《**Windows**核心编程》实验报告

课程： Windows核心编程 实验名称：DLL编程

姓名： 杨佳伲 班级： 信安实验221

# 学号： 2022132006 实验报告日期： 2025/01/10

一、实验名称

# DLL编程

二、实验环境（详细说明运行的系统、平台及代码等）

1．操作系统：Windows；

2. 运行环境：vs2019。

## 三、实验目的

１、 掌握动态链接的两种不同方法，以及动态链接的不同加载方式基本原理

２、 采用api函数加载动态链接，LoadLibrary（），以及FreeLibrary（），

GetProcAddress（）函数的加载动态链接方式

３、 采用MFC和SDK方式编写动态链接，了解动态链接入口函数DllMain

４、 了解Hook编程基本概念，原理。采用动态链接中Hook编程

## 四、实验内容、步骤及结果

### １．实验内容

（1）编写一个win32动态链接库，实现两个数简单加法和减法，乘法，除法，然后采

用显式和隐式两种方式调用。调用界面采用MFC或者wind32窗口程序实现。

（2）编写一个钩子程序KeyBoardProc，在发生键盘事件时，记录用户的按键消息写入

到文件中，并且所有窗口不响应键盘事件（除F1键外）。

### ２．实验步骤

1. 编写一个win32动态链接库，实现两个数简单加法和减法，乘法，乘法，然后采用显式和隐式两种方式调用。

文件→新建→项目→win32项目，创建ArithmeticDll，应用程序类型选择DLL。

(2). 新建头文件ArithmeticDll.h

ArithmeticDll.h代码如下：

#pragma once

#ifndef ArithmeticDll\_H\_

#define ArithmeticDll\_H\_

#ifdef MYLIBDLL

#define MYLIBDLL Arithmetic "C" \_declspec(dllimport)

#else

#define MYLIBDLL extern "C" \_declspec(dllexport)

#endif

MYLIBDLL int Add(int plus1, int plus2);

MYLIBDLL int Sub(int sub1, int sub2);

MYLIBDLL int Mul(int mul1, int mul2);

MYLIBDLL int Div(int div1, int div2);

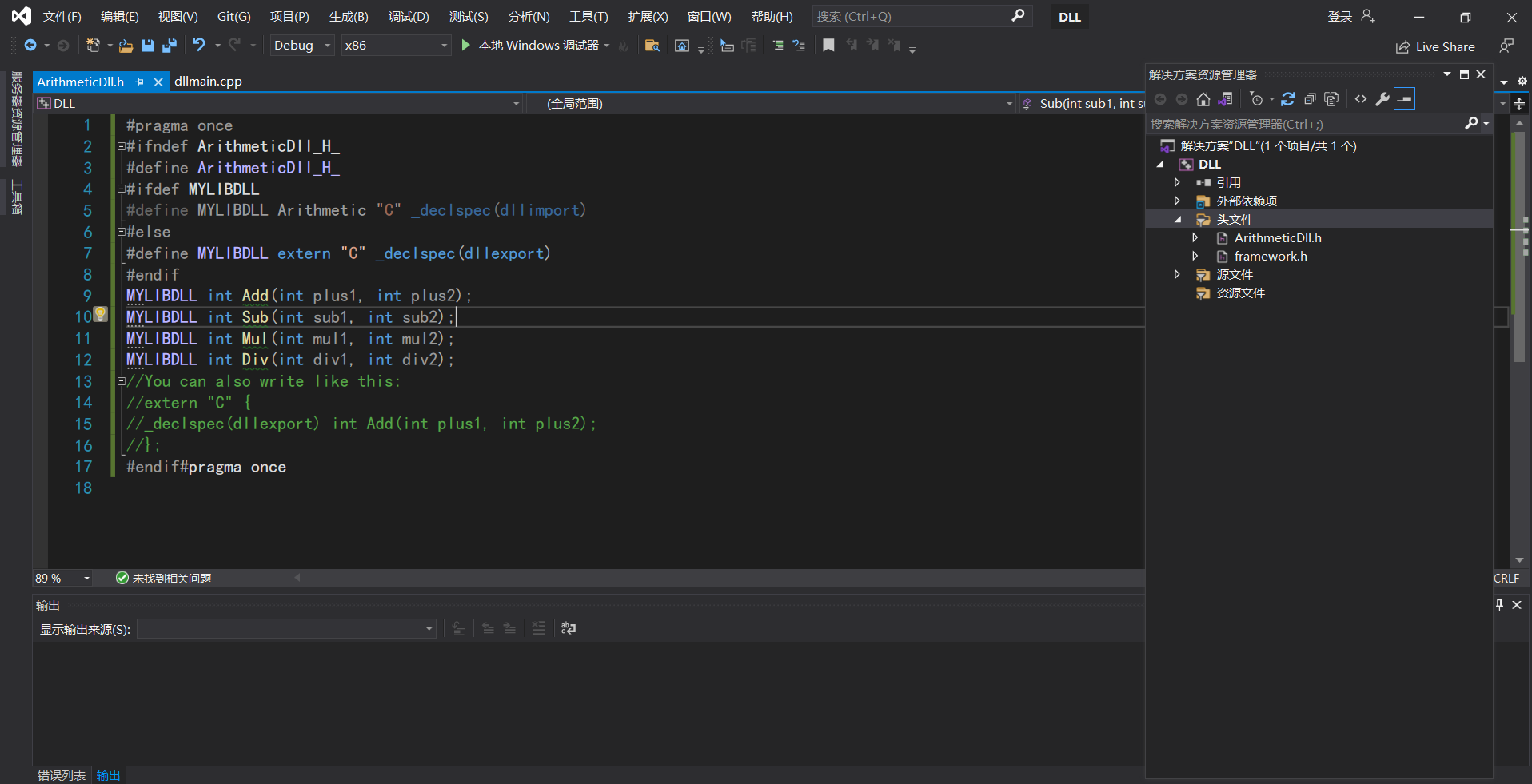
//You can also write like this:

