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

Лабораторная работа №10
«Сохранение данных в файле с использованием потоков»

Выполнил:
студент первого курса
ЭТФ группы РИС-23-3б
Акбашева Софья Руслановна

Проверила:
Доцент кафедры ИТАС О. А. Полякова

Пермь, 2024

Сохранение данных в файле с использованием потоков

Цель задания

1) Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.

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 (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 в основной программе предусмотреть меню.

Задание

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.

Анализ задачи

1) Необходимо реализовать класс Pair. В классе надо прописать конструкторы: по умолчанию, с параметрами и копирования; а также деструктор, для предотвращения утечки памяти.

2) В классе Pair необходимо прописать операторы перегрузки для необходимых операций.

3) Для работы с файлами необходимо подключить директиву #include <fstream>

4) Написать функцию для создания объектов пользовательского класса (ввод исходной информации с клавиатуры) и сохранения их в потоке (файле).

5) Необходимо написать функцию для чтения и просмотра объектов из потока.

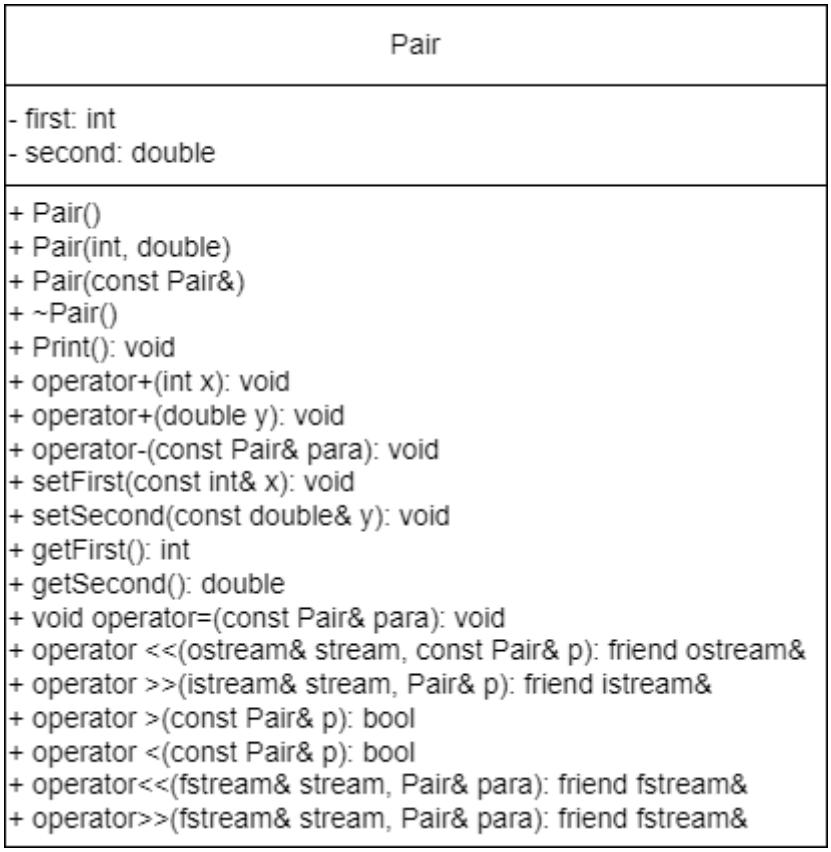
6) Необходимо написать функцию для удаления объектов из потока в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

7) Необходимо написать функцию для добавления объектов в поток в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

8) Необходимо написать функцию для изменения объектов в потоке в соответствии с заданием варианта. Для выполнения задания выполнить перегрузку необходимых операций.

9) Для выполнения операций надо предусмотреть меню в основной функции.

