Пенза 2020

Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

Выполнил:

студенты группы 17ВВ1

Кокин Денис

Беленков Никита

Принял:

Дорошенко И.Н.

**ОТЧЁТ**

по лабораторной работе №6

по курсу «Технологии программирования»

на тему «Маршалинг и работа с внешними библиотеками»

**Цель работы:** Изучение механизма вызова функций из внешних библиотек. Получение навыков передачи параметров и возврата результата из функций внешних библиотек.

Лабораторное задание выполняется на базе лабораторной работы №1.

В работе требуется реализовать программу, состоящую из DLL библиотеки и консольного приложения. Программа должна выполнять задание, указанное в лабораторной работе №1.

Консольное приложение должно реализовать набор классов, приведенный в лабораторной работе №1. Во всех функция, выполняющих ввод/вывод данных, вместо стандартных функций должны использоваться вызовы из DLL библиотеки.

DLL библиотека:

* Библиотека должны быть реализована на языке C/C++.
* В библиотеке должны быть реализованы функции ввода/вывода данных.

Консольное приложение:

* Приложение должно быть реализовано на языке C#.
* Для ввода/вывода данных приложение должно использовать вызовы DLL библиотеки.

Требования к программе:

* Для базового класса (**Base**) ввод и вывод данных должен выполняться индивидуально для каждого поля. Т.е. в DLL библиотеке должны быть реализованы функции печати и сканирования отдельно для чисел, строк и т.д. В приложении должен быть сделан импорт функций из DLL библиотеки.
* Для всех 3 производных классов (**Derived1**, **Derived2** и **Derived3**) ввод и вывод данных должен выполняться с использованием структуры. Т.е. в DLL библиотеке должны быть реализованы 3 структуры (по одной для каждого класса), которые должны использоваться для ввода/вывода данных. Для каждой структуры необходимо реализовать по одной функции ввода и вывода данных. В приложении должен быть сделан маршалинг для 3 структур и импорт функций из DLL библиотеки.

**Листинг программы**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

class Township : Locality

{

public int SchoolCount;

public override void output()

{

base.output();

Console.WriteLine("A count of schools: " + this.SchoolCount);

}

public void input()

{

base.input();

Console.Write("Enter a count of schools: ");

SchoolCount = Int32.Parse(Console.ReadLine());

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

abstract class OutputAbstract

{

public abstract void output();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

interface InputInterface

{

void input();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

class City : Locality

{

public bool isRegionCenter;

public override void output()

{

base.output();

Console.WriteLine("This is a region center: " + this.isRegionCenter);

}

public void input()

{

base.input();

Console.Write("Is this a region center?: ");

isRegionCenter = System.Convert.ToBoolean(Console.ReadLine());

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

class Locality : OutputAbstract, InputInterface

{

public String Name;

public String Region;

public Locality()

{

Population = 0;

Budget = 0;

Code = 0;

}

public Locality(int Population, int Budget, int Code, String Name = "Unnamed", String Region = "Unnamed")

{

this.Population = Population;

this.Budget = Budget;

this.Code = Code;

this.Name = Name;

this.Region = Region;

}

public Locality(String Name, String Region, int Population = 0, int Budget = 0, int Code = 0)

{

this.Population = Population;

this.Budget = Budget;

this.Code = Code;

this.Name = Name;

this.Region = Region;

}

public void ClassInfo()

{

Console.WriteLine("Full region name: " + this.Name + " in " + this.Region);

this.AbleBodied();

this.BudgetToUSD();

}

~Locality()

{

Name = null;

Region = null;

Population = 0;

Budget = 0;

Code = 0;

}

public void input()

{

Console.Write("Input a name of the region: ");

this.Name = Console.ReadLine();

Console.Write("Input a regional center of the region: ");

this.Region = Console.ReadLine();

Console.Write("Input a population of the region: ");

this.Population = Int32.Parse(Console.ReadLine());

Console.Write("Input a budget of the region: ");

this.Budget = Int32.Parse(Console.ReadLine());

Console.Write("Input a code of the region: ");

this.Code = Int32.Parse(Console.ReadLine());

