Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе № 18.10**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Сохранение данных в файле с использованием потоков

Вариант 21

Выполнил:

Студент группы РИС-20-1б

Ознобихин Елисей Андреевич

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

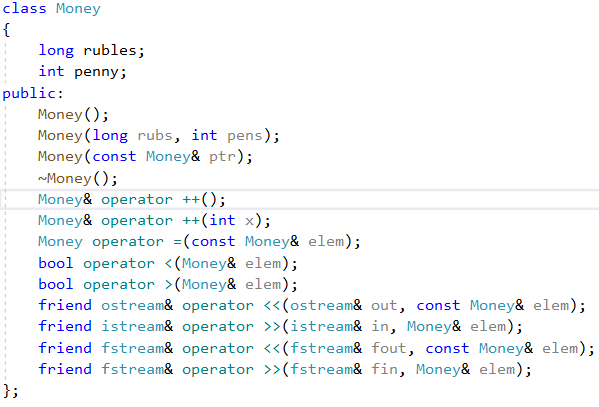
1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Разработка программы, в которой данные сохраняются в файле, корректируются и выводятся из файла на печать. Работа с файлом осуществляется с использованием потоковых классов.

**Постановка задачи**

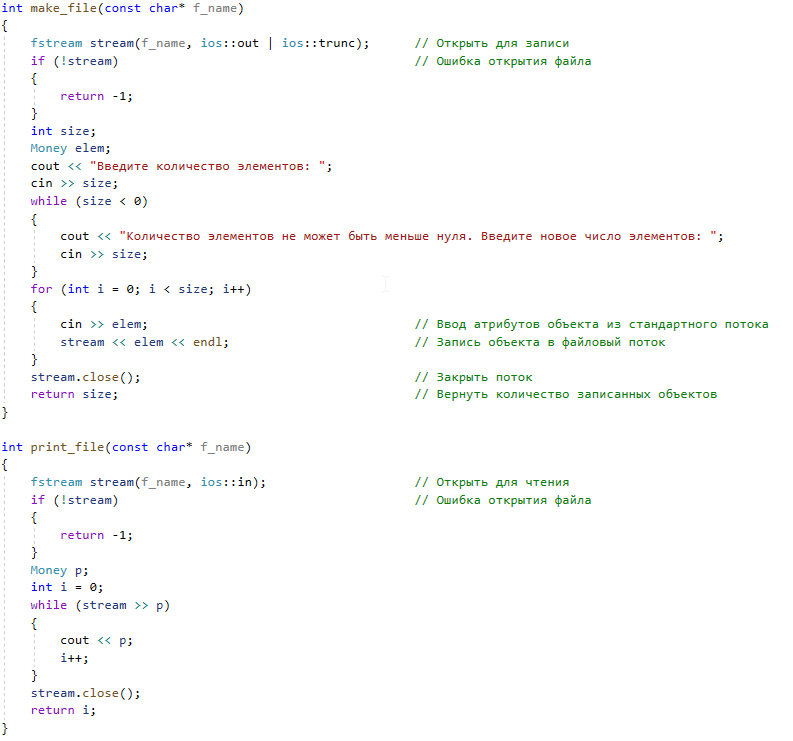
1. Создать пользовательский класс с минимальной функциональностью.
2. Написать функцию для создания объектов пользовательского класса (ввод исходной информации с клавиатуры) и сохранения их в потоке (файле).
3. Написать функцию для чтения и просмотра объектов из потока.
4. Написать функцию для удаления 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.

