Университет ИТМО

Кафедра ИПМ

Операционные системы

Лабораторная работа № 2

Вариант 7

Работу выполнил

cтудент гр. P3315

Назарьев Сергей

Санкт-Петербург

2015 г.

Поток main должен выполнить следующие действия:

* создать массив, размерность и элементы которого вводятся пользователем с консоли
* вывести размерность и элементы исходного массива на консоль
* ввести число k
* запустить поток work
* запустить поток MultElement
* известить поток MultElement о начале работы (момент запуска произойдёт после того, будут выведены на консоль k элементов массива).

Поток work должен выполнить следующие действия (для синхронизации с потоком main – использовать семафор):

* запросить у пользователя временной интервал, требуемый для отдыха после подготовки одного элемента в массиве
* Поиск в массиве элементов < A (разместить их в массиве слева, остальные элементы массива — справа)
* элементы - вещественные числа
* число A ввести в потоке main
* после каждого готового элемента отдыхать в течение заданного интервала времени

Поток MultElement должен выполнить следующие действия (для синхронизации с потоком main использовать бинарный семафор):

* ждёт от потока main сигнал о начале работы
* выполнить произведение элементов итогового массива до заданной позиции k
* вывести итоговое произведение

#include <windows.h>

#include <iostream>

using namespace std;

HANDLE hMultSemaphore = CreateSemaphore(NULL, 0, 1, NULL);

HANDLE hWorkSemaphore;

struct ArrStruct

{

volatile int \*arr;

int n;

};

struct ArrStruct2

{

volatile int \*arr;

int n;

int a;

int rest;

};

DWORD WINAPI work(ArrStruct2 \*arrStr) {

int \*left = new int[arrStr->n];

int \*right = new int[arrStr->n];

int l = 0, r = 0;

for (int i = 0; i < arrStr->n; i++) {

if (arrStr->arr[i] < arrStr->a)

left[l++] = arrStr->arr[i];

else

right[r++] = arrStr->arr[i];

ReleaseSemaphore(hWorkSemaphore, 1, NULL);

Sleep(arrStr->rest \* 1000);

}

for (int i = 0; i < arrStr->n; i++)

if( i < l )

arrStr->arr[i] = left[i];

else

arrStr->arr[i] = right[i-l];

ReleaseSemaphore(hWorkSemaphore, 1, NULL);

return 0;

}

DWORD WINAPI mult(ArrStruct \*arrStr) {

WaitForSingleObject(hMultSemaphore, INFINITE);

int mult = 1;

for (int i = 0; i < arrStr->n; i++)

mult \*= arrStr->arr[i];

cout << endl << "Multiplication = " << mult << endl;

return 0;

}

int main() {

HANDLE hWork, hMult;

DWORD IDWork, IDMult;

int k;

volatile int \*arr;

int rest;

int a;

int n;

cout << "Please enter k: ";

cin >> k;

cout << "Please enter n: ";

cin >> n;

arr = new int[n];

cout << "Please enter " << n << " elements of array divided by space: ";

for (int i = 0; i < n; i++) {

int b;

cin >> b;

arr[i] = b;

}

cout << "Please enter rest time: ";

cin >> rest;

cout << "Please enter a: ";

cin >> a;

cout << "Source array: " << endl;

for (int i = 0; i < n; i++)

cout << arr[i] << " ";

cout << endl;

hWorkSemaphore = CreateSemaphore(NULL, 0, n+1, NULL);

ArrStruct arrStr;

arrStr.n = k;

arrStr.arr = arr;

ArrStruct2 arrStr2;

arrStr2.n = n;

arrStr2.arr = arr;

arrStr2.a = a;

arrStr2.rest = rest;

hWork = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)work, &arrStr2, 0, &IDWork);

hMult = CreateThread(NULL, 0, (LPTHREAD\_START\_ROUTINE)mult, &arrStr, 0, &IDMult);

for (int i = 0; i <= n; i++)

WaitForSingleObject(hWorkSemaphore, INFINITE);

cout << "New array: " << endl;

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

ReleaseSemaphore(hMultSemaphore, 1, NULL);

CloseHandle(hMultSemaphore);

CloseHandle(hWorkSemaphore);

CloseHandle(hMult);

CloseHandle(hWork);

delete[] arr;

system("pause");

return 0;

}