Министерство образования и науки Российской Федерации

Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

**по лабораторной работе №18.10**

Дисциплина: «Информатика»

Тема: Объектно-ориентированное программирование. Потоки

Вариант 8

Выполнил:

Студент группы РИС-20-1б

Тараканов Д. М.

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

Пермь

2021 год

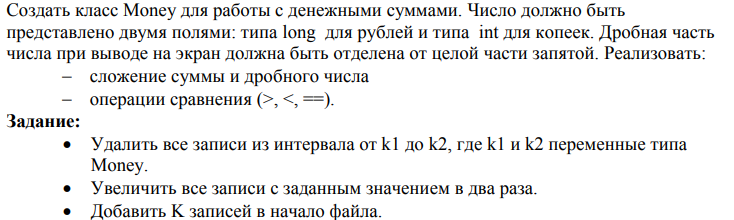
# **Цель работы**

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 чтения и просмотра объектов из потока.
5. Написать функции взаимодействия с потоками (файлами) в зависимости от варианта.
6. Для вызова функций предусмотреть меню.
7. Написать демонстрационную программу, иллюстрирующую выполнение задачи.

(8 вариант)



# **Анализ задачи**

1. Определить какие действия предстоит выполнить:

* Разработать класс Money который хранит в себе значения рублей и копеек.
* Организовать методы для ввода данных, селекторы и модификатор, конструкторы и деструкторы, перегрузки операторов.

class Money {

long int rubles;

int penny;

public:

Money();

Money(long int, int);

Money(const Money&);

Money& operator = (const Money&);

int get\_rubles();

int get\_penny();

void set\_rubles(long int);

void set\_penny(int);

bool operator == (const Money&);

bool operator > (const Money&);

bool operator < (const Money&);

Money& operator + (int);

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

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

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

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

~Money();

};

* Разработать функции взаимодействия с потоком в файле file\_work.

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

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

if (!stream) {

return -1;

}

Money M;

int quantity;

cout << "\nВведите кол-во сумм:\n> "; cin >> quantity;;

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

cout << "\n" << i + 1 << ")";

cin >> M;

stream << M << "\n";

}

stream.close();

return quantity;

}

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

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

if (!stream) {

return -1;

}

Money M;

int quantity = 0;

cout << "\n~~~~~~~~~~~~~~~~~\nСодержимое файла:\n~~~~~~~~~~~~~~~~~\n";

while (stream >> M) {

cout << "\n" << M << "\n";

quantity++;

}

stream.close();

return quantity;

}

int delete\_range(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M, k1, k2;

cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~\nВведите диапозон [k1, k2]\n~~~~~~~~~~~~~~~~~~~~~~~~~\n";

cout << "\nk1?"; cin >> k1; cout << "\nk2?"; cin >> k2;

if (k1 > k2) {

M = k1;

k1 = k2;

k2 = M;

}

int quantity = 0;

while (stream >> M) {

if (stream.eof()) {

break;

}

if (M < k1 || M > k2) {

temp << M << "\n";

}

quantity++;

}

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

remove(file\_name);

rename("temp", file\_name);

return quantity;

}

int add\_begin(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M;

int K;

cout << "\nВведите кол-во новых сумм:\n> "; cin >> K;

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

cout << "\n" << i+1 << ")";

cin >> M;

temp << M << "\n";

}

while (stream >> M) {

temp << M << "\n";

}

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

remove(file\_name);

rename("temp", file\_name);

return K;

}

int change\_file(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M, Const;

cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\nКакую запись в файле изменить?\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"; cin >> Const;

long int rubles, x;

int penny, quantity = 0;

rubles = Const.get\_rubles();

penny = Const.get\_penny();

x = rubles \* 100 + penny;

while (stream >> M) {

if (M == Const) {

M + x;

quantity++;

}

temp << M << "\n";

}

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

remove(file\_name);

rename("temp", file\_name);

return quantity;

}

* Создать меню в функции main и протестировать работу программы.

void main() {

system("chcp 1251");

int op, quantity;

char file\_name[30];

do {

cout << "\n===========================\n1 - Создать файл\n2 - Прочитать файл\n3 - Удалить записи из файла\n4 - Добаить записи в файл\n5 - Изменить запись в файле\n0 - Выход\n===========================\n\nВаша выбор > ";

cin >> op;

switch (op) {

case 1:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = make\_file(file\_name);

if (quantity < 0) {

cout << "\nНевозможно создать файл!\n";

