Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

Лабораторная работа №10

Сохранение данных в файле с использованием потоков

Выполнил

Студент группы РИС-22-2б

Вековшинин Д. А.

Проверила

Доцент кафедры ИТАС

Полякова О. А.

Пермь 2023

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.

Вариант 10: Создать класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой. Реализовать:

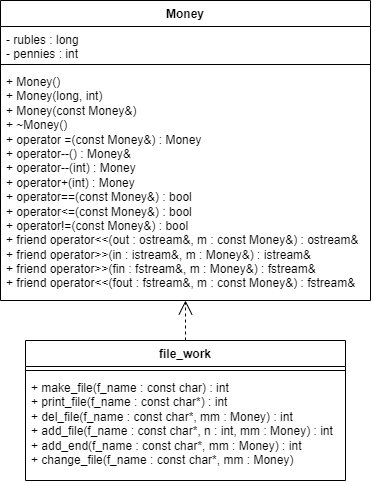
- операции сравнения (==, !=).

- вычитание копеек (--) (постфиксная и префиксная формы)

Задание:

* Удалить все записи большие заданного значения.
* Увеличить все записи с заданным значением на 1 рубль 50 копеек.
* Добавить K записей после записи с номером N

UML-диаграмма



Описание пользовательского класса.

class Money

{

public:

Money();

Money(long, int);

Money(const Money&);

~Money();

Money operator =(const Money&);

Money& operator--();

Money operator--(int);

Money operator+(int);

bool operator==(const Money&);

bool operator<=(const Money&);

bool operator!=(const Money&);

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

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

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

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

private:

long rubles;

int pennies;

};

Перегруженные операции пользовательского класса.

Money& Money::operator--()

{

int temp = rubles \* 100 + pennies;

temp--;

rubles = temp / 100;

pennies = temp % 100;

return \*this;

}

Money Money::operator+(int p)

{

int temp = rubles \* 100 + pennies;

temp += p;

rubles = temp / 100;

pennies = temp % 100;

return \*this;

}

Money Money::operator --(int)

{

int temp = rubles \* 100 + pennies;

temp--;

Money t(rubles, pennies);

rubles = temp / 100;

pennies = temp % 100;

return t;

}

bool Money::operator==(const Money& m)

{

int temp1 = rubles \* 100 + pennies;

int temp2 = m.rubles \* 100 + m.pennies;

bool compare = (temp1 == temp2);

return compare;

}

bool Money::operator<=(const Money& m)

{

int temp1 = rubles \* 100 + pennies;

int temp2 = m.rubles \* 100 + m.pennies;

bool compare = (temp1 < temp2);

return compare;

}

bool Money::operator!=(const Money& m)

{

int temp1 = rubles \* 100 + pennies;

int temp2 = m.rubles \* 100 + m.pennies;

bool compare = (temp1 != temp2);

return compare;

}

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

{

out << m.rubles << ',' << m.pennies << "\n";

return out;

}

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

{

cout << "rubles? "; in >> m.rubles;

cout << "pennies? "; in >> m.pennies;

return in;

}

//дружественные функции для работы с файловыми потоками

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

{

fin >> m.rubles;

fin >> m.pennies;

return fin;

}

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

{

fout << m.rubles << "\n" << m.pennies << "\n";

return fout;

}

Определение функций для работы с файлом (создание, вывод, удаление, добавление, изменение).

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

{

fstream stream(f\_name, ios::out | ios::trunc);//открыть для записи

if (!stream)return -1;//ошибка открытия файла

int n;

Money m;

cout << "N? "; cin >> n;

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

{

cin >> m;//ввод атрибутов объекта из стандартного потока

stream << m << "\n";//запись объекта в файловый поток

}

stream.close();//закрыть поток

return n;//вернуть количество записанных объектов

}

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

{

fstream stream(f\_name, ios::in);//открыть для чтения

if (!stream)return -1;//ошибка открытия файла

Money m; int i = 0;

while (stream >> m)

{

cout << m << "\n";

i++;

}

stream.close();

return i;

}

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

{

fstream temp("temp", ios::out);//открыть для записи

fstream stream(f\_name, ios::in);//открыть для чтения

if (!stream)return -1;//ошибка открытия файла

int i = 0;

Money m;

while (stream >> m)//пока нет конца файла выполняем чтение объекта

{

//если прочитан признак конца файла,то выход из цикла

if (stream.eof())break;

i++;

//если номер объекта не равен k, то записываем его во вспомогательый файл

if (m <= mm) temp << m;

}

//закрыть файлы

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

remove(f\_name);//удалить старый файл

rename("temp", f\_name);// переименовать temp

return i;//количество прочитанных

}

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

{

fstream temp("temp", ios::out);//открыть для записи

fstream stream(f\_name, ios::in);//открыть для чтения

if (!stream)return -1;//ошибка открытия файла

Money m; int i = 0, l = 0;

while (stream >> m)

{

if (stream.eof())break;

i++;

if (i == n)

{

temp << mm;//записать в temp новую запись

l++;

}

temp << m;

