

ФГАОУ ВО «Санкт-Петербургский политехнический университет Петра Великого» Институт компьютерных наук и технологий

Высшая школа киберфизических систем и управления

по дисциплине Практикум по программированию

Выполнил студент группы 23533/2

Руководитель

ассистент

В.Э. Ковалевский

Санкт-Петербург 2019

30.05.2019

А.А. Тарасов

Отчет по лабораторной работе

Двоичные файлы и сериализация

Двоичные файлы

Вообще, любой файл в машине, работающей с двоичным кодом, т.е. машиной, которая на аппаратном уровне может обрабатывать только двоичные числа, тоже является двоичным, представляя собой последовательность битов, интерпретируемых как целое позже операционной системой или некоторой программой.

Текстовые файлы являются частным случаем бинарного файла. Информация, хранимая в серии битов, интерпретируется системой как набор, например, однобайтных символов. В результате каждые N бит переводятся в какой-то символ. Конечно, размер символа в байтах будет зависеть от кодировки. В ASCII – 1 байт, в UTF-8 – от 1 до 4 байт.

Отличие двоичного файла от текстового в том, что данные в нём представляются во внутренней форме – в двоичной системе счисления.

В С и С++ могут работать с памятью на низком уровне, а значит на этих языках без дополнительных библиотек можно писать программы для работы с двоичными файлами.

Содержимое двоичных файлов удобно проверять в HEX-редакторе, который представит каждый байт файла как число в шестнадцатеричной системе и, если возможно, интерпретирует его как символ.

Сериализация

Сериализация – это процесс перевода каких-то структур данных в последовательность битов. Сериализацию используют, чтобы передавать данные по сети, или чтобы компактно упаковать их в файлы. Чтобы распаковать переданные или сохранённые данные, необходимо провести десериализацию.

Была разработана программа, в случайном порядке расставляющая студентов в очередь. Запишем результат в двоичный файл. Будет использована библиотека Serialization из Boost.

Студенты будут отсортированы, данные записаны в двоичный файл, а затем выгружены из него для демонстрации возможностей сериализации в boost.

Для сериализации данных необходимо подключить заголовочные файлы, которые реализуют т.н архив. Данные, подлежащие сериализации должны быть переданы в этот архив. Для архива перегружен << или & что позволяет работать с ним как с любым другим потоком данных. & предпочтительнее.

#include "fstream"

#include "boost/archive/text\_oarchive.hpp"

#include "boost/archive/text\_iarchive.hpp"

Для встроенных типов данных данные записываются в архив напрямую. Если тип пользовательский – класс/структура, то для объекта вызывается функция serialize. Она последовательно записывает всё содержимое класса в архив. Если класс содержит ещё класс, процесс рекурсивно продолжится.

Код, в котором данные из одного массива выгружаются в файл, затем из файла в другой массив, и выводятся на экран.

//Запись в файл

ofstream file("output.txt", ios\_base::out);

boost::archive::text\_oarchive oa(file);

for (int i = 0; i < 19; i++)

oa << students[i];

file.close();

//Создание чистого массива студентов

student students2[19];

//Чтение из файла

ifstream file2("output.txt");

boost::archive::text\_iarchive ia(file2);

for (int i = 0; i < 19; i++)

ia >> students2[i];

for(int i=1; i<20; i++)

cout << i << ") " << students2[i-1].name <<", счастливое число - "<< students2[i-1].randomnumber << endl;

Демонстрация списка на экране:

