**山东师范大学**

**实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 院系部所 | 信息科学与工程学院 | | 课程名称 | | 计算机操作系统 |
| 所在班级 | 计工本1702 | 学号 | 201711010202 | 姓名 | 王汝芸 |
| 实验编号 | 02 | 题目 | 线程的同步 | | |

一、实验目的和要求

（1）进一步掌握Windows系统环境下线程的创建与撤销。

（2）熟悉Windows系统提供的线程同步API。

（3）使用Windows系统提供的线程同步API解决实际问题。

二、实验环境

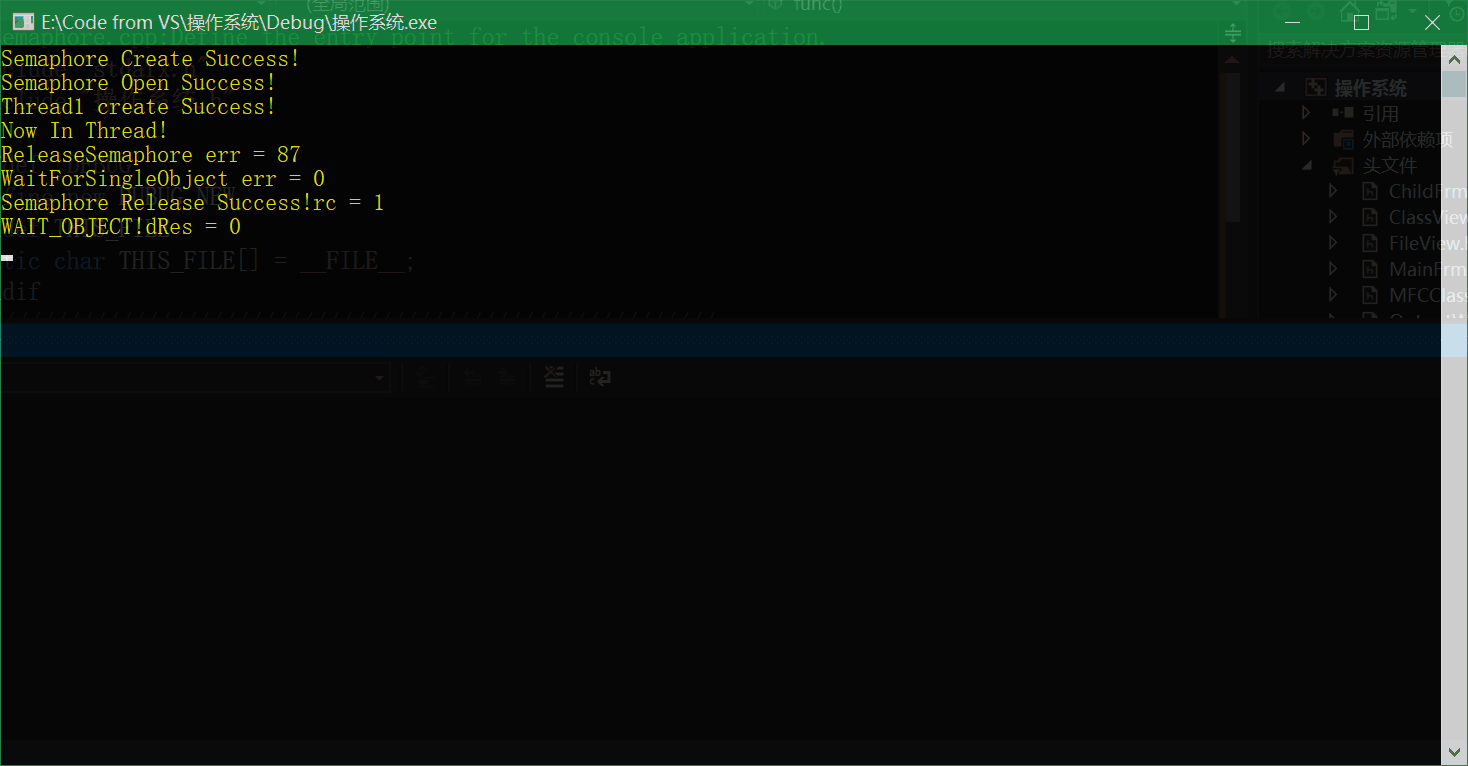
Windows 10

Visual Studio2017

三、实验内容及实施

完成主、子两个线程之间的同步，要求子线程先执行。在主线程中使用系统调用CreateThread()创建一个子线程。主线程创建子线程后进入阻塞状态，直到子线程运行完毕后唤醒主线程。

四、实验结果



五、实验分析

该实验完成了主、子两个线程的同步，主线程创建子线程后，主线程阻塞，让子线程先执行，等子线程执行完毕后，由子线程唤醒主线程。

六、实验讨论（可选）

解决了上一节课VS中使用MFC不弹出控制台的问题，只需要更改VS中子系统-控制台即可

七、附录 （可选）

//Semaphore.cpp:Define the entry point for the console application.

#include "stdafx.h"

#include "操作系统.h"

#ifdef \_DEBUG

#define new DUBUG\_NEW

#undef THIS\_FILE

static char THIS\_FILE[] = \_\_FILE\_\_;

#endif

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

//CWinApp theApp;

using namespace std;

static HANDLE h1;

static HANDLE hHandle1 = NULL;

void func();

int \_tmain(int argc, TCHAR\*atgv[], TCHAR\*envp[])

{

int nRetCode = 0;

DWORD dwThreadID1;

DWORD dRes, err;

hHandle1 = CreateSemaphore(NULL, 0, 1, LPCWSTR("SemaphoreName1")); //创建一个信号量

if (hHandle1 == NULL)printf("Semaphore Create Fail!\n");

else printf("Semaphore Create Success!\n");

hHandle1 = OpenSemaphore(SYNCHRONIZE | SEMAPHORE\_MODIFY\_STATE,

NULL,

LPCWSTR("SemaphoreName1"));

if (hHandle1 == NULL)printf("Semaphore Open Fail!\n");

else printf("Semaphore Open Success!\n");

h1 = CreateThread((LPSECURITY\_ATTRIBUTES)NULL,

0,

(LPTHREAD\_START\_ROUTINE)func,

(LPVOID)NULL,

0, &dwThreadID1);

if (h1 == NULL)printf("Thread1 create Fail!\n");

else printf("Thread1 create Success!\n");

dRes = WaitForSingleObject(hHandle1, INFINITE);

err = GetLastError();

printf("WaitForSingleObject err = %d\n", err);

if (dRes == WAIT\_TIMEOUT)printf("TIMEOUT!dRes = %d\n", dRes);

else if (dRes == WAIT\_OBJECT\_0)printf("WAIT\_OBJECT!dRes = %d\n", dRes);

else if (dRes == WAIT\_ABANDONED)

printf("WAIT\_ABANDONED!dRes = %d\n", dRes);

else printf("dRes = %d\n", dRes);

CloseHandle(h1);

CloseHandle(hHandle1);

ExitThread(0);

return nRetCode;

}

void func()

{

BOOL rc;

DWORD err;

printf("Now In Thread!\n");

rc = ReleaseSemaphore(hHandle1, 1, NULL);

err = GetLastError();

printf("ReleaseSemaphore err = %d\n", err);

if (rc == 0)printf("Semaphore Release Fail!\n");

else printf("Semaphore Release Success!rc = %d\n", rc);

}