Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №13

за 2 семестр

По дисциплине: «ОАиП»

Тема: «Модульное программирование»

Выполнила:

Студентка 1 курса

Группы ПО-3(1)

Гаврилюк Р.И.

Проверил:

Хацкевич М. В.

2019

Лабораторная работа №13

Модульное программирование

Цель работы: изучить принципы модульного программирования; ознакомиться с основными возможностями межмодульного взаимодействия.

Вариант 24

**Задание:**

В программу, разработанную в лабораторной работе 12, внести следующие изменения и дополнения:

1. Программа должна быть разделена на несколько модулей (например, работа с файлами, работа с интерфейсом, обработка запросов к базе данных и т.п.).

2. Взаимодействие модулей организовать при помощи вызова функций и переменных внешнего типа.

**Блок-схема:**



**Текст программы:**

1. **main.cpp**

#include <windows.h>

#include "menu.h"

int key;//переменная для определения с каким типом файлов необходимо работать

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("color F0");

char file\_name[20];

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

cin >> key;

cout << "Введите имя файла:\n";

cin.ignore();

gets\_s(file\_name);

if (key == 1) {

strcat(file\_name, ".txt");

}

else if (key == 0) {

strcat(file\_name, ".bin");

}

else {

cout << "Ошибка.\n";

}

system("cls");

menu(file\_name);

return 0;

}

1. **menu.h**

#pragma once

#include "interface.h"

void menu(char \*file\_name); //организация меню

**menu.cpp**

#include "menu.h"

void menu(char \*file\_name) {

int choise, N = 0;

Country \*array = new Country[N];

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 << "0) Выход\n";

cin >> choise;

system("cls");

switch (choise)

{

case 1:

read(array, N, file\_name);

output(array, N);

cin.ignore();

break;

case 2:

cin.ignore();

output(array, N);

break;

case 3:

cin.ignore();

sort(array, N, file\_name);

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, file\_name);

break;

case 6:

cin.ignore();

add(array, N, file\_name);

break;

default: break;

}

cin.ignore();

system("cls");

} while (choise != 0);

delete[] array;

}

1. **interface.h**

#pragma once

#include "text\_file.h"

void add(Country \*&array, int &N, char \*name); //добавление записей в конец

void sort(Country \*array, int N, char \*name); //сортировка

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

void deleted(Country \*&array, int &N, char \*name); //удаление записи

**interface.cpp**

#include "interface.h"

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

cout << "Выберите пункт:\n";

cout << "1 - Сортировка по названию страны\n";

cout << "2 - Сортировка по количеству населения\n";

int choise;

cin >> choise;

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

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

if (choise == 1) {

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

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

}

}

else if (choise == 0) {

if (array[array[j].num].population < array[array[i].num].population) {

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

}

}

}

}

index\_file(N, array);

rewrite(array, N, name);

output(array, N);

cin.ignore();

}

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, char \*name) {

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) {

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

if (array[j].num > array[i].num) {

array[j].num--;

}

}

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");

index\_file(N, array);

rewrite(array, N, name);

output(array, N);

}

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

ofstream file;

if (key == 1) {

file.open(name, ios\_base::app);

}

else if (key == 0) {

file.open(name, ios\_base::app | ios\_base::binary);

}

input(array, N);

file << endl;

write(array, N, file);

file.close();

output(array, N);

index\_file(N, array);

}

1. **text\_file.h**

#pragma once

#include <fstream>

#include "console.h"

void read(Country \*&array, int &N, char \*name); //чтение из файла

void write(Country \*array, int N, ofstream &file); //запись данных в файл

void rewrite(Country \*array, int N, char \*name); //перезапись данных в файле

void index\_file(int N, Country \*index); //индексные файлы

**text\_file.cpp**

#include "text\_file.h"

void rewrite(Country \*array, int N, char \*name) {

ofstream file;

if (key == 1) {

file.open(name);

}

else if (key == 0) {

file.open(name, ios\_base::binary);

}

write(array, N, file);

file.close();

}

void write(Country \*array, int N, ofstream &file) {

file << setw(15) << left << "Страна" << setw(15) << left << "Столица" << setw(15) << left << "Валюта" << setw(20) << left << "Континент" << setw(15) << left << "Выход к морю" << setw(15) << left << "Население" << setw(15) << left << "Форма правления";

ifstream index\_file;

index\_file.open("Indexes.txt");

if (key == 1) {

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

int k;

if (index\_file.is\_open() && !index\_file.eof()) {

index\_file >> k;

index\_file >> k;

}

else k = i;

file << endl;

file << setw(15) << left << array[k].name;

file << setw(15) << left << array[k].capital;

file << setw(15) << left << array[k].currency;

switch (array[k].continent) {

case Europe:

file << setw(20) << left << "Европа";

break;

case Africa:

file << setw(20) << left << "Африка";

break;

case North\_America:

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

break;

case South\_America:

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

break;

case Australia:

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

break;

case Asia:

file << setw(20) << left << "Азия";

break;

}

file << setw(15) << left << array[k].sea.access;

file << setw(15) << left << array[k].population;

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

file << left << "монархия";

}

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

file << left << "республика";

}

}

}

else if (key == 0) {

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

int k;

if (index\_file.is\_open()) {

index\_file >> k;

index\_file >> k;

}

else k = i;

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

}

}

index\_file.close();

}

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

ifstream file;

if (key == 1) {

file.open(name);

}

else if (key == 0) {

file.open(name, ios\_base::binary);

}

N = 0;

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

input(array, N);

rewrite(array, N, name);

}

else {

array = new Country[N];

char temp[1024];

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

for (int i = 0; file.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;

array[0].num = 0;

if (i >= 1)

array[i].num = array[i - 1].num + 1;

if (key == 1) {

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

char temp[20];

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

file >> temp;

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

array[i].goverment = 1;

}

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

array[i].goverment = 2;

}

}

else if (key == 0) {

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

}

}

file.close();

}

}

void index\_file(int N, Country \*index) {

ofstream fout("Indexes.txt");

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

fout << i + 1 << " " << index[i].num;

if (i < N - 1)

fout << endl;

}

fout.close();

}

1. **console.h**

#pragma once

#include <iostream>

#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,

num;

Continent continent;

Sea sea;

unsigned goverment : 2;

};

extern int key;

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

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

**console.cpp**

#include "console.h"

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();

array[0].num = 0;

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

if (i >= 1) {

array[i].num = array[i - 1].num + 1;

}

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" << 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[array[i].num].name;

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

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

switch (array[array[i].num].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[array[i].num].sea.access;

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

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

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

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

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

cout << endl;

}

}

Вывод: изучила принципы модульного программирования и ознакомилась с основными возможностями межмодульного взаимодействия.