}

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

remove(f\_name);

rename("temp", f\_name);

return l;//количество добавленных

}

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

{

fstream stream(f\_name, ios::app);//открыть для добавления

if (!stream)return -1;//ошибка открытия файла

stream << mm; //записать новую запись

return 1;

}

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

{

fstream temp("temp", ios::out);//открыть для записи

fstream stream(f\_name, ios::in);//открыть для чтения

if (!stream)return -1;//ошибка открытия файла

Money m; int i = 0, l = 0;

char x;

while (stream >> m)

{

if (stream.eof())break;

i++;

if (m == mm)

{

temp << m+150;

l++;

}

else temp << m;

}

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

remove(f\_name);

rename("temp", f\_name);

return l;//количество измененных элементов

}

Функция main()

void main()

{

Money m, m1, m2, toDelete;

int n, k, c;

char file\_name[30];

do

{ //Меню

cout << "\n1. Make file";

cout << "\n2. Print file";

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

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

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

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

cin >> c;

switch (c)

{

case 1: cout << "file name? "; cin >> file\_name;

k = make\_file(file\_name);

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

break;

case 2: cout << "file name? "; cin >> file\_name;

cout << endl;

k = print\_file(file\_name);

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

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

break;

case 3: cout << "file name? "; cin >> file\_name;

int nom;

cout << "value? "; cin >> toDelete;

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

if (k < 0)cout << "Can't read file";

break;

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

int numOfValues;

cin >> file\_name;

cout << "after what nom? "; cin >> nom;

cout << "number of values? "; cin >> numOfValues;

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

{

cout << "New value: ";

cin >> m1;

k = add\_file(file\_name, nom+i+1, m1);

}

if (k < 0) cout << "Can't read file";

if (k == 0)

{

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

k = add\_end(file\_name, m1);

}

break;

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

cin >> file\_name;

cout << "value: ";

cin >> m2;

k = change\_file(file\_name, m2);

if (k < 0) cout << "\nCan't read file";

if (k == 0) cout << "\nNot such record";

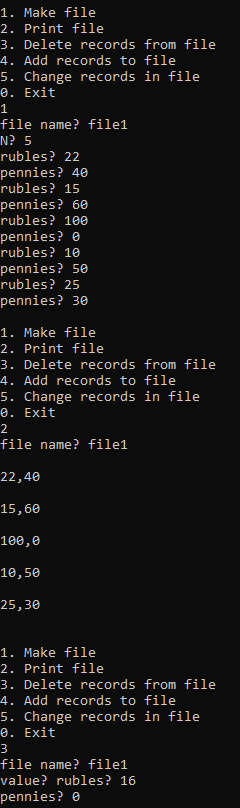
break;

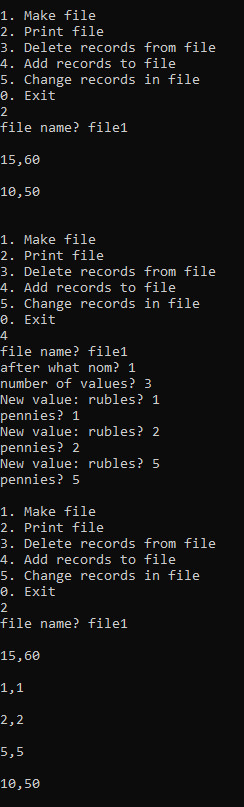
}

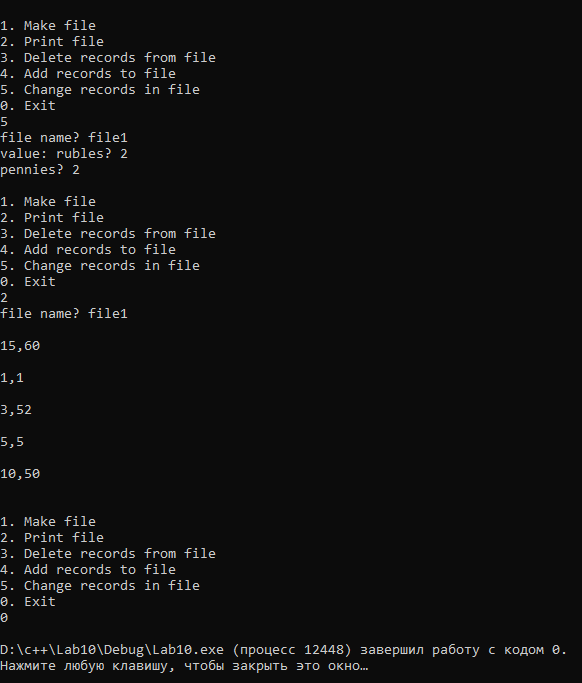
} while (c != 0);

}

Результаты работы программы







Ответы на контрольные вопросы

1. Что такое поток?

Поток - последовательность байтов

1. Какие типы потоков существуют?

