Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №6

по курсу «Технологии программирования»

на тему «Маршалинг и работа с внешними библиотеками»

Выполнили:

студенты группы 21ВВВ1

Аляев А. О.

Кирьянов В. Е.

Принял:

к.т.н., доцент Дорошенко И.Н.

к.т.н., доцент Синёв М.П.

Пенза 2023

**Название**

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**

Лабораторное задание выполняется на базе лабораторной работы №1. Из лабораторной работы используется класс **Derived1**.

Требуется объявить 2 делегата для функций Input и Output.

Требования к программе:

* В основной функции программы необходимо создать 3-5 объектов класса **Derived1**.
* Создать 2 объекта делегата для функций Input и Output. Присвоить им адреса функций 1-го созданного объекта класса **Derived1**. Вызвать функции с использованием делегатов.
* Выполнить группирование делегатов. Для ранее созданных объектов делегатов добавить адреса функций оставшихся объектов класса **Derived1** (со 2 по 3-5). Вызвать функции с использованием делегатов.
* Выполнить разгруппирование делегатов. Удалить из объектов делегатов все адреса функций. Вызвать функции с использованием делегатов (убедиться, что ни одна функция не вызвана).

**Листинг**

Файл - Program.cs

using System;

using System.Runtime.InteropServices;

using System.Text;

namespace Lab1

{

internal class Program

{

class Auto

{

protected string type;

protected int volume;

protected int power;

public int price;

public Auto()

{

this.type = "Auto";

this.volume = 0;

this.power = 0;

this.price = 0;

}

public Auto(string type, int price = 200)

{

this.type = type;

this.price = price;

}

public Auto(string type = "Легковой", int volume = 50, int power = 57, int price = 100)

{

this.type = type;

this.volume = volume;

this.power = power;

this.price = price;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool InputStr(

[MarshalAs(UnmanagedType.LPWStr)] StringBuilder str, int len);

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern int InputInt();

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool OutputInt(int num);

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool OutputStr(

[MarshalAs(UnmanagedType.LPWStr)] StringBuilder str, int length);

public virtual void Input()

{

Console.WriteLine("Enter type, volume, power, price");

StringBuilder str = new StringBuilder(10);

InputStr(str, str.Capacity);

type = str.ToString();

volume = InputInt();

power = InputInt();

price = InputInt();

}

public virtual void Output()

{

OutputStr(new StringBuilder(type), type.Length);

OutputInt(volume);

OutputInt(power);

OutputInt(price);

Console.WriteLine("\n");

}

private void SetType(string type)

{

this.type = type;

}

private void SetVolume(int volume)

{

this.volume = volume;

}

private void SetPower(int power)

{

this.power = power;

}

private void SetPrice(int price)

{

this.price = price;

}

protected string Gettype()

{

return type;

}

protected int GetVolume() {

return volume;

}

protected int GetPower()

{

return power;

}

protected int GetPrice()

{

return price;

}

~Auto()

{

this.type = null;

this.volume = 0;

this.power = 0;

this.price = 0;

}

}

[StructLayout(

LayoutKind.Sequential,

CharSet = CharSet.Unicode)]

public struct PassengerStruct

{

[MarshalAs(

UnmanagedType.ByValTStr,

SizeConst = 100)]

public string type;

public int volume;

public int power;

public int price;

public int doors;

public PassengerStruct(string type, int volume, int power, int price, int doors)

{

this.type = type;

this.volume = volume;

this.power = power;

this.price = price;

this.doors = doors;

}

}

class Passenger : Auto

{

private int doors;

public Passenger(string type, int volume, int power, int price, int doors) : base(type, volume, power, price)

{

this.doors = 0;

}

public Passenger()

{

this.type = "Passenger";

this.doors = 0;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool InputPassenger(

ref PassengerStruct ptr);

public override void Input()

{

PassengerStruct structure = new PassengerStruct();

InputPassenger(ref structure);

type = structure.type;

volume = structure.volume;

power = structure.power;

price = structure.price;

doors = structure.doors;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool OutputPassenger(

ref PassengerStruct ptr);

public override void Output()

{

PassengerStruct structure = new PassengerStruct(type, volume, power, price, doors);

OutputPassenger(ref structure);

Console.WriteLine("\n");