//extern "C" {

//\_declspec(dllexport) int Add(int plus1, int plus2);

//};

#endif



(3).  源文件ArithmeticDLL.cpp的代码如下：

// ArithmeticDll.cpp : 定义 DLL 应用程序的导出函数。

//

#include "stdafx.h"

#include <iostream>

#include "ArithmeticDll.h"

using namespace std;

int Add(int plus1, int plus2)

{

int add\_result = plus1 + plus2;

return add\_result;

}

int Sub(int sub1, int sub2)

{

int sub\_result = sub1 - sub2;

return sub\_result;

}

int Mul(int mul1, int mul2)

{

int mul\_result = mul1\*mul2;

return mul\_result;

}

int Div(int div1, int div2)

{

if (div2 != 0)

{

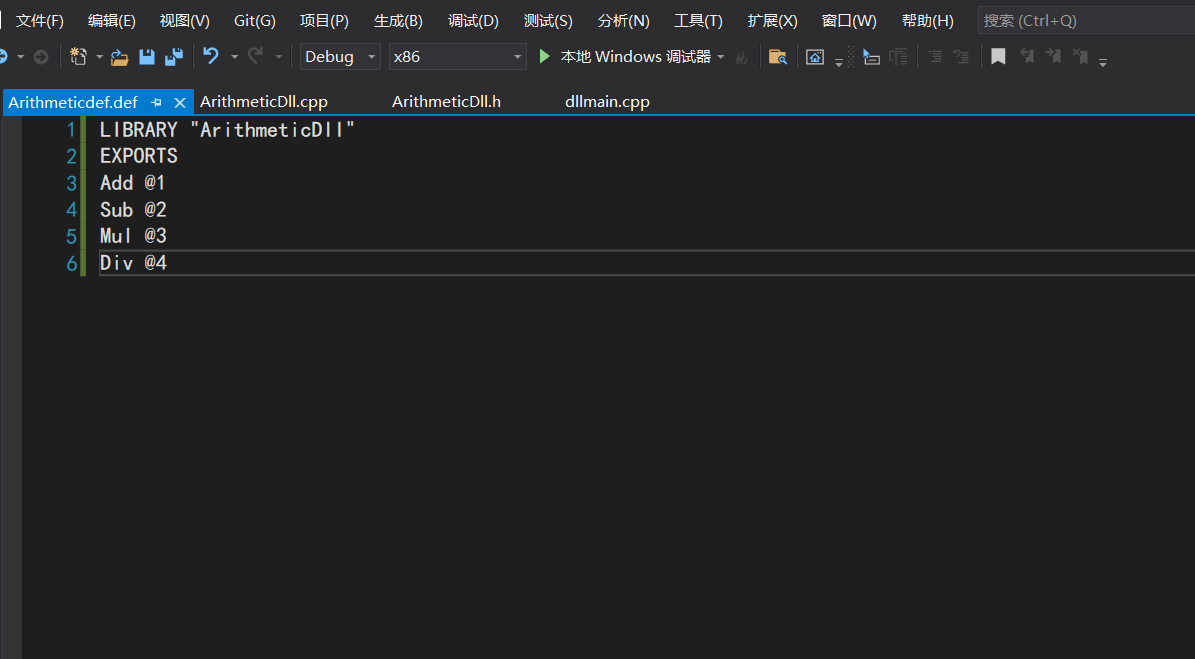
int div\_result = div1 / div2;