}

public override void output()

{

Console.WriteLine("A name of the region: " + this.Name);

Console.WriteLine("A regional center of the region: " + this.Region);

Console.WriteLine("A population of the region: " + this.Population);

Console.WriteLine("A budget of the region: " + this.Budget);

Console.WriteLine("A code of the region: " + this.Code);

}

protected int Population;

protected int Code;

protected int Budget;

protected void AbleBodied()

{

Console.WriteLine("Able-bodied population: " + (this.Population / 2));

}

private void BudgetToUSD()

{

Console.WriteLine("Budget of locality: " + (this.Budget / 70) + " USD");

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Confrontation

{

class Village : Township

{

public int ShopCount;

public override void output()

{

base.output();

Console.WriteLine("A count of shops: " + this.ShopCount);

}

public void input()

{

base.input();

Console.Write("Enter a count of shops: ");

ShopCount = Int32.Parse(Console.ReadLine());

}

public static bool operator <=(Village pSampleVillage, int pParam)

{

if ((pSampleVillage.Name.Length <= pParam))

return true;

else

return false;

}

public static bool operator >=(Village pSampleVillage, int pParam)

{

if ((pSampleVillage.Name.Length >= pParam))

return true;

else

return false;

}

public static bool operator <=(Village pSampleVillage, string pParam)

{

if (pSampleVillage.Name.Length < pParam.Length)

return true;

if (pSampleVillage.Name.Length >= 5 && pParam.Length >= 5)

{

if (pSampleVillage.Name.Substring(

pSampleVillage.Name.Length - 5).Equals(

pParam.Substring(pParam.Length - 5)

))

return true;

else

return false;

}

return false;

}

public static bool operator >=(Village pSampleVillage, string pParam)

{

if (pSampleVillage.Name.Length > pParam.Length)

return true;

if (pSampleVillage.Name.Length >= 5 && pParam.Length >= 5)

{

if (pSampleVillage.Name.Substring(

pSampleVillage.Name.Length - 5).Equals(

pParam.Substring(pParam.Length - 5)

))

return true;

else

return false;

}

return false;

}

public static bool operator >=(Village pSampleVillage, Village pParam)

{

if (pSampleVillage.Name.Length > pParam.Name.Length)

return true;

if (pSampleVillage.Name.Length >= 5 && pParam.Name.Length >= 5)

{

if (pSampleVillage.Name.Substring(

pSampleVillage.Name.Length - 5).Equals(

pParam.Name.Substring(pParam.Name.Length - 5)

))

return true;

else

return false;

}

return false;

}

public static bool operator <=(Village pSampleVillage, Village pParam)

{

if (pSampleVillage.Name.Length < pParam.Name.Length)

return true;

if (pSampleVillage.Name.Length >= 5 && pParam.Name.Length >= 5)

{

if (pSampleVillage.Name.Substring(

pSampleVillage.Name.Length - 5).Equals(

pParam.Name.Substring(pParam.Name.Length - 5)

))

return true;

else

return false;

}

return false;

}

static public Village operator &(Village pSampleVillage, int pParam){

pSampleVillage.Name = pSampleVillage.Name.Substring(0, 6) + pParam + pSampleVillage.Name.Substring(6);

return pSampleVillage;

}

static public Village operator &(Village pSampleVillage, string pParam)

{

pSampleVillage.Name = pSampleVillage.Name.Substring(0, 6) + pParam +pSampleVillage.Name.Substring(6);

return pSampleVillage;

}

static public Village operator &(Village pSampleVillage, Village pParam)

{

pSampleVillage.Name = pSampleVillage.Name.Substring(0, 6) + pParam.Name + pSampleVillage.Name.Substring(6);

return pSampleVillage;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Reflection;

using System.Runtime.InteropServices;

namespace Confrontation {

class Program{

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void InputCity(ref City CitySample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void OutputCity(ref City CitySample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void InputTownship(ref Township TownshipSample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void OutputTownship(ref Township TownshipSample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void InputVillage(ref Village VillageSample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void OutputVillage(ref Village VillageSample);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern int InputInt();