}

}

[StructLayout(

LayoutKind.Sequential,

CharSet = CharSet.Unicode)]

public struct CargoStruct

{

[MarshalAs(

UnmanagedType.ByValTStr,

SizeConst = 100)]

public string type;

public int volume;

public int power;

public int price;

public int weight;

public CargoStruct(string type, int volume, int power, int price, int weight)

{

this.type = type;

this.volume = volume;

this.power = power;

this.price = price;

this.weight = weight;

}

}

class Cargo : Auto

{

public int weight;

public Cargo()

{

this.weight = 0;

this.type = "Cargo";

}

public Cargo(string type, int volume, int power, int price, int weight) : base(type, volume, power, price)

{

this.weight = 0;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool InputCargo(

ref CargoStruct ptr);

public override void Input()

{

CargoStruct structure = new CargoStruct();

InputCargo(ref structure);

type = structure.type;

volume = structure.volume;

power = structure.power;

price = structure.price;

weight = structure.weight;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool OutputCargo(

ref CargoStruct ptr);

public override void Output()

{

CargoStruct structure = new CargoStruct(type, volume, power, price, weight);

OutputCargo(ref structure);

Console.WriteLine("\n");

}

}

[StructLayout(

LayoutKind.Sequential,

CharSet = CharSet.Unicode)]

public struct KamazStruct

{

[MarshalAs(

UnmanagedType.ByValTStr,

SizeConst = 100)]

public string type;

public int volume;

public int power;

public int price;

public int weight;

[MarshalAs(

UnmanagedType.ByValTStr,

SizeConst = 100)]

public string model;

public KamazStruct(string type, int volume, int power, int price, int weight, string model)

{

this.type = type;

this.volume = volume;

this.power = power;

this.price = price;

this.weight = weight;

this.model = model;

}

}

class Kamaz : Cargo

{

private string model;

public Kamaz()

{

this.model = "";

this.type = "Kamaz";

}

public Kamaz(string type, int volume, int power, int price, int weight, string model) : base(type, volume, power, price, weight)

{

this.model = model;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool InputKamaz(

ref KamazStruct ptr);

public override void Input()

{

KamazStruct structure = new KamazStruct();

InputKamaz(ref structure);

type = structure.type;

volume = structure.volume;

power = structure.power;

price = structure.price;

weight = structure.weight;

model = structure.model;

}

[DllImport(@"DLL.dll",

CallingConvention = CallingConvention.Cdecl,

CharSet = CharSet.Unicode)]

public static extern bool OutputKamaz(

ref KamazStruct ptr);

public override void Output()

{

KamazStruct structure = new KamazStruct(type, volume, power, price, weight, model);

OutputKamaz(ref structure);

Console.WriteLine("\n");

}

}

static void Main(string[] args)

{

Auto auto = new Auto();

auto.Input();

auto.Output();

Passenger passenger = new Passenger();

passenger.Input();

passenger.Output();

Cargo cargo = new Cargo();

cargo.Input();

cargo.Output();

Kamaz kamaz = new Kamaz();

kamaz.Input();

kamaz.Output();

}

}

}

**Файл - dllmain.cpp**

// dllmain.cpp : Определяет точку входа для приложения DLL.

#include "pch.h"

#include <iostream>

#include "dllmain.h"

BOOL APIENTRY DllMain( HMODULE hModule,

DWORD ul\_reason\_for\_call,

LPVOID lpReserved

)

{

switch (ul\_reason\_for\_call)

{

case DLL\_PROCESS\_ATTACH:

case DLL\_THREAD\_ATTACH:

case DLL\_THREAD\_DETACH:

case DLL\_PROCESS\_DETACH:

break;

}

return TRUE;

}

\_\_declspec(dllexport)

bool InputPassenger(PassengerStruct\* ptr)

{

printf("Enter type, volume, power, price, doors: \n");

wscanf\_s(L"%s %d %d %d %d", ptr->type, 100, &ptr->volume, &ptr->power, &ptr->price, &ptr->doors);

return TRUE;

}

\_\_declspec(dllexport)

bool OutputPassenger(PassengerStruct\* ptr)

{

wprintf\_s(L"%s %d %d %d %d", ptr->type, ptr->volume, ptr->power, ptr->price, ptr->doors);

return TRUE;

}

\_\_declspec(dllexport)

bool InputCargo(CargoStruct\* ptr)

{

printf("Enter type, volume, power, price, weight: \n");

wscanf\_s(L"%s %d %d %d %d", ptr->type, 100, &ptr->volume, &ptr->power, &ptr->price, &ptr->weight);

return TRUE;