return div\_result;

}

}

1. .  新建模块定义文件Arithmeticdef.def



LIBRARY "ArithmeticDll"

EXPORTS

Add @1

Sub @2

Mul @3

Div @4

(5). 在vs2015自动创建的dllmain.cpp代码如下：

// dllmain.cpp : 定义 DLL 应用程序的入口点。

#include "stdafx.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:

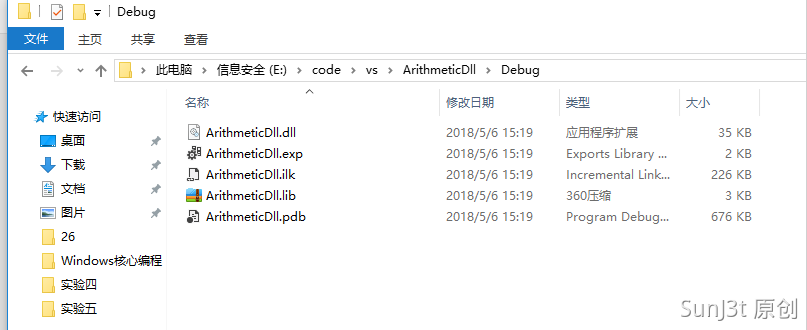
break;

}

return TRUE;

}

(6) . 最后，编译生成ArithmeticDLL.dll文件和ArithmeticDLL.lib文件



(7).  然后采用win32 或者MFC 程序中，调用动态链接库，显式和隐式两种方式调用，完成调用。

隐式加载动态链接库

// experience5.cpp : 定义控制台应用程序的入口点。

#include "stdafx.h"

#include "ArithmeticDll.h"

#pragma comment (lib,"ArithmeticDLL")

int main(int argc, \_TCHAR\* argv[])

{

printf("隐式加载动态链接库 Add: %d", Add(5, 10));

getchar();

return 0;

}

显式加载动态链接库

// experience5.cpp : 定义控制台应用程序的入口点。

//

#include "stdafx.h"

#include "ArithmeticDll.h"

#include <windows.h>

//#pragma comment (lib,"ArithmeticDLL")

int main(int argc, \_TCHAR\* argv[])

{

//printf("隐式加载动态链接库 Add: %d", Add(5, 10));

HINSTANCE hInst;

hInst = LoadLibrary(\_T("ArithmeticDll.dll "));

typedef int(\*ADDPROC)(int sub1, int sub2);

ADDPROC Sub = (ADDPROC)GetProcAddress(hInst, "Sub");

printf("显示调用动态链接库减函数 Sub：%d", Sub(5, 10));

getchar();

return 0;

}

1. 编写一个钩子程序，实现鼠标钩子和键盘钩子。当测试程序运行时，对话框窗口放到最大，鼠标不可用。只有在发生键盘F2按键按下事件发生时，则对话框测试应用程序关闭。鼠标重新可用。