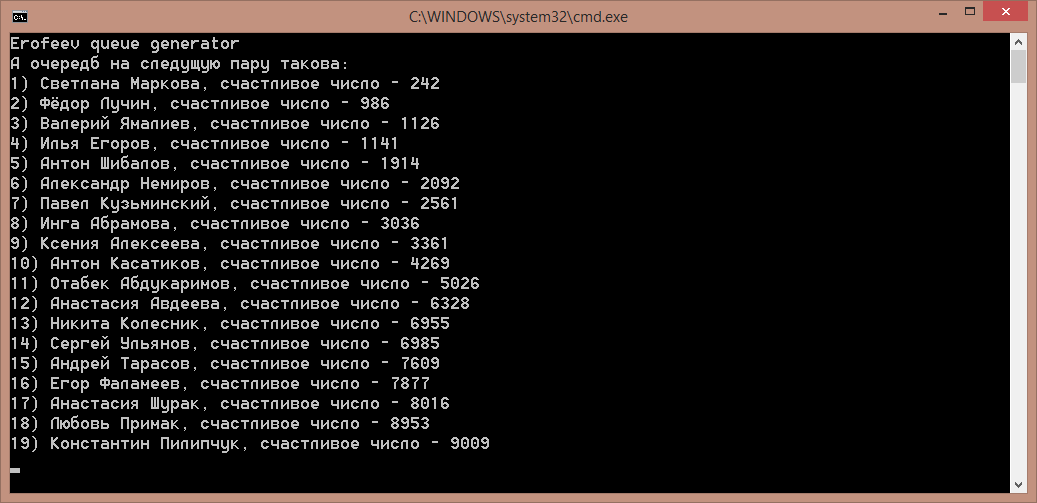


Рисунок – Работа программы

Если открыть output.txt в редакторе, можно увидеть следующее:

*22 serialization::archive 17 0 0 16 Светлана Маркова 242 11 Фёдор Лучин 986 15 Валерий Ямалиев 1126 11 Илья Егоров 1141 13 Антон Шибалов 1914 17 Александр Немиров 2092 17 Павел Кузьминский 2561 13 Инга Абрамова 3036 16 Ксения Алексеева 3361 15 Антон Касатиков 4269 18 Отабек Абдукаримов 5026 17 Анастасия Авдеева 6328 15 Никита Колесник 6955 14 Сергей Ульянов 6985 14 Андрей Тарасов 7609 13 Егор Фаламеев 7877 15 Анастасия Шурак 8016 13 Любовь Примак 8953 19 Константин Пилипчук 9009*

Кроме чистых данных в файл записываются и некоторые служебные.

# Заключение

Была создана программа, записывающая данные в двоичный файл. Данные записываются и читаются без искажений. Boost serialization хорошо показала себя. Использование сторонней библиотеки сберегло время и улучшило читаемость кода.

# Исходный код

#include "stdafx.h"

#include <iostream>

#include <ctime>

#include "string"

#include "fstream"

#include "boost/archive/text\_oarchive.hpp"

#include "boost/archive/text\_iarchive.hpp"

using namespace std;

struct student

{

string name;

int randomnumber;

friend class boost::serialization::access;

template <class Archive>

void serialize(Archive& ar, const unsigned int version)

{

ar & name;

ar & randomnumber;

}

};

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "Erofeev queue generator" << endl;

cout << "А очередб на следущую пару такова:" << endl;

//Массив имён студентов

string names[19]={

"Егор Фаламеев",

"Валерий Ямалиев",

"Илья Егоров",

"Анастасия Шурак",

"Андрей Тарасов",

"Отабек Абдукаримов",

"Любовь Примак",

"Инга Абрамова",

"Антон Шибалов",

"Антон Касатиков",

"Ксения Алексеева",

"Никита Колесник",

"Александр Немиров",

"Анастасия Авдеева",

"Фёдор Лучин",

"Светлана Маркова",

"Павел Кузьминский",

"Константин Пилипчук",

"Сергей Ульянов"};

srand(time(0));

student students[19];

for(int i=0; i<19;i++)

{

students[i].name = names[i];

students[i].randomnumber = rand() % 10000;

}

//сортировка по возрастанию randomnumber

for(int i=0;i<19;i++)

{

for(int j=i; j<19;j++)

{

if (students[j].randomnumber < students[i].randomnumber)

{

//свапнуть студентов

student temp = students[i];

students[i] = students[j];

students[j] = temp;

}

}

}

//Запись в файл

ofstream file("output.txt", ios\_base::out);

boost::archive::text\_oarchive oa(file);

for (int i = 0; i < 19; i++)

oa << students[i];

file.close();

//Создание чистого массива студентов

student students2[19];

//Чтение из файла

ifstream file2("output.txt");

boost::archive::text\_iarchive ia(file2);

for (int i = 0; i < 19; i++)

ia >> students2[i];

for(int i=1; i<20; i++)

cout << i << ") " << students2[i-1].name <<", счастливое число - "<< students2[i-1].randomnumber << endl;

cin.get();

return 0;

}