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void OutputInt(int Param);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern int InputStr([MarshalAs(UnmanagedType.LPWStr)] StringBuilder dst);

[DllImport("DLLdl.dll", CallingConvention = CallingConvention.Cdecl, CharSet = CharSet.Unicode)]

public static extern void OutputStr(string src);

static void Main(string[] args){

Locality SampleLocality = new Locality();

StringBuilder BufferString = new StringBuilder(100);

Console.WriteLine("Input a name of the region: ");

InputStr(BufferString);

SampleLocality.Name = BufferString.ToString();

BufferString.Clear();

Console.WriteLine("Input a regional center of the region: ");

InputStr(BufferString);

SampleLocality.Region = BufferString.ToString();

BufferString.Clear();

Console.WriteLine("Input a population of the region: ");

SampleLocality.Population = InputInt();

SampleLocality.output();

City SampleCity = new City();

InputCity(ref SampleCity);

OutputCity(ref SampleCity);

Township SampleTownship = new Township();

InputTownship(ref SampleTownship);

OutputTownship(ref SampleTownship);

Village SampleVillage = new Village();

InputVillage(ref SampleVillage);

OutputVillage(ref SampleVillage);

Console.Read();

}

}

}

**Листинг DLL-библиотеки**

#include <iostream>

#include<windows.h>

#include <locale>

#include <codecvt>

#include <string>

#pragma warning(disable, 4996)

typedef struct \_struct1 {

public:

int Population;

int Code;

int Budget;

int isRegionCenter;

wchar\_t Name[10];

wchar\_t Region[10];

}CityStruct;

int main() {

}

extern "C" int \_\_declspec(dllexport)\_cdecl InputInt() {

int buf;

std::cin >> buf;

return buf;

}

extern "C" void \_\_declspec(dllexport)\_cdecl OutputInt(int Param) {

std::cout << Param << std::endl;

}

extern "C" void \_\_declspec(dllexport)\_cdecl InputStr(wchar\_t\* dst) {

wchar\_t\* src = new wchar\_t[10];

std::wcin >> src;

wcscpy(dst, src);

}

extern "C" void \_\_declspec(dllexport)\_cdecl OutputStr(wchar\_t\* src) {

std::wcout << src << std::endl;

}

extern "C" void \_\_declspec(dllexport)\_cdecl InputCity(CityStruct \* pSampleStruct) {

std::cout << "Input a name of the region: ";

std::wcin >> pSampleStruct->Name;

std::cout <<"Input a population of the region: ";

std::wcin >> pSampleStruct->Region;

std::cout << "Input a budget of the region: ";

std::cin >> pSampleStruct->Population;

std::cout << "Input a regional center of the region: ";

std::cin >> pSampleStruct->Budget;

std::cout <<"Input a code of the region: ";

std::cin >> pSampleStruct->Code;

std::cout <<"This is a region center: ";

std::cin >> pSampleStruct->isRegionCenter;

}

extern "C" void \_\_declspec(dllexport)\_cdecl OutputCity(CityStruct \* pSampleStruct) {

std::wcout << "A name of the region: " << pSampleStruct->Name << std::endl;

std::wcout << "A regional center of the region: " << pSampleStruct->Region << std::endl;

std::cout <<"A population of the region: " << pSampleStruct->Population << std::endl;

std::cout <<"A budget of the region: " << pSampleStruct->Budget << std::endl;

std::cout <<"A code of the region: " << pSampleStruct->Code << std::endl;

std::cout <<"This is a region center:" << pSampleStruct->isRegionCenter << std::endl;

}

typedef struct \_struct2 {

public:

int Population;

int Code;

int Budget;

int SchoolCount;

wchar\_t Name[10];

wchar\_t Region[10];