////////////////////////////////////////////////

// KeyHookApp.cpp文件

#include "resource.h"

#include "KeyHookApp.h"

#include "KeyHookLib.h"

#pragma comment(lib, "09KeyHookLib")

CMyApp theApp;

BOOL CMyApp::InitInstance()

{

CMainDialog dlg;

m\_pMainWnd = &dlg;

dlg.DoModal();

return FALSE;

}

CMainDialog::CMainDialog(CWnd\* pParentWnd):CDialog(IDD\_MAIN, pParentWnd)

{

}

BEGIN\_MESSAGE\_MAP(CMainDialog, CDialog)

ON\_MESSAGE(HM\_KEY, OnHookKey)

END\_MESSAGE\_MAP()

BOOL CMainDialog::OnInitDialog()

{

CDialog::OnInitDialog();

SetIcon(theApp.LoadIcon(IDI\_MAIN), FALSE);

::SetWindowPos(m\_hWnd, HWND\_TOPMOST, 0, 0,

0, 0, SWP\_NOSIZE|SWP\_NOREDRAW|SWP\_NOMOVE);

if(!SetKeyHook(TRUE, 0, m\_hWnd)) // 安装钩子

MessageBox("安装钩子失败！");

return TRUE;

}

void CMainDialog::OnCancel()

{

SetKeyHook(FALSE); // 卸载钩子

CDialog::OnCancel();

return;

}

long CMainDialog::OnHookKey(WPARAM wParam, LPARAM lParam)

{

// 此时参数wParam为用户按键的虚拟键码，

// lParam参数包含按键的重复次数、扫描码、前一个按键状态等信息

char szKey[80];

::GetKeyNameText(lParam, szKey, 80);

CString strItem;

strItem.Format(" 用户按键：%s \r\n", szKey);

CString strEdit; // 添加到编辑框中

GetDlgItem(IDC\_KEYMSG)->GetWindowText(strEdit);

GetDlgItem(IDC\_KEYMSG)->SetWindowText(strItem + strEdit);

::MessageBeep(MB\_OK);

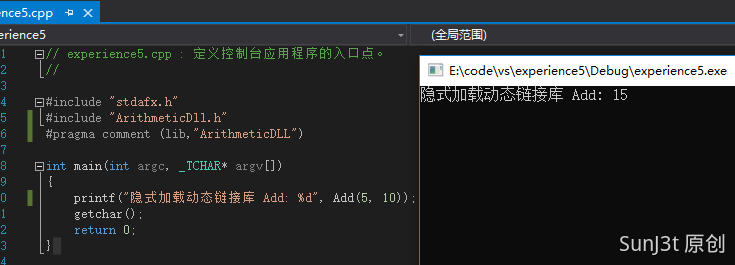
return 0;

}

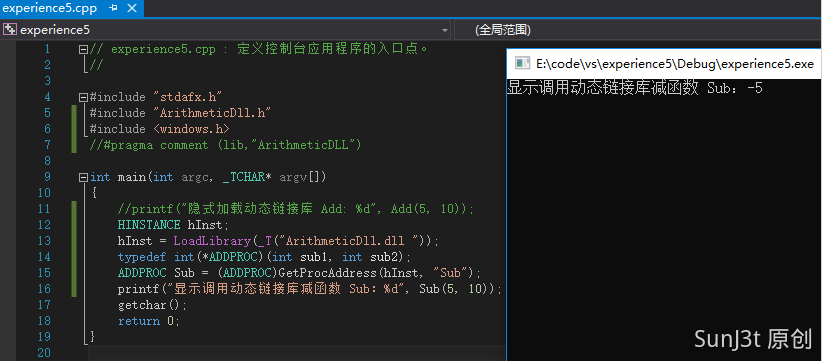
### 3. 实验结果(实验运行截图)