}

\_\_declspec(dllexport)

bool OutputCargo(CargoStruct\* ptr)

{

wprintf\_s(L"%s %d %d %d %d", ptr->type, ptr->volume, ptr->power, ptr->price, ptr->weight);

return TRUE;

}

\_\_declspec(dllexport)

bool InputKamaz(KamazStruct\* ptr)

{

printf("Enter type, volume, power, price, weight, model: \n");

wscanf\_s(L"%s %d %d %d %d %s", ptr->type, 100, &ptr->volume, &ptr->power, &ptr->price, &ptr->weight, ptr->model, 100);

return TRUE;

}

\_\_declspec(dllexport)

bool OutputKamaz(KamazStruct\* ptr)

{

wprintf\_s(L"%s %d %d %d %d %s", ptr->type, ptr->volume, ptr->power, ptr->price, ptr->weight, ptr->model);

return TRUE;

}

\_\_declspec(dllexport)

bool InputStr(wchar\_t\* src, int length)

{

wscanf\_s(L"%s", src, length);

return TRUE;

}

\_\_declspec(dllexport)

int InputInt()

{

int num;

scanf\_s("%d", &num);

return num;

}

\_\_declspec(dllexport)

bool OutputStr(wchar\_t\* src, int length)

{

wprintf\_s(L"%s ", src, length);

return TRUE;

}

\_\_declspec(dllexport)

bool OutputInt(int num)

{

wprintf\_s(L"%d ", num);

return TRUE;

}

**Файл – dllmain.h**

typedef struct \_PassengerStruct

{

wchar\_t type[100];

int volume;

int power;

int price;

int doors;

} PassengerStruct;

typedef struct \_CargoStruct

{

wchar\_t type[100];

int volume;

int power;

int price;

int weight;

} CargoStruct;

typedef struct \_KamazStruct

{

wchar\_t type[100];

int volume;

int power;

int price;

int weight;

wchar\_t model[100];

} KamazStruct;

#pragma once

extern "C" \_\_declspec(dllexport) bool InputPassenger(PassengerStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool OutputPassenger(PassengerStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool InputCargo(CargoStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool OutputCargo(CargoStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool InputKamaz(KamazStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool OutputKamaz(KamazStruct\* ptr);

extern "C" \_\_declspec(dllexport) bool InputStr(wchar\_t\*, int length);

extern "C" \_\_declspec(dllexport) int InputInt();

extern "C" \_\_declspec(dllexport) bool OutputStr(wchar\_t\*, int length);

extern "C" \_\_declspec(dllexport) bool OutputInt(int num);

**Результаты** **работы программы**

Результаты работы программы показаны на рисунке 1.

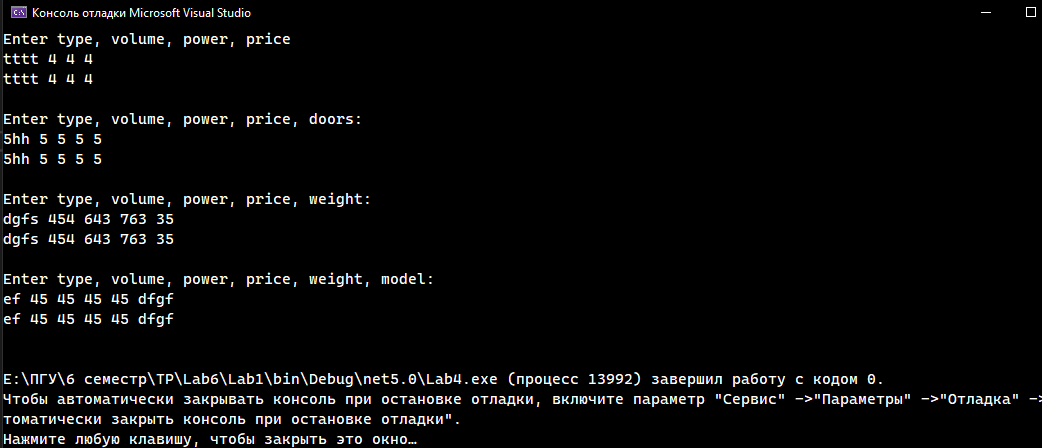


Рисунок 1 — Результаты работы программы

**Вывод**

Изучили механизм вызова функций из внешних библиотек. Получили навыки передачи параметров и возврата результата из функций внешних библиотек.