UML диаграмма



Код

Файл Pair.cpp

```
#include "Pair.h"
using namespace std;

Pair::Pair() {
	this->first = 0;
	this->second = 0;
}

Pair::Pair(int first, double second) {
	this->first = first;
	this->second = second;
}

Pair::Pair(const Pair& other) {//конструктор копирования
	this->first = other.first;
	this->second = other.second;
}

Pair::~Pair() {//деструктор
}

void Pair::Print() {//вывод параметров класса person
	cout << "(" << this->first << " : " << this->second << ")" << endl;
}

Pair Pair::operator+(Pair& p) {
	return (Pair(this->first + p.first, this->second + p.second));
}
```

```
Pair Pair::operator-(const Pair& para) {
	return (Pair(this->first - para.first, this->second - para.second));
}

Pair Pair::operator+(const int& x) {
	return (Pair(this->first + x, this->second));
}

Pair Pair::operator+(const double& y) {
	return (Pair(this->first, this->second + y));
}

void Pair::setFirst(const int& x) {
	this->first = x;
}

void Pair::setSecond(const double& y) {
	this->second = y;
}

int Pair::getFirst() {
	return this->first;
}

double Pair::getSecond() {
	return this->second;
}

void Pair::operator=(const Pair& para) {
	this->first = para.first;
	this->second = para.second;
}

ostream& operator <<(ostream& stream, const Pair& p) {
	stream << p.first << " : " << p.second;
	return stream;
}
istream& operator >>(istream& stream, Pair& p) {
	cout << "Введите число first и second через пробел: ";
	stream >> p.first >> p.second;
	return stream;
}

bool Pair::operator >(const Pair& p) {
	if (this->first < p.first) {
		return false;
	}
	if (this->first == p.first && this->second < p.second) {
		return false;
	}
	return true;
}

bool Pair::operator <(const Pair& p) {
	if (this->first > p.first) {
		return false;
	}
	if (this->first == p.first && this->second > p.second) {
		return false;
	}
	return true;
```

```
}

fstream& operator<<(fstream& stream, Pair& para) {
	stream << para.first << endl << para.second << endl;
	return stream;
}

fstream& operator>>(fstream& stream, Pair& para) {
	stream >> para.first;
	stream >> para.second;
	return stream;
}
```

Файл Pair.h

```
#pragma once //предотвращает повторную загрузку заголовочного файла, если он уже был включен
#include <iostream> //стандартные потоки ввода и вывода
#include <fstream>
using namespace std;

class Pair {
	int first;
	double second;

public:
	Pair();//конструктор класса по умолчанию
	Pair(int, double);//конструктор класса с параметрами
	Pair(const Pair&);//конструктор копирования класса
	~Pair();//деструктор класса
	void Print();

	Pair operator+(Pair& p);
	Pair operator-(const Pair& para);
	Pair operator+(const int& x);
	Pair operator+(const double& y);

	void setFirst(const int& x);
	void setSecond(const double& y);
	int getFirst();
	double getSecond();
	void operator=(const Pair& para);
	friend ostream& operator <<(ostream& stream, const Pair& p);
	friend istream& operator >>(istream& stream, Pair& p);
	bool operator >(const Pair& p);
	bool operator <(const Pair& p);
	friend fstream& operator<<(fstream& stream, Pair& para);
	friend fstream& operator>>(fstream& stream, Pair& para);
};
```