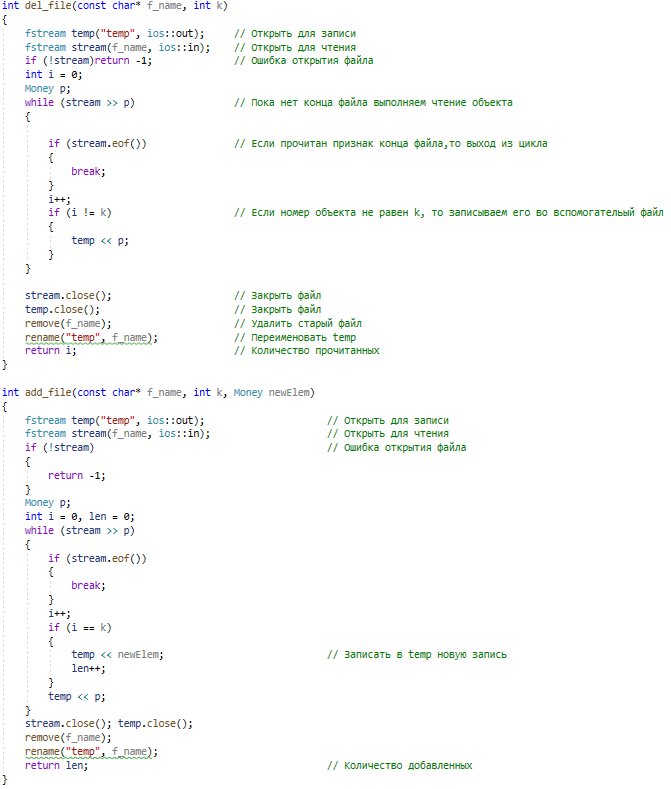
**Анализ задачи**

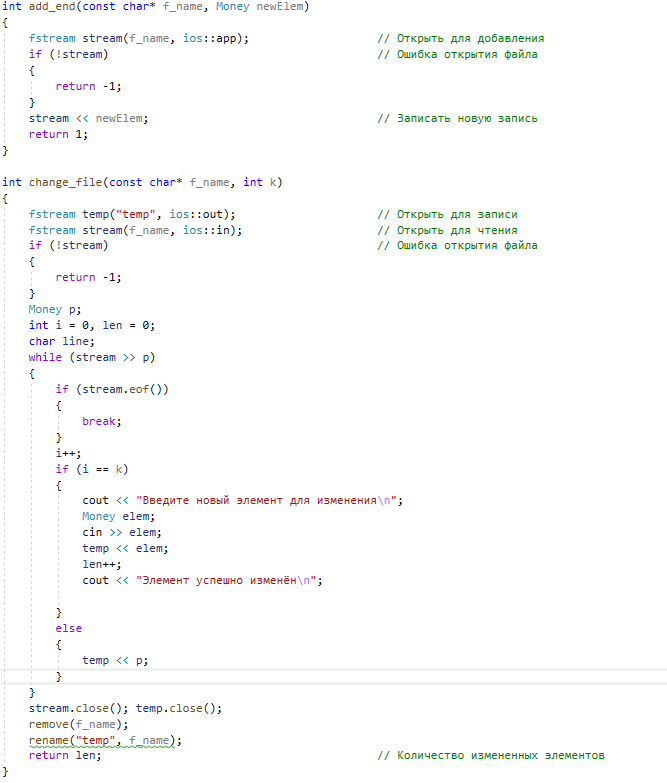
1. Для решения задачи необходимо:
   1. Реализовать определение абстрактного класса Money и определение необходимых методов данного класса.



* 1. Реализовать определение функций для работы с файловыми потоками и с самими файлами.

