// lab3.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <queue>

#include <iostream>

#include <random>

#include <Windows.h>

using namespace std;

queue<int> numbers;

const int NUMBER\_COUNT = 10;

volatile bool GeneratorCompleted = FALSE;

// предварительное объявление функций

void GenerateNumbers();

void ProcessNumbers(int\*);

CRITICAL\_SECTION cs;

DWORD WINAPI GeneratorThread(LPVOID context)

{

GenerateNumbers();

return 0;

}

DWORD WINAPI ConsumerThread(LPVOID context)

{

int sum = 0;

ProcessNumbers(&sum);

return 0;

}

// Точка входа

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

{

InitializeCriticalSection(&cs);

HANDLE hThreads[3];

hThreads[0] = CreateThread(NULL, NULL, GeneratorThread, 0, 0, 0);

hThreads[1] = CreateThread(NULL, NULL, ConsumerThread, 0, 0, 0);

hThreads[2] = CreateThread(NULL, NULL, ConsumerThread, 0, 0, 0);

WaitForMultipleObjects(3, hThreads, TRUE, INFINITE);

// сброс генератора случайных чисел (при необходимости).

//srand(200);

return 0;

}

void GenerateNumbers()

{

for (size\_t i = 0; i < NUMBER\_COUNT; i++)

{

EnterCriticalSection(&cs);

// помещаю в очередь случайное число 0-99

numbers.push(rand()%100);

cout << "Generated number " << numbers.back() << endl;

// жду две секунды (имитация долгой работы)

LeaveCriticalSection(&cs);

Sleep(200);

}

GeneratorCompleted = TRUE;

}

void ProcessNumbers(int\* sumAddress)

{

while (!GeneratorCompleted)

{

EnterCriticalSection(&cs);

if (numbers.size() != 0)

{

// прочитал очередной элемент из очереди

int current = numbers.front();

cout << "Processing element " << current << endl;

// удалил его из очереди

numbers.pop();

// прибавил к переменной, накапливающей сумму

\*sumAddress += current;

// жду две секунды (имитация долгой работы)

}

LeaveCriticalSection(&cs);

Sleep(200);

}

}

// Подсказка о том, как выглядит функция - тело потока

/\*DWORD WINAPI ThreadBody(LPVOID context)

{

return 0;

}\*/