Файл file_work.h

```
#pragma once
using namespace std;

void randomize(Pair& p) {
	int x = rand() % 1001;
	double y = (rand() % 100001) * 0.01;
	p = Pair(x, y);
}

bool write_to_a_file(int N) {
	fstream F1("Class_10.txt", ios::out | ios::trunc); //открываю файл для записи
```

```
	//out - перезаписывание
	//trunk - удалить старое содержимое файла (по умолчанию для ofstream)
	if (!F1) {//если файл не открылся
		cout << "Ошибка открытия файла" << endl;
		return 0;
	}

	for (int i = 0; i < N; i++) {//записываю N пар
		Pair p_1;
		randomize(p_1);
		F1 << p_1 << endl;
	}
	F1.close();//закрываю файл
	return 1;
}

bool print_file() {
	fstream F1("Class_10.txt", ios::in);//открываю файл для чтения
	//in чтение

	if (!F1) {
		cout << "Ошибка открытия файла" << endl;
		return 0;
	}

	Pair p_1;

	while (F1 >> p_1) {//пока не считаю весь файл
		cout << p_1 << endl;
	}
	return 1;
}
void error_mes(bool N) {
	if (!N) {
		cout << "Файл не найден" << endl;
	}
}

bool removing_el_smaller_than(Pair& para) {
	fstream F1("Class_10.txt", ios::in);
	//in чтение
	if (!F1) {
		cout << "Ошибка открытия файла" << endl;
		return 0;
	}

	fstream tmp("temporary_file.txt", ios::out | ios::trunc); //создаю временный файл
	//out - перезаписывание
	//trunk - удалить старое содержимое файла (по умолчанию для ofstream)

	Pair p_3;

	while (F1 >> p_3) {//пока не пройду весь файл
		if (p_3 > para) { //если меньше элемента
			tmp << p_3 << endl;//записываю в tmp
		}
	}

	F1.close();//закрываю файл
	tmp.close();//закрываю файл

	remove("Class_10.txt");//удаляю основной файл
```

```
	rename("temporary_file.txt", "Class_10.txt");//переназначаю основной файл
	return 1;
}

template<typename T>
bool add_L(T L) {
	fstream F1("Class_10.txt", ios::in);
	//in чтение

	if (!F1) {
		cout << "Ошибка открытия файла" << endl;
		return 0;
	}

	fstream tmp("temporary_file.txt", ios::out | ios::trunc); //создаю временный файл
	//out - перезаписывание
	//trunk - удалить старое содержимое файла (по умолчанию для ofstream)

	Pair p_4;

	while (F1 >> p_4) {
		p_4 = p_4 + L;
		tmp << p_4 << endl;
	}

	F1.close();//закрываю файл
	tmp.close();//закрываю файл

	remove("Class_10.txt");//удаляю основной файл
	rename("temporary_file.txt", "Class_10.txt");//переназначаю основной файл
	return 1;
}

bool add_k_el_after(int number, int quantity) {
	fstream F1("Class_10.txt", ios::in);
	//in чтение

	if (!F1) {
		cout << "Ошибка открытия файла" << endl;
		return 0;
	}

	fstream tmp("temporary_file.txt", ios::out | ios::trunc); //создаю временный файл
	//out - перезаписывание
	//trunk - удалить старое содержимое файла (по умолчанию для ofstream)

	Pair p_4;

	int counter = 0;

	while (F1 >> p_4) {
		counter++;
		tmp << p_4 << endl;

		if (counter == number) {//когда достигаю заданного номера
			for (int i = 0; i < quantity; i++) {
				Pair new_pair;
				randomize(new_pair);
				tmp << new_pair << endl;
			}
		}
	}
```

```
	if (counter < number)
		cout << "Достигнут конец файла! Элемент с номером " << number << "не найден" << endl;

	F1.close();//закрываю файл
	tmp.close();//закрываю файл

	remove("Class_10.txt");//удаляю основной файл
	rename("temporary_file.txt", "Class_10.txt");//переназначаю основной файл
	return 1;
}

int choosing_an_action() {
	cout << "-----------------------------------------------------" << endl;
	cout << "Выберите действие" << endl;
	cout << "1 - Вывести содержимое файла" << endl;//+
	cout << "2 - Перезаписать содержимое файла" << endl;//+
	cout << "3 - Удалить все записи меньшие заданного значения" << endl;//+
	cout << "4 - Увеличить все записи с заданным значением на число L" << endl;
	cout << "5 - Добавить K записей после элемента с заданным номером" << endl;
	cout << "0 - Конец работы" << endl;//+
	int choice;
	do {
		cout << "> ";
		cin >> choice;
	} while (choice < 0 || choice > 6);
	cout << endl;
	return choice;
}
```