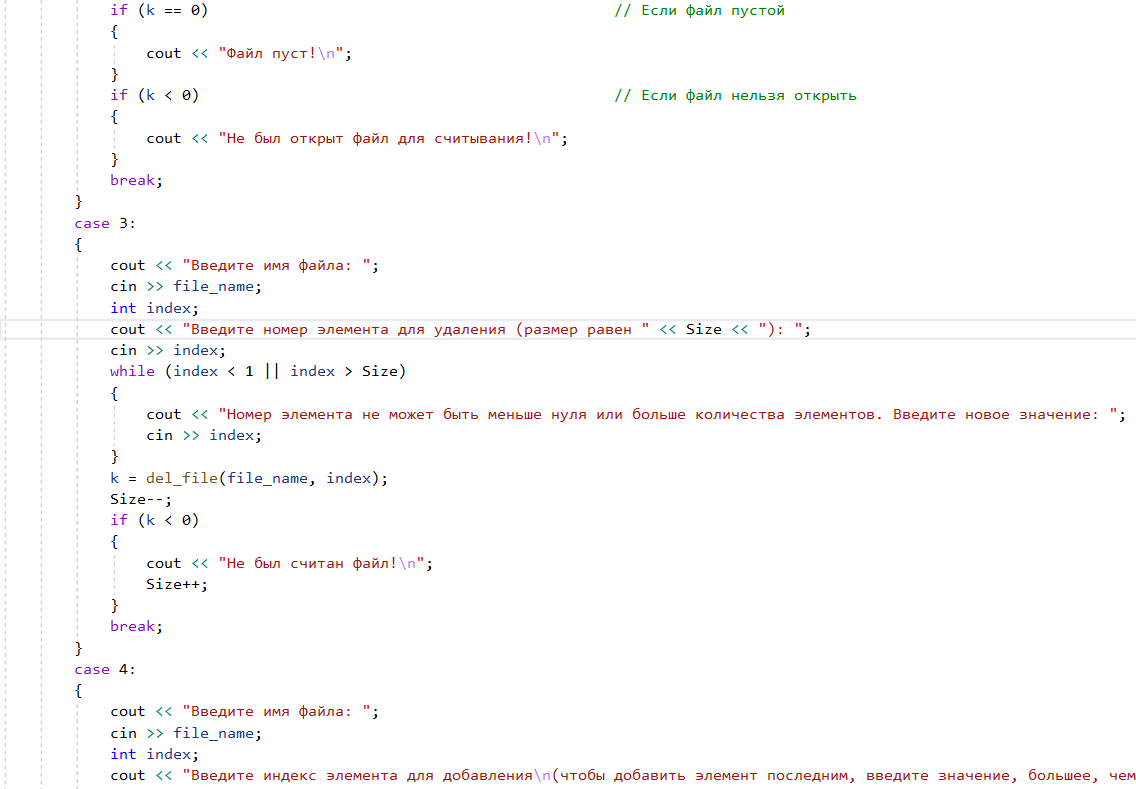




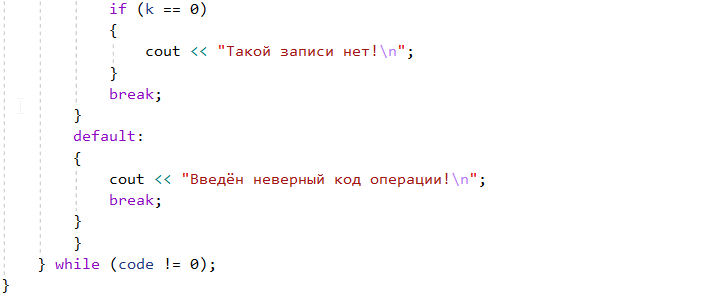


* 1. Реализовать применение этих функций в главной функции.

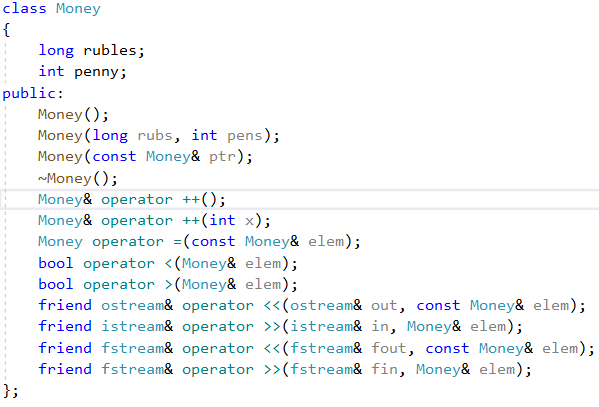








1. Для решения задачи данные были представлены в следующем виде:
   1. Для работы с данными используются атрибуты класса.



1. Для операций ввода и вывода использовались следующие операторы и функции:
   1. Ввод данных реализован с помощью оператора cin, используемых при реализации в главной функции и функций перегрузки оператора.

