Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №11

за 2 семестр

По дисциплине: «ОАиП»

Тема: «Бинарные и текстовые файлы»

Выполнила:

Студентка 1 курса

Группы ПО-3(1)

Гаврилюк Р.И.

Проверил:

Хацкевич М. В.

2019

Лабораторная работа №11

Бинарные и текстовые файлы

Цель работы: изучить принципы программирования с использованием бинарных файлов. Ознакомиться с основными функциями для работы с бинарными файлами.

Вариант 24

**Задание:**

В программу, разработанную в лабораторной работе 10, добавить чтение и сохранение данных массива структур при помощи бинарных файлов следующим образом:

1. При первом запуске программы должен создаваться бинарный или текстовый файл на выбор пользователя для хранения данных из массива структур.
2. При добавлении новой записи в массив структур в файл должна дописываться новая запись, без изменения остальных записей.
3. При повторном запуске программы, если файл уже существует, то информация в массив структур должна читаться из этого файла. Если файл отсутствует, то он должен создаваться (см. Пункт 1).
4. Все изменения (сортировка, изменения полей записи, удаление записи) – сохраняются в файле при помощи полной перезаписи содержимого.
5. Сделать вывод о том, какие преимущества использования конкретного типа файлов (бинарные или текстовые) в решаемой вами задаче.

Блок-схема:





Текст программы:

#include <iostream>

#include <windows.h>

#include <fstream>

#include <iomanip>

using namespace std;

enum Continent { Europe = 1, Africa, North\_America, South\_America, Australia, Asia };

union Sea {

int num;

char access;

};

struct Country {

char name[20],

capital[20],

currency[20];

int population;

Continent continent;

Sea sea;

unsigned goverment : 2;

};

int menu(); //организация меню

void input(Country \*&array, int &N); //ввод данных массива структуры

void output(Country \*array, int N); //вывод данных массива структур

void sort(Country \*array, int N, int key); //сортировка во алфавиту по фамилии

int search(Country \*array, int N); //поиск по заданному параметру

void deleted(Country \*&array, int &N, int key); //удаление структуры из массива структур

void write\_txt(Country \*array, int N);

void read\_txt(Country \*&array, int &N);

void add\_txt(Country \*&array, int &N);

void write\_bin(Country \*array, int N);

void read\_bin(Country \*&array, int &N);

void add\_bin(Country \*&array, int &N);

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

menu();

return 0;

}

int menu() {

int choise, N = 0;

Country \*array = new Country[N];

int key;

cout << "Работать с текстовым(1) или бинарным файлом(0)?:\n";

cin >> key;

system("cls");

do {

cout << "Выберите действие:\n";

cout << "1) Ввод массива структур\n";

cout << "2) Вывод массива структур\n";

cout << "3) Сортировка массива структур по алфавит\n";

cout << "4) Поиск страны с наибольшим населением\n";

cout << "5) Удаление заданной структуры\n";

cout << "6) Добавление новых элементов\n";

cout << "7) Выход\n";

cin >> choise;

system("cls");

switch (choise)

{

case 1:

if (key == 1) {

read\_txt(array, N);

}

else if (key == 0) {

read\_bin(array, N);

}

break;

case 2:

cin.ignore();

output(array, N);

break;

case 3:

cin.ignore();

sort(array, N, key);

break;

case 4:

cin.ignore();

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

if (array[i].population == search(array, N)) {

cout << "Страна с самым большим населением:\n";

cout << array[i].name << " (" << array[i].population << " человек)\n";

}

}

break;

case 5:

cin.ignore();

deleted(array, N, key);

break;

case 6:

cin.ignore();

if (key == 1) {

add\_txt(array, N);

}

else if (key == 0) {

add\_bin(array, N);

}

break;

case 7: return 0;

}

cin.ignore();

system("cls");

} while (true);

delete[] array;

}

void input(Country \*&array, int &N) {

int n;

cout << "Введите количество стран: \n";

cin >> n;

N = N + n;

Country\* tempArr = new Country[N];

for (int i = 0; i < N - n; i++) {

tempArr[i] = array[i];

}

array = new Country[N];

for (int i = 0; i < N - n; i++) {

array[i] = tempArr[i];

}

delete tempArr;

system("cls");

cin.ignore();

for (int i = N - n; i < N; i++) {

int choise;

cout << "1) Название страны: ";

gets\_s(array[i].name);

cout << "2) Столица: ";

gets\_s(array[i].capital);

cout << "3) Валюта: ";

gets\_s(array[i].currency);

cout << "4) Континент (1-Европа, 2-Африка, 3-Северная Америка, 4-Южная Америка, 5-Австралия, 6-Азия): ";

cin >> choise;

array[i].continent = (Continent)choise;

cout << "5) Выход к морю (1-есть, 2-нет): ";

cin >> array[i].sea.num;

if (array[i].sea.num == 1)

array[i].sea.access = '+';

if (array[i].sea.num == 2)

array[i].sea.access = '-';

cout << "6) Население: ";

cin >> array[i].population;

cout << "7) Форма правления (1-монархия, 2-республика): ";

cin >> choise;

array[i].goverment = choise;

cin.ignore();

system("cls");

