Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.10**

Дисциплина: «Основы теории алгоритмов и структуры данных»  
Тема: Сохранение данных в файле с использованием потоков

Вариант 14

Выполнил:

студент группы РИС-20-2б

Морозова Е.М.

Проверила:

доцент кафедры ИТАС

Полякова Ольга Андреевна

Пермь, 2021

**Цель работы**

1. Создание консольного приложения, состоящего из нескольких файлов.

2. Разработка программы, в которой данные сохраняются в файле, корректируются и выводятся из файла на печать. Работа с файлом осуществляется с использованием потоковых классов.

**Постановка задачи**

1. Создать пользовательский класс с минимальной функциональностью.
2. Написать функцию для создания объектов пользовательского класса и сохранения их в файле.
3. Написать функцию для чтения и просмотра объектов из файла.
4. Написать функцию для удаления объектов из файла.
5. Написать функцию для добавления объектов в файл.
6. Написать функцию для изменения объектов в файле.
7. Перегрузить необходимые операции.
8. Для вызова функций в основной программе предусмотреть меню.

**Задание варианта:** Пользовательский 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.

-операция сравнения (<,>).

-операция ++, которая работает следующим образом: если форма операции префиксная, то увеличивается первое число, если форма операции постфиксная, то увеличивается второе число.

Требуется:

1. Удалить все записи большие заданного значения.

2. Увеличить все записи с заданным значением на число L.

3. Добавить K записей после записи с номером N.

**Анализ задачи**

1. Описание классов:

class Pair

{

public:

Pair();

Pair(int, double);

Pair(const Pair&);

Pair& operator=(const Pair&);

Pair& operator++();

Pair operator++(int);

Pair operator+(const Pair&);

friend bool operator<(const Pair& p1, const Pair& p2) { return p1.ch1 < p2.ch1; }

friend bool operator>(const Pair& p1, const Pair& p2) { return p1.ch1 > p2.ch1; }

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

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

friend fstream& operator>>(fstream& fin, Pair& p);

friend fstream& operator<<(fstream& fout, const Pair& p);

friend int change\_file(const char\*, int, int);

public:

~Pair();

private:

int ch1;

double ch2;

};

2. Определение компонентных функций:  
Pair::Pair()

{

ch1 = 0; ch2 = 0;

}

Pair::Pair(int A, double B)

{

ch1 = A;

ch2 = B;

}

Pair::Pair(const Pair& p)

{

ch1 = p.ch1;

ch2 = p.ch2;

}

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

{

if (&p == this) return \*this;

ch1 = p.ch1;

ch2 = p.ch2;

return \*this;

}

Pair& Pair::operator++()

{

if (ch1 > ch2)

cout << ch1 << " > " << ch2 << endl;

else

cout << ch2 << " > " << ch1 << endl;

ch1++;

return\*this;

}

Pair::~Pair()

{

}

Pair Pair::operator++(int)

{

int h1 = ch1;

double h2 = ch2;

if (h1 > h2)

cout << h1 << " > " << h2 << endl;

else

cout << h2 << " > " << h1 << endl;

Pair p(ch1, ch2);

ch2++;

return p;

}

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

{

int h1 = ch1;

double h2 = ch2;

if (h1 > h2)

{

cout << h1 << " > " << h2 << endl;

}

else {

cout << h2 << " > " << h1 << endl;

}

Pair t;

t.ch1++;

t.ch2++;

return t;

}

3. Определение глобальных функций:  
int make\_file(const char\* f\_name)

{

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

if (!stream) return -1;

int n;

Pair p;

cout << "N=";

cin >> n;

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

{

cin >> p;

stream << p << "\n";

}

stream.close();

return n;

}

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

{

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

if (!stream) return -1;

Pair p;

int i = 0;

while (stream >> p)

{

cout << p << "\n";

i++;

}

stream.close();

return i;

}

int del\_file(const char\* f\_name, Pair& m)

{

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

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

if (!stream) return -1;

Pair p;

int i = 0;

while (stream >> p)