C:\Users\Асус\Documents\ShareX\Screenshots\2021-06\devenv_HTutM2i3FS.png

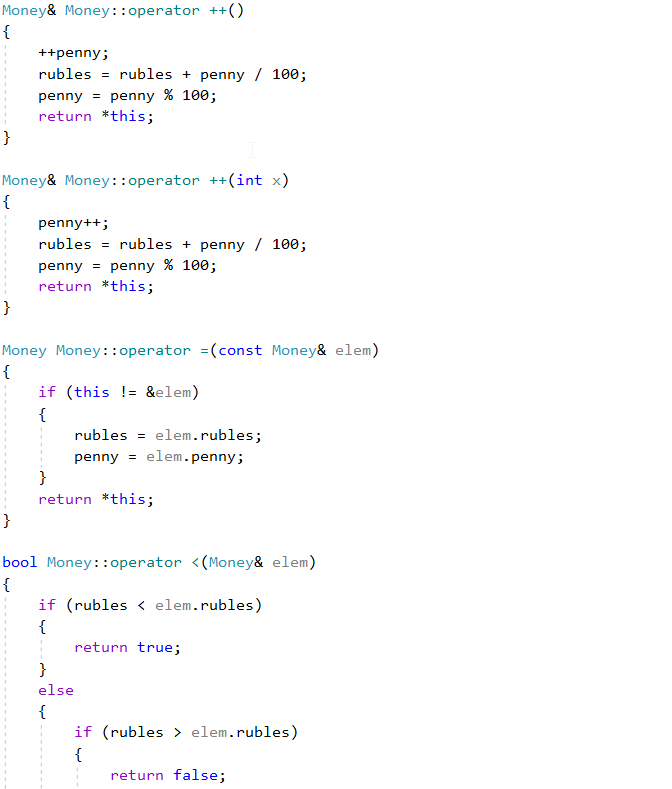
C:\Users\Асус\Documents\ShareX\Screenshots\2021-06\devenv_VcjfIsVS3l.png

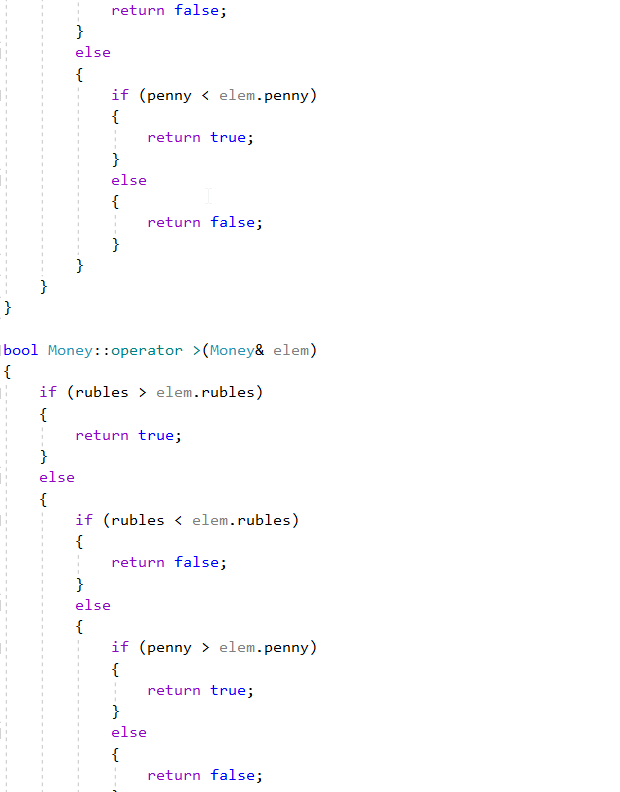
* 1. Вывод данных реализован с помощью оператора cout, используемого при реализации в главной функции и функций перегрузки операторов ввода-вывода.

