Университет ИТМО

Кафедра ИПМ

Системное программное обеспечение

Лабораторная работа № 6

Работу выполнил:

Студент 2 курса

Группы № 2125

Назарьев Сергей

Санкт-Петербург

2015 г.

**Цель работы:**

Реализовать примеры работы 4 основных способов синхронизации потоков в среде ОС Windows

**Код программы:**

#include <windows.h>

#include <stdio.h>

#define MAX\_THREADS 2

HANDLE semaphore\_gl;

DWORD WINAPI **ThreadRoutine**(LPVOID lpArg)

{

 **while**(**1**) {

 WaitForSingleObject(semaphore\_gl, INFINITE);

 printf("Thread %d: A**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: B**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: C**\n**", (**int**)lpArg);

 ReleaseSemaphore(semaphore\_gl, **1**, NULL);

 }

 **return** **0**;

}

**int** **main**()

{

 DWORD ThreadId;

 HANDLE hHandles[MAX\_THREADS];

 semaphore\_gl = CreateSemaphore(

 NULL,

 MAX\_THREADS,

 MAX\_THREADS,

 NULL);

 **if**( semaphore\_gl == NULL ) {

 fprintf( stderr, "OH SHI~. Semaphore can't be created." );

 **return** -**1**;

 }

 **int** i;

 **for**(i = **0**; i < MAX\_THREADS; i++)

 hHandles[i] = CreateThread(NULL,**0**,ThreadRoutine,(LPVOID)i,**0**,&ThreadId);

 WaitForMultipleObjects(MAX\_THREADS, hHandles, **1**, INFINITE);

 **return** **0**;

}

#include <windows.h>

#include <stdio.h>

#define MAX\_THREADS 2

CRITICAL\_SECTION gCS;

DWORD WINAPI **ThreadRoutine**(LPVOID lpArg)

{

 **while**(**1**) {

 EnterCriticalSection(&gCS);

 printf("Thread %d: A**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: B**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: C**\n**", (**int**)lpArg);

 LeaveCriticalSection(&gCS);

 }

 **return** **0**;

}

**int** **main**()

{

 DWORD ThreadId;

 HANDLE hHandles[MAX\_THREADS];

 InitializeCriticalSection(&gCS);

 **int** i;

 **for**(i = **0**; i < MAX\_THREADS; i++)

 hHandles[i] = CreateThread(NULL,**0**,ThreadRoutine,(LPVOID)i,**0**,&ThreadId);

 WaitForMultipleObjects(MAX\_THREADS, hHandles, **1**, INFINITE);

 **return** **0**;

}

#include <windows.h>

#include <stdio.h>

#define MAX\_THREADS 2

HANDLE event;

DWORD WINAPI **ThreadRoutine**(LPVOID lpArg)

{

 **while**(**1**) {

 WaitForSingleObject(

 event,

 INFINITE

 );

 ResetEvent( event );

 printf("Thread %d: A**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: B**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: C**\n**", (**int**)lpArg);

 SetEvent( event );

 }

 **return** **0**;

}

**int** **main**()

{

 DWORD ThreadId;

 HANDLE hHandles[MAX\_THREADS];

 event = CreateEvent(

 NULL,

 TRUE,

 TRUE,

 TEXT("Multiplication")

 );

 **int** i;

 **for**(i = **0**; i < MAX\_THREADS; i++)

 hHandles[i] = CreateThread(NULL,**0**,ThreadRoutine,(LPVOID)i,**0**,&ThreadId);

 WaitForMultipleObjects(**2**, hHandles, **1**, INFINITE);

 **return** **0**

}

#include <windows.h>

#include <stdio.h>

#define MAX\_THREADS 2

HANDLE mutex\_gl;

DWORD WINAPI **ThreadRoutine**(LPVOID lpArg)

{

 **while**(**1**) {

 WaitForSingleObject(mutex\_gl, INFINITE);

 printf("Thread %d: A**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: B**\n**", (**int**)lpArg);

 sleep(**1**);

 printf("Thread %d: C**\n**", (**int**)lpArg);

 ReleaseMutex(mutex\_gl);

 }

 **return** **0**;

}

**int** **main**()

{

 DWORD ThreadId;

 HANDLE hHandles[MAX\_THREADS];

 mutex\_gl = CreateMutex(NULL, FALSE, NULL);

 **if**( mutex\_gl == NULL ) {

 fprintf( stderr, "OH SHI~. Mutex can't be created." );

 **return** -**1**;

 }

 **int** i;

 **for**(i = **0**; i < MAX\_THREADS; i++)

 hHandles[i] = CreateThread(NULL,**0**,ThreadRoutine,(LPVOID)i,**0**,&ThreadId);

 WaitForMultipleObjects(MAX\_THREADS, hHandles, **1**, INFINITE);

 **return** **0**;

}

**Вывод:**

В ходе лабораторной работы были изучены WinAPI-функции для реализации 4 способов синхронизации потоков.