}

break;

case 2:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = print\_file(file\_name);

if (quantity == 0) {

cout << "\nФайл пуст!\n";

}

if (quantity < 0) {

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

}

break;

case 3:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = delete\_range(file\_name);

if (quantity < 0) {

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

}

break;

case 4:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = add\_begin(file\_name);

if (quantity < 0) {

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

}

break;

case 5:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = change\_file(file\_name);

if (quantity < 0) {

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

}

if (quantity == 0) {

cout << "\nЗапись в файле не найдена!\n";

}

if (quantity > 0) {

cout << "\nКол-во измененых записей = " << quantity << endl;

}

break;

}

} while (op != 0);

system("pause");

}

1. C какими типами данных предстояло работать:

* Для хранения кол-ва рублей реализована переменная rubles типа long int.

long int rubles;

* Для хранения кол-ва копее реализована переменная penny типа int.

int penny;

* Для выбора метода взаимодействия с потоком и управлением меню реализована переменная op типа int.

int op;

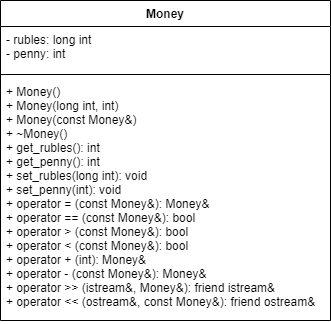
* Для обращения к потоку реализована переменная file\_name типа char[30].

char file\_name[30];

* Для подсчета записей фала реализована переменная quantity типа int.

int quantity;

# **UML диаграмма**



# **Код программы на языке C++**

Заголовочный файл file\_work.h

#pragma once

#include <iostream>

#include "Money.h"

#include <fstream>

#include <string>

using namespace std;

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

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

if (!stream) {

return -1;

}

Money M;

int quantity;

cout << "\nВведите кол-во сумм:\n> "; cin >> quantity;;

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

cout << "\n" << i + 1 << ")";

cin >> M;

stream << M << "\n";

}

stream.close();

return quantity;

}

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

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

if (!stream) {

return -1;

}

Money M;

int quantity = 0;

cout << "\n~~~~~~~~~~~~~~~~~\nСодержимое файла:\n~~~~~~~~~~~~~~~~~\n";

while (stream >> M) {

cout << "\n" << M << "\n";

quantity++;

}

stream.close();

return quantity;

}

int delete\_range(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M, k1, k2;

cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~\nВведите диапозон [k1, k2]\n~~~~~~~~~~~~~~~~~~~~~~~~~\n";

cout << "\nk1?"; cin >> k1; cout << "\nk2?"; cin >> k2;

if (k1 > k2) {

M = k1;

k1 = k2;

k2 = M;

}

int quantity = 0;

while (stream >> M) {

if (stream.eof()) {

break;

}

if (M < k1 || M > k2) {

temp << M << "\n";

}

quantity++;

}

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

remove(file\_name);

rename("temp", file\_name);

return quantity;

}

int add\_begin(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M;

int K;

cout << "\nВведите кол-во новых сумм:\n> "; cin >> K;

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

cout << "\n" << i+1 << ")";

cin >> M;

temp << M << "\n";

}

while (stream >> M) {

temp << M << "\n";

}

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

remove(file\_name);

rename("temp", file\_name);

return K;

}

int change\_file(const char\* file\_name) {

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

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

if (!stream) {

return -1;

}

Money M, Const;

cout << "\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\nКакую запись в файле изменить?\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"; cin >> Const;

long int rubles, x;

int penny, quantity = 0;

rubles = Const.get\_rubles();

penny = Const.get\_penny();

x = rubles \* 100 + penny;

while (stream >> M) {

if (M == Const) {

M + x;

quantity++;

}

temp << M << "\n";

}

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

remove(file\_name);

rename("temp", file\_name);

return quantity;

}

Заголовочный файл Money.h

#pragma once

#include <iostream> //Библиотека для работы с стандартными потоками

#include <fstream> //Библиотека для работы с файловыми потоками

#include <string>

using namespace std;

class Money {

long int rubles;

int penny;

public:

Money();

Money(long int, int);

Money(const Money&);