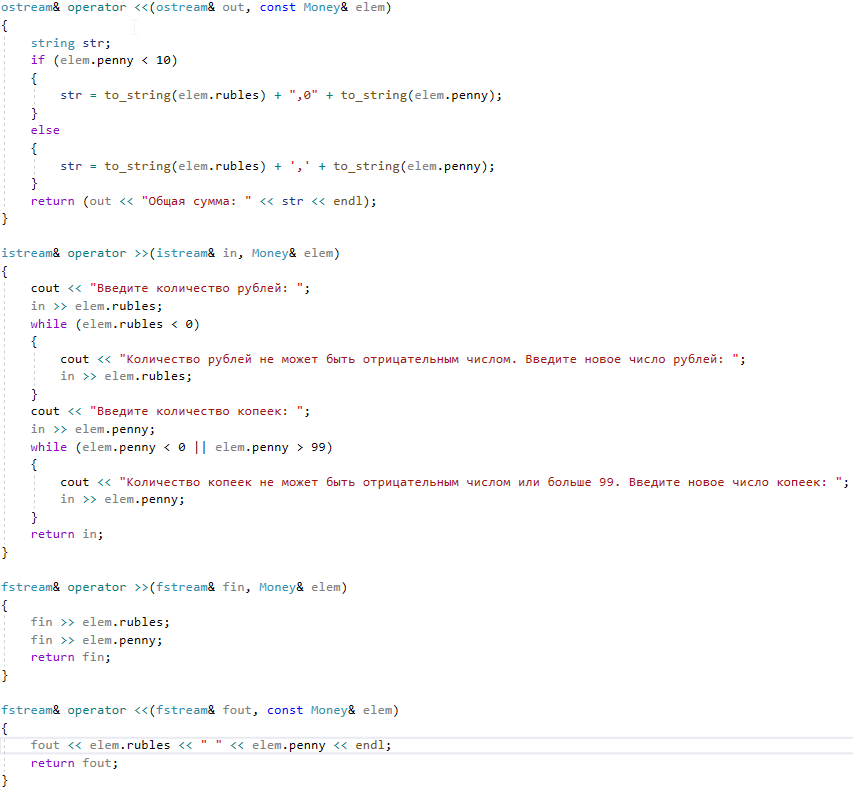
C:\Users\Асус\Documents\ShareX\Screenshots\2021-06\devenv_akGwsYiFv5.png

C:\Users\Асус\Documents\ShareX\Screenshots\2021-06\devenv_g7ZppTmaCP.png

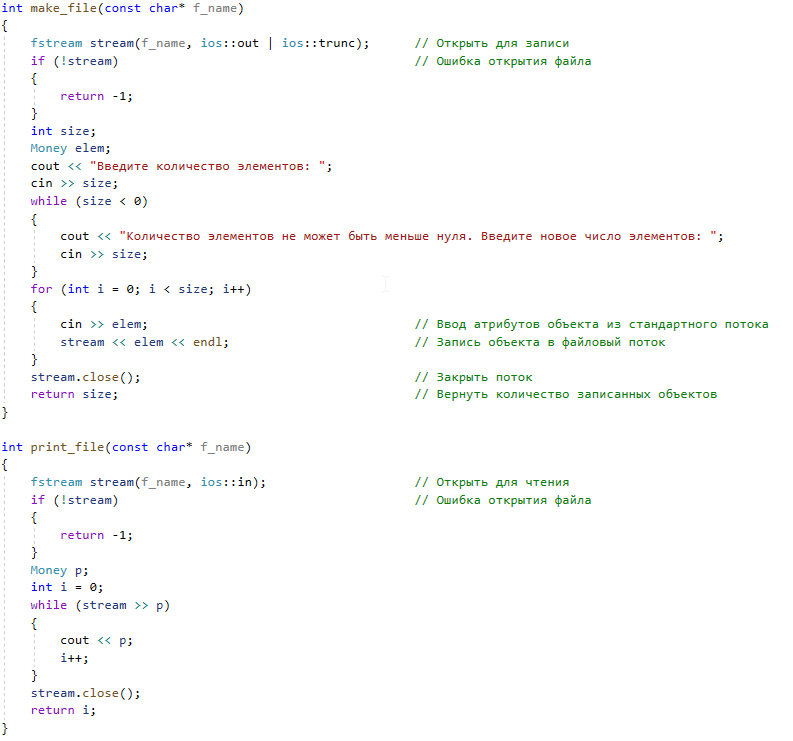
1. Поставленные задачи решены следующими действиями:
   1. В классе Money находится реализация методов для работы с деньгами: префиксный и постфиксный инкремент, операторы сравнения, файловые и консольные потоки ввода-вывода.

