostream>
5. <fstream>
6. <sstream>
7. <<
8. >>
9. stream.write(s, strlen(s)).put(‘\n);
10. ostream& write(const char\* buf, streamsize size)
11. in, out, trunk, app, ate, binary
12. in, app
13. in
14. in | out
15. Дозапись в файл
16. Неизвестный режим - trunc
17. Стирание, чтение и запись
18. ifstream f1(“file.txt”)
19. ofstream f1(“file.txt”)
20. ifstream f1(“file.txt”), ofstream f1(“file.txt”), iofstream f1(“file.txt”)
21. f1 >> var
22. var << f1
23. создаётся поток в режиме trunk
24. var << f1
25. создаётся поток в режиме in (app)
26. var << f1
27. создаётся поток в режиме out | trunk
28. var << f1