编写一个win32动态链接库，实现两个数简单加法和减法，乘法，乘法，然后采用显式和隐式两种方式调用。

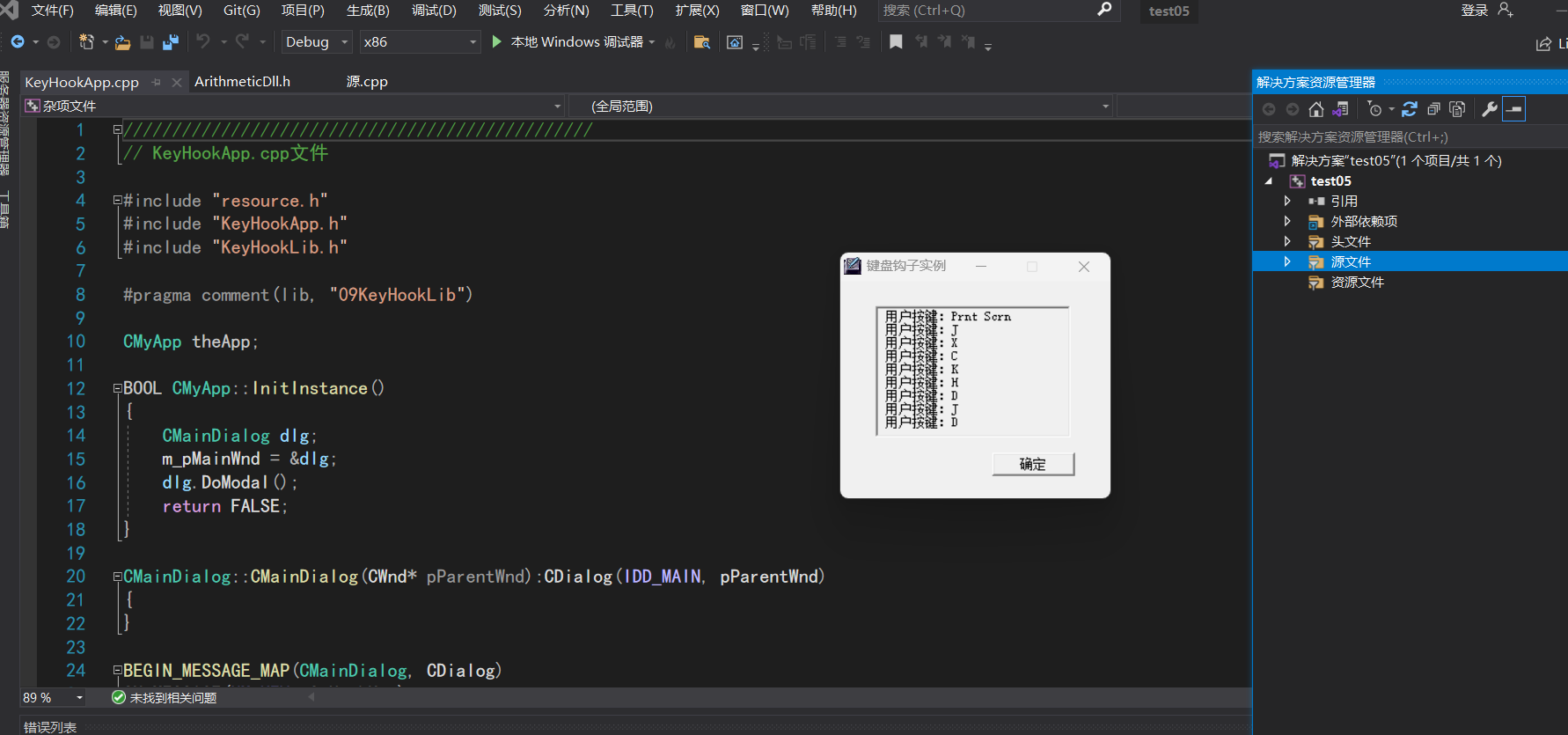
隐式加载动态链接库



显式加载动态链接库



编写一个钩子程序，实现鼠标钩子和键盘钩子。当测试程序运行时，对话框窗口放到最大，鼠标不可用。只有在发生键盘F2按键按下事件发生时，则对话框测试应用程序关闭。鼠标重新可用。



五、实验中的问题及解决办法。

学会了使用动态链接库以及如何使用显式和隐式来调用。

隐式调用一般使用program comment，而显式调用一般使用LoadLibrary函数。

2