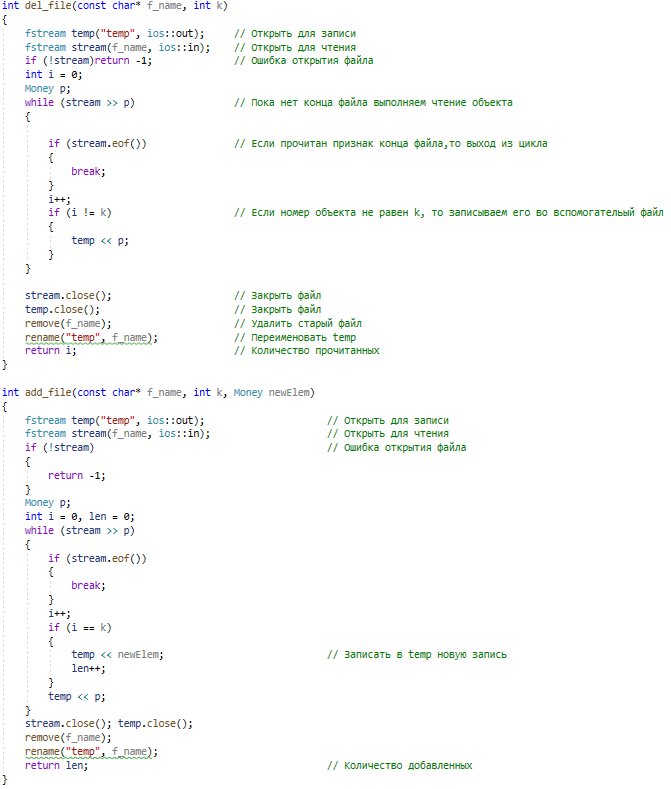


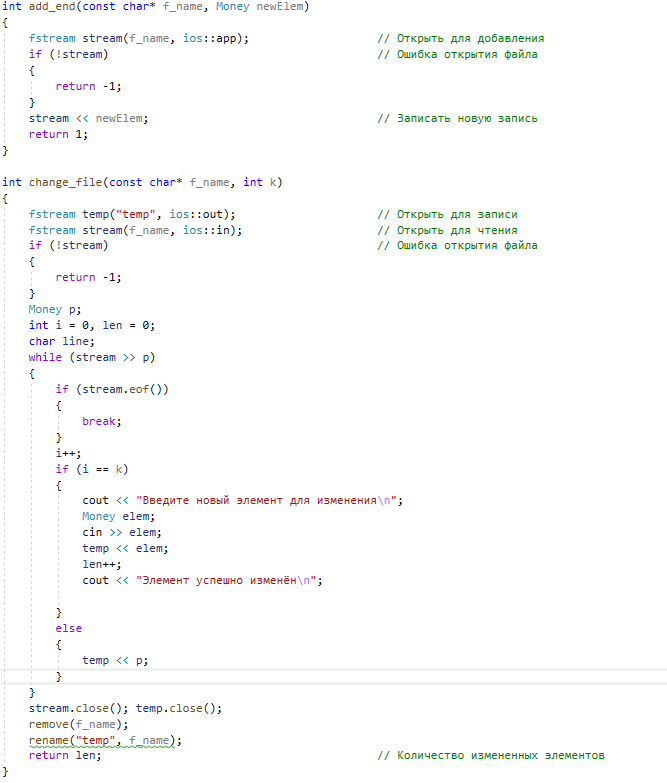




* 1. Реализованы следующие функции для работы с файлами: создание файла, редактирование файла (добавление, удаление или редактирование элемента), вывод содержимого файла.