{

if (stream.eof())break;

i++;

if (!(p > m)) {

temp << p;

}

}

stream.close(); temp.close();

remove(f\_name);

rename("temp", f\_name);

return i;

}

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

{

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

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

if (!stream) return -1;

Pair p;

int i = 0,f = 0;

while (stream >> p)

{

if (stream.eof())break;

i++;

while (i > n && k > 0) {

Pair pp;

cin >> pp;

temp << pp;

k--;

f = 1;

i++;

}

temp << p;

}

stream.close(); temp.close();

remove(f\_name);

rename("temp", f\_name);

return f;

}

int add\_end(const char\* f\_name, int k)

{

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

if (!stream) return -1;

for (int i = k; i > 0; i--) {

Pair pp;

cin >> pp;

stream << pp;

}

return 1;

}

int change\_file(const char\* f\_name, int k, int L)

{

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

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

if (!stream) return -1;

Pair p;

int i = 0,c=0;

while (stream >> p) {

if (stream.eof()) break;

i++;

if (p.ch1 == k) {

cout << p << " - is changing...";

p.ch1 += L;

c++;

}

temp << p;

}

stream.close(); temp.close();

remove(f\_name);

rename("temp",f\_name);

return c;

}

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

{

cout << "ch1? "; in >> p.ch1;

cout << "ch2? "; in >> p.ch2;

return in;

}

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

{

return (out << p.ch1 << " : " << p.ch2);

}

fstream& operator>>(fstream& fin, Pair& p) {

fin >> p.ch1;

fin >> p.ch2;

return fin;

}

fstream& operator<<(fstream& fout, const Pair& p) {

fout << p.ch1 << "\n" << p.ch2 << "\n";

return fout;

}

4. Функция main():

void main()

{

Pair a;

Pair b;

Pair c;

cin >> a;

cin >> b;

cout << a << endl;

cout << b << endl;

b = a;

cout << b << endl;

++a;

cout << a << endl;

c = (a++) + b;

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

Pair p;

int flag, s,k;

char file\_name[30];

do

{

cout << "\n1. Make file";

cout << "\n2. Print file";

cout << "\n3. Delete record from file";

cout << "\n4. Add record to file";

cout << "\n5. Change record in file";

cout << " \n0.Exit\n";

cin >> s;

cout << "file name:"; cin >> file\_name;

switch(s)

{

case 1: {

flag = make\_file(file\_name);

if (flag < 0) cout << "cant't make file";

break;

}

case 2: {

flag = print\_file(file\_name);

if (flag == 0)cout << "Empty file\n";

if (flag < 0) cout << "Can't read file\n";

break;

}

case 3: {

Pair o; cout << "o:"; cin >> o;

flag = del\_file(file\_name, o);

if (flag < 0) cout << "Can't read file\n";

break;

}

case 4: {

int n;

cout << "K:"; cin >> k;

cout << "N: ";

cin >> n;

flag = add\_file(file\_name, k, n);

if (flag < 0) cout << "Can't read file\n";

if (flag == 0) k = add\_end(file\_name,k);

break; }

case 5: {

int L;

cout << "K:"; cin >> k;

cout << "L: ";

cin >> L;

flag = change\_file(file\_name, k, L);

if (flag < 0) cout << "Can't read file\n";

if (flag == 0) cout << "\n Not such record";

break;

