Министерство науки и высшего образования Российской Федерации  
Пензенский государственный университет  
Кафедра вычислительная техника

**ОТЧЕТ**

по лабораторной работе №2

по дисциплине «Основы операционных систем»

на тему «Синхронизация нитей с использованием событий и

Interlocked-функций»

Выполнили студенты группы 22ВВП1

Хоссейни Нежад С.А.С.М.

Захаров А. С.

Приняли:

Дубинин В.Н.

Карамышева Н.С.

Пенза 2024

**Название**

Синхронизация нитей с использованием событий и Interlocked-функций

**Цель работы**

Изучение способа организации критических секций с использованием

группы Interlocked-функций и структур CRITICAL\_SECTION. Изучение

способа синхронизации процессов и нитей с использованием событий

(Events).

**Лабораторное задание**

Имеется 3 нити. Две из них производят инкремент переменной на 1.

Одна – декремент на 2. Работа с переменной должна осуществляться

с помощью событий с автоматическим сбросом. Вывести на экран текущее значение переменной и крайние значения.

**Описание программы**

В начале создаем событие с начальным несигнальным состоянием и переменную равную 0. Создаем 3 нити, в первых 2 при создании передаем функцию с инкрементом 1, а в 3 с декрементом 2, в каждую нить передаем параметром переменную. В каждой нити в конце добавили функцию SetEvent(), чтобы установить объект события в сигнальное состояние. Далее в основной нити устанавливаем объект события в сигнальное состояние, что разблокирует одну нить, так как используем автоматический сброс. В конце ждем завершения всех нитей функцией WaitForMultipleObjects().

**Текст программы**

#include <windows.h>

#include <stdio.h>

#include <locale.h>

#define THREAD\_COUNT 3

HANDLE hEvent;

DWORD WINAPI threadFunctionInc(LPVOID lpNumber)

{

int\* number = (int\*)lpNumber;

WaitForSingleObject(hEvent, INFINITE);

int threadId = GetCurrentThreadId();

printf("Нить %d запущена\n", threadId);

int b = 1;

\*number = \*number + b;

printf("a = a + %d = %d\n", b, \*number);

printf("Нить %d завершилась\n\n", threadId);

SetEvent(hEvent);

return 0;

}

DWORD WINAPI threadFunctionDec(LPVOID lpNumber)

{

int\* number = (int\*)lpNumber;

WaitForSingleObject(hEvent, INFINITE);

int threadId = GetCurrentThreadId();

printf("Нить %d запущена\n", threadId);

int b = 2;

\*number = \*number - b;

printf("a = a - %d = %d\n", b, \*number);

printf("Нить %d завершилась\n\n", threadId);

SetEvent(hEvent);

return 0;

}

int main()

{

setlocale(LC\_ALL, "Rus");

HANDLE hThreads[THREAD\_COUNT];

int number = 0;

printf("a = %d\n\n", number);

hEvent = CreateEvent(NULL, FALSE, FALSE, NULL);

if (hEvent == NULL) {

printf("Событие не создано (%d)\n", GetLastError());

return 1;

}

for (int i = 0; i < THREAD\_COUNT - 1; i++)

{

hThreads[i] = CreateThread(NULL, 0, threadFunctionInc, &number, NULL, NULL);

if (hThreads[i] == NULL)

{

printf("Не удалось создать нить %d\n", i + 1);

return 1;

}

}

hThreads[THREAD\_COUNT - 1] = CreateThread(NULL, 0, threadFunctionDec, &number, NULL, NULL);

SetEvent(hEvent);

WaitForMultipleObjects(THREAD\_COUNT, hThreads, TRUE, INFINITE);

printf("a = %d\n", number);

printf("Все нити завершены.\n");

CloseHandle(hEvent);

for (int i = 0; i < THREAD\_COUNT; i++)

{

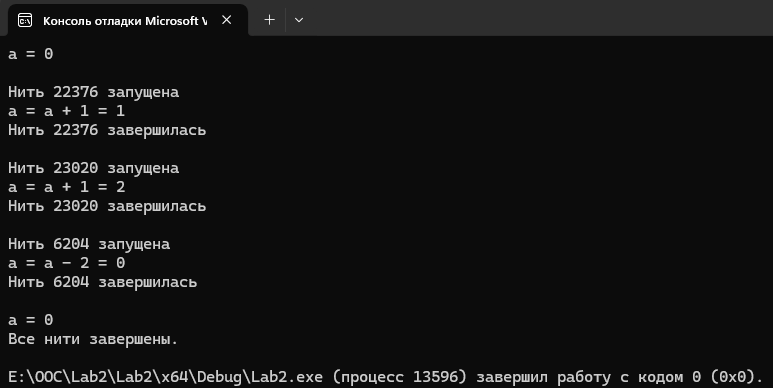
CloseHandle(hThreads[i]);

}

return 0;

}

**Результат работы программы**

****

**Вывод**

Изучили способ организации критических секций с использованием

группы Interlocked-функций и структур CRITICAL\_SECTION. Изучили

способ синхронизации процессов и нитей с использованием событий (Events).