**Реализация задачи на языке С++**

**Заголовочный файл FireWork.h**

#pragma once

#include "Money.h"

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

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

{

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

if (!stream)

{

return -1;

}

int size;

Money elem;

cout << "Введите количество элементов: ";

cin >> size;

while (size < 0)

{

cout << "Количество элементов не может быть меньше нуля. Введите новое число элементов: ";

cin >> size;

}

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

{

cin >> elem;

stream << elem << endl;

}

stream.close();

return size;

}

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

{

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

if (!stream)

{

return -1;

}

Money p;

int i = 0;

while (stream >> p)

{

cout << p;

i++;

}

stream.close();

return i;

}

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

{

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

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

if (!stream)return -1;

int i = 0;

Money p;

while (stream >> p)

{

if (stream.eof())

{

break;

}

i++;

if (i != k)

{

temp << p;

}

}

stream.close();

temp.close();

remove(f\_name);

rename("temp", f\_name);

return i;

}

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

{

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

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

if (!stream)

{

return -1;

}

Money p;

int i = 0, len = 0;

while (stream >> p)

{

if (stream.eof())

{

break;

}

i++;

if (i == k)

{

temp << newElem;

len++;

}

temp << p;

}

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

remove(f\_name);

rename("temp", f\_name);

return len;

}

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

{

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

if (!stream)

{

return -1;

}

stream << newElem;

return 1;

}

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

{

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

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

if (!stream)

{

return -1;

}

Money p;

int i = 0, len = 0;

char line;

while (stream >> p)

{

if (stream.eof())

{

break;

}

i++;

if (i == k)

{

cout << "Введите новый элемент для изменения\n";

Money elem;

cin >> elem;

temp << elem;

len++;

cout << "Элемент успешно изменён\n";

}

else

{

temp << p;

}

}

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

remove(f\_name);

rename("temp", f\_name);

return len;

}

**Заголовочный файл Money.h**

#pragma once

#include <iostream>

#include <fstream>

using namespace std;

class Money

{

long rubles;

int penny;

public:

Money();

Money(long rubs, int pens);

Money(const Money& ptr);

~Money();

Money& operator ++();

Money& operator ++(int x);

Money operator =(const Money& elem);

bool operator <(Money& elem);

bool operator >(Money& elem);

friend ostream& operator <<(ostream& out, const Money& elem);

friend istream& operator >>(istream& in, Money& elem);

friend fstream& operator <<(fstream& fout, const Money& elem);

friend fstream& operator >>(fstream& fin, Money& elem);

};

**Файл с описанием методов класса Money.cpp**

#include "Money.h"

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

Money::Money()

{

rubles = 0;

penny = 0;

}

Money::Money(long rubs, int pens)

{

rubles = rubs;

penny = pens;

}

Money::Money(const Money& ptr)

{

rubles = ptr.rubles;

penny = ptr.penny;

}

Money::~Money()

{

}

Money& Money::operator ++()

{

++penny;

rubles = rubles + penny / 100;

penny = penny % 100;

return \*this;

}

Money& Money::operator ++(int x)

{

penny++;

rubles = rubles + penny / 100;

penny = penny % 100;

return \*this;

}

Money Money::operator =(const Money& elem)

{

if (this != &elem)

{

rubles = elem.rubles;

penny = elem.penny;

}

return \*this;

}

bool Money::operator <(Money& elem)

{

if (rubles < elem.rubles)

{

return true;

}

else

{

if (rubles > elem.rubles)

{

return false;

}

else

{

if (penny < elem.penny)

{

return true;

}

else

{

return false;

}

}

}

}

bool Money::operator >(Money& elem)

{

if (rubles > elem.rubles)

{

return true;

}

else

{

if (rubles < elem.rubles)

{

return false;

}

else

{

if (penny > elem.penny)

{

return true;

}

else

{

return false;

}

}

}

}

ostream& operator <<(ostream& out, const Money& elem)

{

string str;

if (elem.penny < 10)

{

str = to\_string(elem.rubles) + ",0" + to\_string(elem.penny);

}

else

{

str = to\_string(elem.rubles) + ',' + to\_string(elem.penny);

}

return (out << "Общая сумма: " << str << endl);

}

istream& operator >>(istream& in, Money& elem)

{

cout << "Введите количество рублей: ";

in >> elem.rubles;

while (elem.rubles < 0)

{

cout << "Количество рублей не может быть отрицательным числом. Введите новое число рублей: ";

in >> elem.rubles;

}

cout << "Введите количество копеек: ";

in >> elem.penny;

while (elem.penny < 0 || elem.penny > 99)

{

cout << "Количество копеек не может быть отрицательным числом или больше 99. Введите новое число копеек: ";

in >> elem.penny;