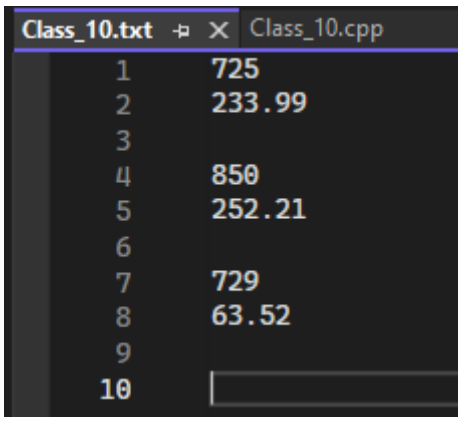
Файл Class_10.cpp

```
#include <iostream>
#include <fstream>
#include "Pair.h"
#include "file_work.h"

using namespace std;

int main() {
	system("chcp 1251 > Null");
	srand(time(0));
	int N;
	do
	{
		cout << "Введите кол-в0 элементов: ";
		cin >> N;
	} while (N < 1);

	cout << endl;

	error_mes(write_to_a_file(N));

	bool flag = true;
	while (flag) {
		int current_action = choosing_an_action();
		int tmp_1;
		double tmp_2;

		Pair p_2;
		switch (current_action){
		case 1://содержимое файла
```

```
                cout << "Текущее содержимое файла" << endl;
                error_mes(print_file());
                cout << endl;
                break;
        case 2:
                do {
                        cout << "Введите кол-в0 элементов: ";
                        cin >> tmp_1;
                } while (tmp_1 < 1);
                error_mes(write_to_a_file(tmp_1));
                break;
        case 3:
                cout << "Введите значение, меньше которого, будут удалены все записи " << endl;
                cin >> p_2;
                error_mes(removing_el_smaller_than(p_2));
                break;
        case 4:
                do{
                        cout << "Какое число вы хотите ввести?" << endl;
                        cout << "1 - целое" << endl << "2 - вещественное" << endl;
                        cin >> N;
                } while (N < 1 || N>2);
                switch (N){
                case 1:
                        cout << "Введите ЦЕЛОЕ число: ";
                        cin >> tmp_1;
                        error_mes(add_L(tmp_1));
                        break;
                default:
                        cout << "Введите ВЕЩЕСТВЕННОЕ число: ";
                        cin >> tmp_2;
                        error_mes(add_L(tmp_2));
                        break;
                }
                break;
        case 5:
                do {
                        cout << "Введите НОМЕР, после которого необходимо ввести k элементов" <<
endl;
                        cin >> N;
                } while (N < 1);

                do {
                        cout << "Введите количество элементов, которые необходимо добавить" <<
endl;
                        cin >> tmp_1;
                } while (tmp_1 < 1);
                error_mes(add_k_el_after(N, tmp_1));
                break;
        default:
                flag = false;
                break;
        }
    }
    return 0;
}
```

Результаты работы

```
Введите кол-в0 элементов: 5

------------------------------------------------------
Выберите действие
1 - Вывести содержимое файла
2 - Перезаписать содержимое файла
3 - Удалить все записи меньшие заданного значения
4 - Увеличить все записи с заданным значением на число L
5 - Добавить K записей после элемента с заданным номером
0 - Конец работы
>  1

Текущее содержимое файла
725 : 233.99
116 : 87.44
850 : 252.21
285 : 37.63
729 : 63.52

------------------------------------------------------
Выберите действие
1 - Вывести содержимое файла
2 - Перезаписать содержимое файла
3 - Удалить все записи меньшие заданного значения
4 - Увеличить все записи с заданным значением на число L
5 - Добавить K записей после элемента с заданным номером
0 - Конец работы
>  3

Введите значение, меньше которого, будут удалены все записи
Введите число first и second через пробел: 300 456.4
------------------------------------------------------
Выберите действие
1 - Вывести содержимое файла
2 - Перезаписать содержимое файла
3 - Удалить все записи меньшие заданного значения
4 - Увеличить все записи с заданным значением на число L
5 - Добавить K записей после элемента с заданным номером
0 - Конец работы
>  1

Текущее содержимое файла
725 : 233.99
850 : 252.21
729 : 63.52

------------------------------------------------------
Выберите действие
1 - Вывести содержимое файла
2 - Перезаписать содержимое файла
3 - Удалить все записи меньшие заданного значения
4 - Увеличить все записи с заданным значением на число L
5 - Добавить K записей после элемента с заданным номером
0 - Конец работы
>  0
```