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

1. Какую библиотеку надо подключить при использовании стандартных потоков?

iostream

#include <iostream>

1. Какую библиотеку надо подключить при использовании файловых потоков?

fstream

#include <fstream>

1. Какую библиотеку надо подключить при использовании строковых потоков?

sstream

#include <sstream>

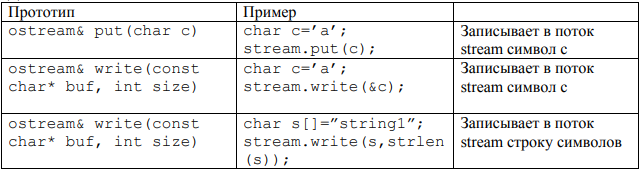
1. Какая операция используется при выводе в форматированный поток?

Перегруженная операция сдвига влево operator<<

1. Какая операция используется при вводе из форматированных потоков?

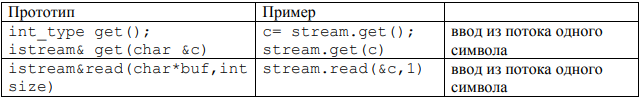
Перегруженная операция сдвига вправо operator>>

1. Какие методы используются при выводе в форматированный поток?

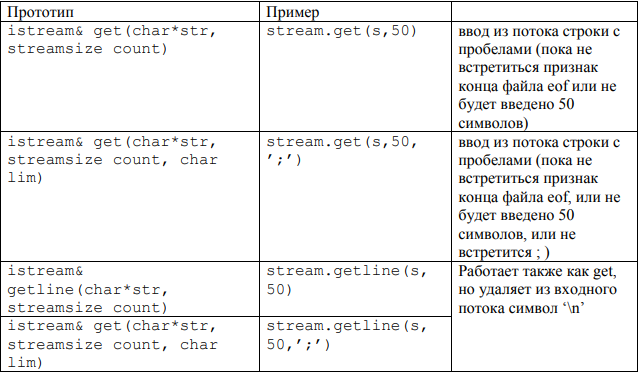


1. Какие методы используется при вводе из форматированного потока?

Поместить в символьную переменную любой символ (в том числе и символ разделитель) можно с помощью методов

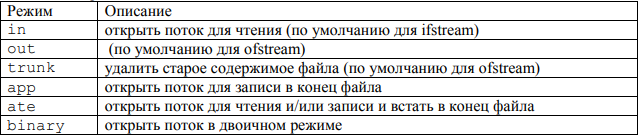


Ввод символьных массивов и строк выполняется с помощью операции >> до первого символа-разделителя (обычно пробела). Для ввод строк с пробелами используют методы get() и getline().

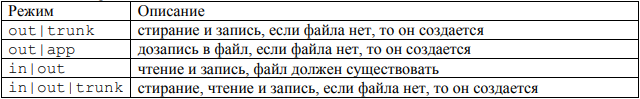


1. Какие режимы для открытия файловых потоков существуют?

Режимы открытия потока



Комбинации режимов



1. Какой режим используется для добавления записей в файл?

out

fstream temp("temp", ios::out);//открыть для записи

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

in

1. Какой режим (комбинация режимов) используется в конструкторе

fstream file(“f.txt”)?

in | out

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

out

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

Дозапись в файл, если файла нет, то он создается

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

Стирание и запись, если файла нет, то он создается

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

Стирание, чтение и запись, если файла нет, то он создается

1. Каким образом можно открыть файл для чтения?

При помощи режима in

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

1. Каким образом можно открыть файл для записи?

При помощи режима out

fstream temp("temp", ios::out);//открыть для записи

1. Привести примеры открытия файловых потоков в различных режимах.

fstream temp("temp", ios::out);//открыть для записи

fstream stream(f\_name, ios::in);//открыть для чтения

fstream stream(f\_name, ios::app);//открыть для дозаписи

1. Привести примеры чтения объектов из потока.

int a, b;

stream>>a>>b;

22.Привести примеры записи объектов в поток.

int a, b;

stream<<a<<b<<’\n’;

1. Сформулировать алгоритм удаления записей из файла.
2. Открыть для записи вспомогательный файл
3. Открыть для чтения основной файл
4. Выполнять чтение объекта из файла, пока не достигнут конец файла и проверять каждую запись на соответствие критерию удаления.

4) Записать не подходящие под критерий записи во временный файл.

5) Закрыть исходный файл и временный файл.

6) Удалить исходный файл с помощью функции.

7) Переименовать временный файл в имя исходного файла.

1. Сформулировать алгоритм добавления записей в файл.
2. Открыть для записи вспомогательный файл
3. Открыть для чтения основной файл
4. Создать объект записи, который будет содержать данные для записи в файл.
5. Заполнить вспомогательный файл записями из основного файла

5) Записать объект записи для добавления в файл

6) Закрыть исходный файл и временный файл.

7) Удалить исходный файл с помощью функции.

8) Переименовать временный файл в имя исходного файла.

1. Сформулировать алгоритм изменения записей в файле.
2. Открыть для записи вспомогательный файл
3. Открыть для чтения основной файл
4. Создать объект записи, который будет содержать данные для записи в файл.
5. Заполнить вспомогательный файл записями из основного файла и заменить нужные записи

5) Закрыть исходный файл и временный файл.

6) Удалить исходный файл с помощью функции.

7) Переименовать временный файл в имя исходного файла.