}

return in;

}

fstream& operator >>(fstream& fin, Money& elem)

{

fin >> elem.rubles;

fin >> elem.penny;

return fin;

}

fstream& operator <<(fstream& fout, const Money& elem)

{

fout << elem.rubles << " " << elem.penny << endl;

return fout;

}

**Файл с главной программой Лабораторная работа № 18.10.cpp**

#include "Money.h"

#include "FileWork.h"

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int Size = 0;

int main()

{

system("chcp 1251>nul");

Money p;

int k, code;

char file\_name[30];

do

{

cout << "1. Создание файла\n";

cout << "2. Вывод данных из файла\n";

cout << "3. Удалить запись из файла\n";

cout << "4. Добавить запись в файл\n";

cout << "5. Изменить запись в файле\n";

cout << "0. Выход из программы\n";

cout << "Введите код операции: ";

cin >> code;

switch (code)

{

case 1:

{

cout << "Введите имя файла: ";

cin >> file\_name;

k = make\_file(file\_name);

Size = k;

if (k < 0)

{

cout << "Не был создан файл!\n";

}

break;

}

case 2:

{

cout << "Введите имя файла: ";

cin >> file\_name;

k = print\_file(file\_name);

Size = k;

if (k == 0)

{

cout << "Файл пуст!\n";

}

if (k < 0)

{

cout << "Не был открыт файл для считывания!\n";

}

break;

}

case 3:

{

cout << "Введите имя файла: ";

cin >> file\_name;

int index;

cout << "Введите номер элемента для удаления (размер равен " << Size << "): ";

cin >> index;

while (index < 1 || index > Size)

{

cout << "Номер элемента не может быть меньше нуля или больше количества элементов. Введите новое значение: ";

cin >> index;

}

k = del\_file(file\_name, index);

Size--;

if (k < 0)

{

cout << "Не был считан файл!\n";

Size++;

}

break;

}

case 4:

{

cout << "Введите имя файла: ";

cin >> file\_name;

int index;

cout << "Введите индекс элемента для добавления\n(чтобы добавить элемент последним, введите значение, большее, чем текущее количество элементов)\nНа данный момент количество элементов равно " << Size << "\nВведите индекс элемента для добавления: ";

cin >> index;

while (index < 1)

{

cout << "Номер элемента не может быть меньше нуля. Введите новое значение: ";

cin >> index;

}

Money elem;

cout << "Введите новый элемент для добавления\n";

cin >> elem;

k = add\_file(file\_name, index, elem);

Size++;

if (k == 0)

{

k = add\_end(file\_name, elem);

}

if (k < 0)

{

cout << "Не был считан файл!\n";

Size--;

}

break;

}

case 5:

{

cout << "Введите имя файла: ";

cin >> file\_name;

int index;

cout << "Введите номер элемента для изменения: ";

cin >> index;

k = change\_file(file\_name, index);

if (k < 0)

{

cout << "Не был считан файл!\n";

}

if (k == 0)

{

cout << "Такой записи нет!\n";

}

break;

}

default:

{

cout << "Введён неверный код операции!\n";

break;

}

}

} while (code != 0);

}

**Ответы на вопросы**

1. Последовательность байтов для обмена информацией с программой
2. Стандартный, строковый, файловый
3. Iostream
4. Fstream
5. Sstream
6. <<
7. >>
8. Write
9. Getline
10. In, out, trunk, app, ate, binary
11. App
12. In
13. In, out
14. Out, trunk
15. Добавление в конец
16. Запись в файл
17. Чтение и запись
18. Fstream stream(“name.txt”,ios::in);
19. Fstream stream(“name.txt”,ios::out);
20. Fstream stream(“name.txt”,ios::out|ios::app);
21. Stream>>p;
22. Stream<<p;
23. Записать во вспомогательный файл все, кроме удаляемого элемента. Удалить исходный документ и переименовать новый.
24. Открыть в режиме app и добавить
25. Все, до нужной записи, во вспомогательный файл. Записать измененный во вспомогательный файл, дописать оставшиеся записи из основного файла. Удалить исходный документ и переименовать новый.