}

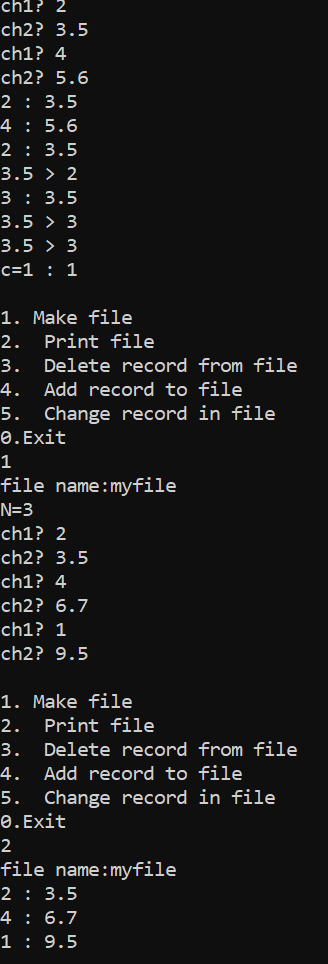
}

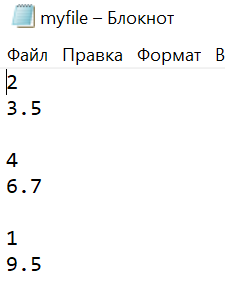
}

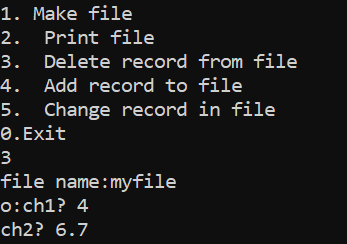
while (s != 0);

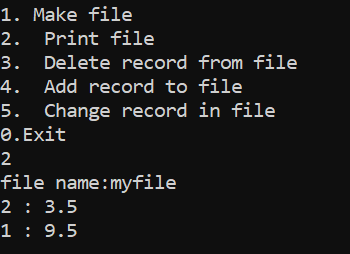
}

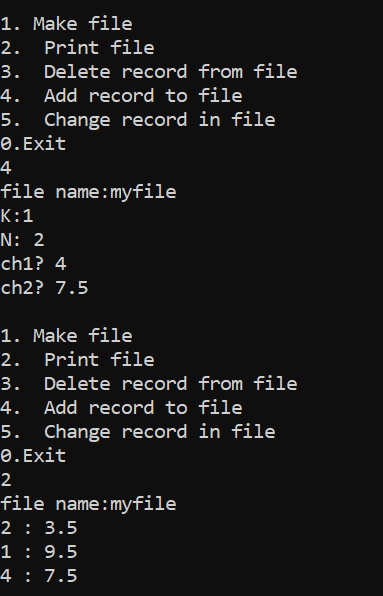
**Результаты работы программы**

****

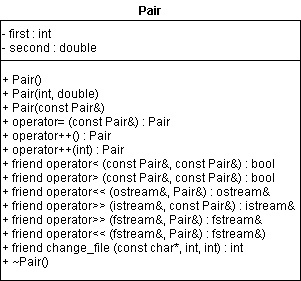
****

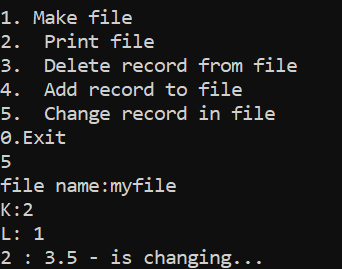
****

****

****

**Диаграмма классов**

****

****

**Ответы на вопросы**

1. Последовательность байтов, не зависящая от конкретного устройства.
2. Стандартные, строковые, файловые.
3. Iostream
4. Fstream
5. Stringstream
6. Операция сдвига влево.
7. Операция сдвига вправо.
8. Put, write.
9. Read, get, getline.
10. In, out, trunk, app, ate, binary.
11. App.
12. In.
13. In|out.
14. Out.
15. Для добавления элементов в поток (запись).
16. Для удаления старого содержимого потока (запись).
17. Для удаления старого содержимого потока (чтение и запись).
18. ifstream file(“f.txt”).
19. ofstream file(“f.txt”).
20. fstream file(“f.txt”,ios::in|ios::out);
21. file.getline(s,10);
22. file.put(simvol);
23. Скопировать в другой файл (или в оперативную память) все элементы, которые не подлежат удалению. По желанию заменить содержимое первого файла содержимым второго файла.
24. Скопировать в другой файл все элементы, стоящие до добавляемого, вставить добавляемый элемент, скопировать из первого файла все оставшиеся элементы.
25. Скопировать все элементы из файла в оперативную память, изменить необходимые, очистить файл и вставить в него элементы из оперативной памяти.