}

}

void output(Country \*array, int N) {

cout << setw(8) << left << "№" << setw(15) << left << "Страна" << setw(15) << left << "Столица" << setw(15) << left << "Валюта";

cout << setw(20) << left << "Континент" << setw(15) << left << "Выход к морю" << setw(15) << left << "Население" << setw(15) << left << "Форма правления\n";

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

cout << i + 1 << "\t";

cout << setw(15) << left << array[i].name;

cout << setw(15) << left << array[i].capital;

cout << setw(15) << left << array[i].currency;

switch (array[i].continent) {

case Europe:

cout << setw(20) << left << "Европа";

break;

case Africa:

cout << setw(20) << left << "Африка";

break;

case North\_America:

cout << setw(20) << left << "Северная Америка";

break;

case South\_America:

cout << setw(20) << left << "Южная Америка";

break;

case Australia:

cout << setw(20) << left << "Австралия";

break;

case Asia:

cout << setw(20) << left << "Азия";

break;

}

cout << setw(15) << left << array[i].sea.access;

cout << setw(15) << left << array[i].population;

if (array[i].goverment == 1)

cout << setw(15) << left << "монархия";

if (array[i].goverment == 2)

cout << setw(15) << left << "республика";

cout << endl;

}

}

void sort(Country \*array, int N, int key) {

for (int i = 0; i < N - 1; i++) {

for (int j = i; j < N; j++) {

if (strcmp(array[j].name, array[i].name) < 0)

swap(array[i], array[j]);

}

}

if (key == 1) {

write\_txt(array, N);

}

else if (key == 0) {

write\_bin(array, N);

}

output(array, N);

}

int search(Country \*array, int N) {

int max = array[0].population, max\_ind = 0;

for (int i = 1; i < N; i++) {

if (array[i].population > max)

max = array[i].population;

}

return max;

}

void deleted(Country \*&array, int &N, int key) {

char Name[20];

cout << "Введите название страны, которую хотите удалить.\n";

gets\_s(Name);

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

if (strcmp(array[i].name, Name) == 0) {

N--;

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

array[j] = array[j + 1];

i--;

}

}

Country\* tempArr = new Country[N];

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

tempArr[i] = array[i];

array = new Country[N];

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

array[i] = tempArr[i];

delete tempArr;

system("cls");

output(array, N);

if (key == 1) {

write\_txt(array, N);

}

else if (key == 0) {

write\_bin(array, N);

}

}

void write\_txt(Country \*array, int N) {

ofstream file\_txt;

file\_txt.open("file\_txt.txt");

if (!file\_txt) {

std::cout << "Ошибка при открытии файла.\n";

}

file\_txt << setw(15) << left << "Страна" << setw(15) << left << "Столица";

file\_txt << setw(15) << left << "Валюта" << setw(20) << left << "Континент";

file\_txt << setw(15) << left << "Выход к морю" << setw(15) << left << "Население";

file\_txt << left << "Форма правления";

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

file\_txt << endl;

file\_txt << setw(15) << left << array[i].name;

file\_txt << setw(15) << left << array[i].capital;

file\_txt << setw(15) << left << array[i].currency;

switch (array[i].continent) {

case Europe:

file\_txt << setw(20) << left << "Европа";

break;

case Africa:

file\_txt << setw(20) << left << "Африка";

break;

case North\_America:

file\_txt << setw(20) << left << "Северная\_Америка";

break;

case South\_America:

file\_txt << setw(20) << left << "Южная\_Америка";

break;

case Australia:

file\_txt << setw(20) << left << "Австралия";

break;

case Asia:

file\_txt << setw(20) << left << "Азия";

break;

}

file\_txt << setw(15) << left << array[i].sea.access;

file\_txt << setw(15) << left << array[i].population;

if (array[i].goverment == 1) {

file\_txt << left << "монархия";

}

else if (array[i].goverment == 2) {

file\_txt << left << "республика";

}

}

file\_txt.close();

}

void read\_txt(Country \*&array, int &N) {

ifstream file\_txt;

file\_txt.open("file\_txt.txt");

N = 0;

if (!file\_txt.is\_open()) {

input(array, N);

write\_txt(array, N);

}

else {

array = new Country[N];

char temp[1024];

file\_txt.getline(temp, 1024, '\n');

for (int i = 0; !file\_txt.eof(); i++) {

N++;