}TownshipStruct;

extern "C" void \_\_declspec(dllexport)\_cdecl InputTownship(TownshipStruct \* TownshipStructSample) {

std::cout <<"Input a name of the region: ";

std::wcin >> TownshipStructSample->Name;

std::cout <<"Input a regional center of the region: ";

std::wcin >> TownshipStructSample->Region;

std::cout <<"Input a population of the region: ";

std::cin >> TownshipStructSample->Population;

std::cout <<"Input a budget of the region: ";

std::cin >> TownshipStructSample->Budget;

std::cout <<"Input a code of the region: ";

std::cin >> TownshipStructSample->Code;

std::cout <<"Input a count of schools: ";

std::cin >> TownshipStructSample->SchoolCount;

}

extern "C" void \_\_declspec(dllexport)\_cdecl OutputTownship(TownshipStruct \* TownshipStructSample) {

std::wcout <<"A name of the region: " << TownshipStructSample->Name << std::endl;

std::wcout <<"A regional center of the region: " << TownshipStructSample->Region << std::endl;

std::cout <<"A population of the region: " << TownshipStructSample->Population << std::endl;

std::cout <<"A budget of the region: " << TownshipStructSample->Budget << std::endl;

std::cout <<"A code of the region: " << TownshipStructSample->Code << std::endl;

std::cout <<"A count of schools: " << TownshipStructSample->SchoolCount << std::endl;

}

typedef struct \_struct3 {

public:

int Population;

int Code;

int Budget;

int SchoolCount;

int ShopCount;

wchar\_t Name[10];

wchar\_t Region[10];

}VillageStruct;

extern "C" void \_\_declspec(dllexport)\_cdecl InputVillage(VillageStruct \* VillageStructSample) {

std::cout <<"Input a name of the region: ";

std::wcin >> VillageStructSample->Name;

std::cout <<"Input a regional center of the region: ";

std::wcin >> VillageStructSample->Region;

std::cout <<"Input a population of the region: ";

std::cin >> VillageStructSample->Population;

std::cout <<"Input a budget of the region: ";

std::cin >> VillageStructSample->Budget;

std::cout <<"Input a code of the region: ";

std::cin >> VillageStructSample->Code;

std::cout <<"Input a count of schools: ";

std::cin >> VillageStructSample->SchoolCount;

std::cout <<"Input a count of shops: ";

std::cin >> VillageStructSample->ShopCount;

}

extern "C" void \_\_declspec(dllexport)\_cdecl OutputVillage(VillageStruct \* VillageStructSample) {

std::wcout <<"A name of the region: " << VillageStructSample->Name << std::endl;

std::wcout <<"A regional center of the region: " << VillageStructSample->Region << std::endl;

std::cout <<"A population of the region: " << VillageStructSample->Population << std::endl;

std::cout <<"A budget of the region: " << VillageStructSample->Budget << std::endl;

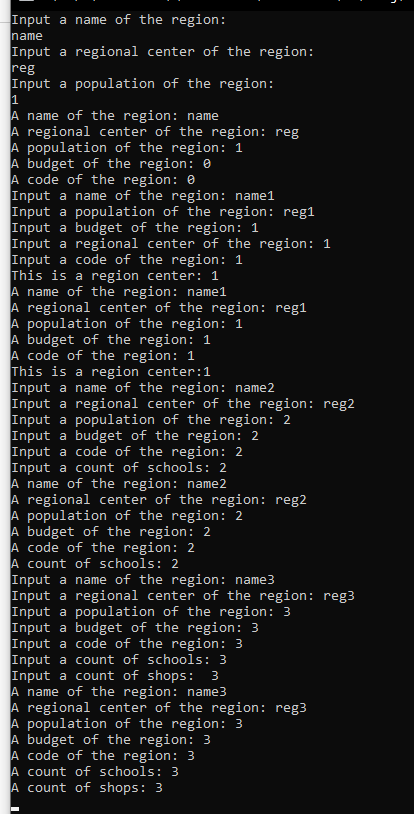
std::cout <<"A code of the region: " << VillageStructSample->Code << std::endl;

std::cout <<"A count of schools: " << VillageStructSample->SchoolCount << std::endl;

std::cout <<"A count of shops: " << VillageStructSample->ShopCount << std::endl;

}

**Результаты работы программы**



**Вывод:** студенты изучили механизмы сохранения и восстановления объектов данных. Получили навыки разработки самовосстанавливающихся структур данных.