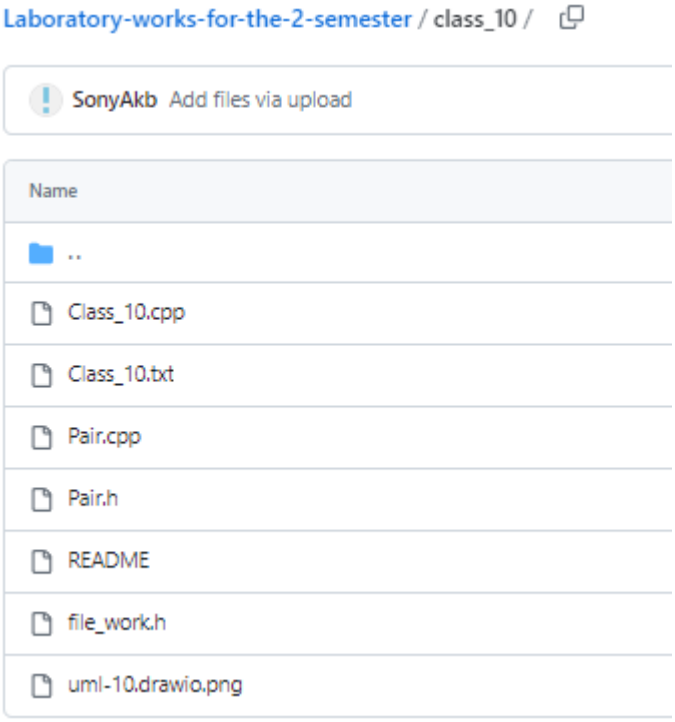


Вывод

В ходе работы я применила знания о работе с потоками и файлами. Был создан пользовательский класс Pair для представления пары чисел. Класс имеет два поля: first типа int и second типа double. Были также реализованы функции для работы с потоком (файлом), включая чтение, просмотр, удаление, добавление и изменение объектов класса Pair. Выполнение задания включало в себя: 1. Удаление всех записей меньших заданного значения. 2. Увеличение всех записей с заданным значением на число L. 3. Добавление K записей после элемента с заданным номером. Все функции были вызваны в основной программе через меню.

GitHub

Ссылка: https://github.com/SonyAkb/Laboratory-works-for-the-2-semester/tree/main/class_10



Контрольные вопросы

1. Что такое поток?

Поток в программировании — это последовательность данных, которая может быть прочитана или записана. Поток - определяется как последовательность байтов и не зависит от конкретного устройства, с которым производится обмен (оперативная память, файл на диске, клавиатура или принтер). Обмен с потоком для увеличения скорости передачи данных производится, как правило, через специальную область оперативной памяти — буфер. Буфер накапливает байты, и фактическая передача данных выполняется после заполнения буфера. При вводе это дает возможность исправить ошибки, если данные из буфера еще не отправлены в программу.

2. Какие типы потоков существуют?

Потоки бывают

- Стандартные: только однонаправленные, либо входные, либо выходные.
- Строковые: могут быть и однонаправленными и двунаправленными

• Файловые: могут быть и однонаправленными и двунаправленными.

3. Какую библиотеку надо подключить при использовании стандартных потоков?

iostream

4. Какую библиотеку надо подключить при использовании файловых потоков?

fstream

5. Какую библиотеку надо подключить при использовании строковых потоков?

stream

6. Какая операция используется при выводе в форматированный поток?

Операция << используется при выводе в форматированный поток.

7. Какая операция используется при вводе из форматированных потоков?

Операция >> используется при вводе из форматированных потоков.

8. Какие методы используются при выводе в форматированный поток?

Прототип	Пример	
`ostream& put(char c)`	`char c='a'; stream.put(c);`	Записывает в поток stream символ c
`ostream& write(const char* buf, int size)`	`char c='a'; stream.write(&c);`	Записывает в поток stream символ c
`ostream& write(const char* buf, int size)`	`char s[]="string1"; stream.write(s,strlen (s));`	Записывает в поток stream строку символов

9. Какие методы используется при вводе из форматированного потока?

Прототип	Пример	
`int_type get(); istream& get(char &c)`	`c= stream.get(); stream.get(c)`	ввод из потока одного символа
`istream&read(char*buf,int size)`	`stream.read(&c,1)`	ввод из потока одного символа

10. Какие режимы для открытия файловых потоков существуют?