Country \*tempArr = new Country[N - 1];

for (int j = 0; j < N - 1; j++) {

tempArr[j] = array[j];

}

array = new Country[N];

for (int j = 0; j < N - 1; j++) {

array[j] = tempArr[j];

}

delete tempArr;

file\_txt >> array[i].name >> array[i].capital >> array[i].currency;

char temp[20];

file\_txt >> temp;

if (strcmp(temp, "Европа") == NULL) {

array[i].continent = Europe;

}

else if (strcmp(temp, "Африка") == NULL) {

array[i].continent = Africa;

}

else if (strcmp(temp, "Северная\_Америка") == NULL) {

array[i].continent = North\_America;

}

else if (strcmp(temp, "Южная\_Америка") == NULL) {

array[i].continent = South\_America;

}

else if (strcmp(temp, "Австралия") == NULL) {

array[i].continent = Australia;

}

else if (strcmp(temp, "Азия") == NULL) {

array[i].continent = Asia;

}

file\_txt >> array[i].sea.access >> array[i].population;

file\_txt >> temp;

if (strcmp(temp, "монархия") == NULL) {

array[i].goverment = 1;

}

if (strcmp(temp, "республика") == NULL) {

array[i].goverment = 2;

}

}

cout << "Данные считаны из файла.\n";

cin.ignore();

file\_txt.close();

}

}

void add\_txt(Country \*&array, int &N) {

ofstream file\_txt;

file\_txt.open("file\_txt.txt", ios\_base::app);

input(array, N);

write\_txt(array, N);

file\_txt.close();

}

void write\_bin(Country \*array, int N) {

ofstream file\_bin;

file\_bin.open("file\_bin.bin", ios\_base:: binary);

if (!file\_bin) {

std::cout << "Ошибка при открытии файла.\n";

}

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

file\_bin.write((char\*)&array[i], sizeof(array[i]));

}

file\_bin.close();

}

void read\_bin(Country \*&array, int &N) {

ifstream file\_bin;

file\_bin.open("file\_bin.bin", ios\_base::binary);

N = 0;

if (!file\_bin.is\_open()) {

input(array, N);

write\_bin(array, N);

}

else {

for (int i = 0; !file\_bin.eof() && (file\_bin.peek() != std::ifstream::traits\_type::eof()); i++) {

N++;

Country \*tempArr = new Country[N - 1];

for (int j = 0; j < N - 1; j++) {

tempArr[j] = array[j];

}

array = new Country[N];

for (int j = 0; j < N - 1; j++) {

array[j] = tempArr[j];

}

delete tempArr;

file\_bin.read((char\*)&array[i], sizeof(array[i]));

}

cout << "Данные считаны из файла.\n";

cin.ignore();

file\_bin.close();

}

}

void add\_bin(Country \*&array, int &N) {

ofstream file\_bin;

file\_bin.open("file\_bin.bin", ios\_base::binary | ios\_base::app);

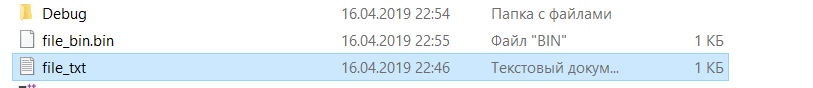
input(array, N);

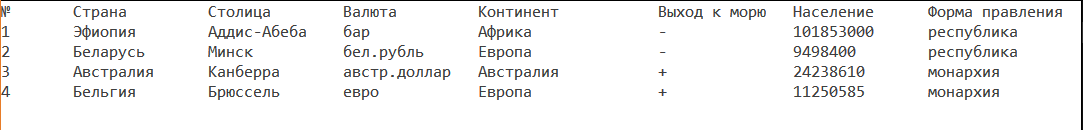
write\_bin(array, N);

file\_bin.close();

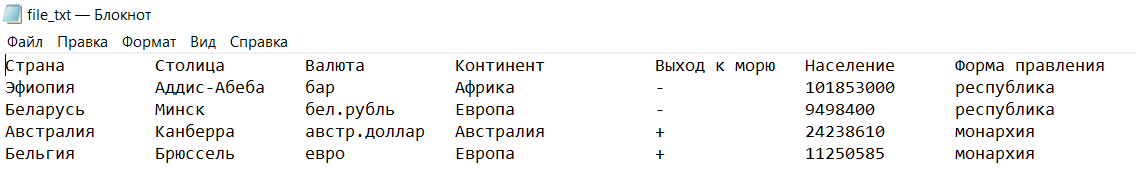
}

Результаты выполнения программы:









Вывод: в данном случае удобнее использовать текстовые файлы, т.к. текстовые файлы более универсальны и файл можно открыть в текстовом редакторе и получить читаемый текст, но двоичный файл может иметь меньший размер.