Money& operator = (const Money&);

int get\_rubles();

int get\_penny();

void set\_rubles(long int);

void set\_penny(int);

bool operator == (const Money&);

bool operator > (const Money&);

bool operator < (const Money&);

Money& operator + (int);

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

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

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

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

~Money();

};

Обьявление класса в Money.cpp

#include "Money.h"

Money::Money() {

rubles = 0;

penny = 0;

}

Money::Money(long int r, int p) {

rubles = r;

penny = p;

}

Money::Money(const Money& temp\_Money) {

rubles = temp\_Money.rubles;

penny = temp\_Money.penny;

}

Money& Money::operator=(const Money& temp\_Money) {

if (\*this == temp\_Money) {

return \*this;

}

rubles = temp\_Money.rubles;

penny = temp\_Money.penny;

return \*this;

}

int Money::get\_rubles() {

return rubles;

}

void Money::set\_rubles(long int r) {

rubles = r;

}

int Money::get\_penny() {

return penny;

}

void Money::set\_penny(int p) {

penny = p;

}

bool Money::operator==(const Money& temp\_Money) {

return (rubles == temp\_Money.rubles && penny == temp\_Money.penny);

}

bool Money::operator>(const Money& temp\_Money) {

return (rubles \* 100 + penny > temp\_Money.rubles \* 100 + temp\_Money.penny);

}

bool Money::operator<(const Money& temp\_Money) {

return (rubles \* 100 + penny < temp\_Money.rubles \* 100 + temp\_Money.penny);

}

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

long int Money = rubles \* 100 + penny;

Money += x;

rubles = Money / 100;

penny = Money % 100;

return \*this;

}

Money::~Money() { }

istream& operator>>(istream& in, Money& temp\_Money) {

do {

cout << "\nРубли > "; in >> temp\_Money.rubles;

} while (temp\_Money.rubles < 0);

do {

cout << "Копейки > "; in >> temp\_Money.penny;

} while (temp\_Money.penny > 99 || temp\_Money.penny < 0);

return in;

}

ostream& operator<<(ostream& out, const Money& temp\_Money) {

if (temp\_Money.penny < 10) {

out << temp\_Money.rubles << ",0" << temp\_Money.penny;

}

else {

out << temp\_Money.rubles << "," << temp\_Money.penny;

}

return out;

}

fstream& operator>>(fstream& fin, Money& temp\_Money) {

string str;

getline(fin, str, ',');

if (str.length() > 0) {

temp\_Money.rubles = stoi(str);

}

getline(fin, str);

if (str.length() > 0) {

temp\_Money.penny = stoi(str);

}

return fin;

}

fstream& operator<<(fstream& fout, const Money& temp\_Money) {

if (temp\_Money.penny < 10) {

fout << temp\_Money.rubles << ",0" << temp\_Money.penny;

}

else {

fout << temp\_Money.rubles << "," << temp\_Money.penny;

}

return fout;

}

return \*this;

}

Main файл

#include <iostream>

#include "Money.h"

#include "file\_work.h"

#include <fstream>

#include <string>

using namespace std;

void main() {

system("chcp 1251");

int op, quantity;

char file\_name[30];

do {

cout << "\n===========================\n1 - Создать файл\n2 - Прочитать файл\n3 - Удалить записи из файла\n4 - Добаить записи в файл\n5 - Изменить запись в файле\n0 - Выход\n===========================\n\nВаша выбор > ";

cin >> op;

switch (op) {

case 1:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = make\_file(file\_name);

if (quantity < 0) {

cout << "\nНевозможно создать файл!\n";

}

break;

case 2:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = print\_file(file\_name);

if (quantity == 0) {

cout << "\nФайл пуст!\n";

}

if (quantity < 0) {

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

}

break;

case 3:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = delete\_range(file\_name);

if (quantity < 0) {

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

}

break;

case 4:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = add\_begin(file\_name);

if (quantity < 0) {

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

}

break;

case 5:

cout << "\nИмя файла > "; cin >> file\_name;

quantity = change\_file(file\_name);

if (quantity < 0) {

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

}

if (quantity == 0) {

cout << "\nЗапись в файле не найдена!\n";

}

if (quantity > 0) {

cout << "\nКол-во измененых записей = " << quantity << endl;

}

break;

}

} while (op != 0);

system("pause");

}

# **Скриншоты тестов**