Режимы открытия потока

Режим	Описание
in	открыть поток для чтения (по умолчанию для ifstream)
out	(по умолчанию для ofstream)
trunk	удалить старое содержимое файла (по умолчанию для ofstream)
app	открыть поток для записи в конец файла
ate	открыть поток для чтения и/или записи и встать в конец файла
binary	открыть поток в двоичном режиме

Комбинации режимов

Режим	Описание
out|trunk	стирание и запись, если файла нет, то он создается
out|app	дозапись в файл, если файла нет, то он создается
in|out	чтение и запись, файл должен существовать
in|out|trunk	стирание, чтение и запись, если файла нет, то он создается

11. Какой режим используется для добавления записей в файл?

Режим ios::out используется для добавления записей в файл.

12. Какой режим (комбинация режимов) используется в конструкторе ifstream file(“f.txt”)?

ios::in

13. Какой режим (комбинация режимов) используется в конструкторе fstream file(“f.txt”)?

ios::in

ios::out

14. Какой режим (комбинация режимов) используется в конструкторе ofstream file(“f.txt”)?

ios::out

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

ios::out указывает на то, что файл будет открыт для записи, а ios::app гаранти-рует, что при каждой записи данные будут добавляться в конец файла. Если файл не существует, он будет создан. Если файл уже существует, то данные бу-дут добавляться в конец файла, не перезаписывая его содержимое.

16. Каким образом открывается поток в режиме ios::out |ios::trunc?

ios::out - файл будет открыт в режиме вывода и при этом, если файл уже суще-ствует, его содержимое будет удалено (ios::trunc). Если же файл не существует, он будет создан.

17. Каким образом открывается поток в режиме ios::out |ios::in|ios::trunk?

ios::out указывает на то, что файл будет использоваться для записи, ios::in ука-зывает на то, что файл будет использоваться для чтения, а ios::trunc указывает на то, что содержимое файла будет очищено перед записью в него.

18. Каким образом можно открыть файл для чтения?

ifstream file;

file.open("file.txt", ios::in);

19. Каким образом можно открыть файл для записи?

ifstream file;

file.open("file.txt", ios::out);

20. Привести примеры открытия файловых потоков в различных режимах.

ifstream file;

file.open("file.txt", ios::out | ios::app);

Добавляет новое содержимое в конец файла, не удаляя предыдущее содержимое

21. Привести примеры чтения объектов из потока.

```
class Person {
public:
    std::string name;
    int age;
    double salary;
};
int main() {
    Person person;
    ifstream file("persons.txt");
    while (file >> person.name >> person.age >> person.salary) {
        std::cout << "Name: " << person.name << ", Age: " << person.age << ", Salary: " << person.salary << std::endl;
    }
}
```

22. Привести примеры записи объектов в поток.

```cpp
class Person {
public:
    std::string name;
    int age;
    double salary;
};
int main() {
    Person person{"Ivan",30,100000.0};
    ofstream file("persons.txt");
    file << person.name << person.age << person.salary;
}
```

23. Сформулировать алгоритм удаления записей из файла.

1) открыть файл в режиме чтения

2) открыть вспомогательный файл в режиме записи

3) пока не конец файла, копировать записи из файла в вспомогательный, пропуская те, которые нужно удалить

4) удалить изначальный файл

5) переименовать вспомогательный файл в изначальный

24. Сформулировать алгоритм добавления записей в файл.

1) открыть файл в режиме чтения

2) открыть вспомогательный файл в режиме записи

3) пока не конец файла, копировать записи из файла в вспомогательный, если встретится запись, после которой нужно добавить новую, остановиться, начать добавлять новые, затем продолжить копирование старых записей из изначального файла

4) удалить изначальный файл

5) переименовать вспомогательный файл в изначальный

25. Сформулировать алгоритм изменения записей в файле.

1) открыть файл в режиме чтения

2) открыть вспомогательный файл в режиме записи

3) пока не конец файла, копировать записи из файла в вспомогательный, пропуская те, которые нужно удалить, если встретится запись, после которой нужно добавить новую, остановиться, начать добавлять новые, затем продолжить копирование старых записей из изначального файла

4) удалить изначальный файл

5) переименовать вспомогательный файл в изначальный