**ЗАДАНИЕ 14**

Задание 1. Создать консольное приложение, в котором реализовано потока. Первый выводит числа от 0 до 9, второй – от 10 до 19, третий – от 20 до 29. Вывод значений на экран происходит следующим образом: сначала своё число выводит первый поток, а затем второй и третий. Организовать возможность использования методов Start и Sleep, а также изменение свойства Priority, позволяющее изменить приоритет потока. Можно воспользоваться инструментами синхронизации потоков lock, AutoResetEvent и т.д.

Листинг программы:

using System;

using System.Threading;

class Program

{

static AutoResetEvent firstEvent = new AutoResetEvent(false);

static AutoResetEvent secondEvent = new AutoResetEvent(false);

static void Main()

{

Thread t1 = new Thread(PrintNumbers);

Thread t2 = new Thread(PrintNumbers);

Thread t3 = new Thread(PrintNumbers);

t1.Name = "Поток 1";

t2.Name = "Поток 2";

t3.Name = "Поток 3";

t1.Start(0);

t2.Start(10);

t3.Start(20);

firstEvent.Set();

Console.ReadLine();

}

static void PrintNumbers(object startingNumber)

{

int number = (int)startingNumber;

for (int i = 0; i < 10; i++)

{

firstEvent.WaitOne();

Console.WriteLine($"{Thread.CurrentThread.Name}: {number}");

secondEvent.Set();

number++;

}

}

}

Таблица 14.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Поток 1, Поток 2, Поток 3 | Поток 3: 20 |

Анализ результатов:

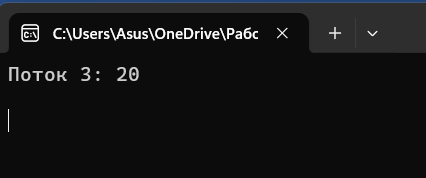


Рисунок 14.1 – Результат работы программы

Задание 2. Создать консольное приложение, в котором несколько потоков будут выполнять один и тот же метод. (Количество потоков: 2, Метод: Сумма чисел от 1 до 10). Произвести расчет затраченного времени в миллисекундах на выполнение потока и вывести его на экран.

Листинг программы:

using System;

using System.Threading;

class Program

{

static int SumNumbers()

{

int sum = 0;

for (int i = 1; i <= 10; i++)

{

sum += i;

}

return sum;

}

static void Main()

{

Thread t1 = new Thread(() =>

{

var watch = System.Diagnostics.Stopwatch.StartNew();

int result = SumNumbers();

watch.Stop();

Console.WriteLine($"Поток 1: Сумма чисел от 1 до 10 равна {result}. Время выполнения: {watch.ElapsedMilliseconds} мс");

});

Thread t2 = new Thread(() =>

{

var watch = System.Diagnostics.Stopwatch.StartNew();

int result = SumNumbers();

watch.Stop();

Console.WriteLine($"Поток 2: Сумма чисел от 1 до 10 равна {result}. Время выполнения: {watch.ElapsedMilliseconds} мс");

});

t1.Start();

t2.Start();

Console.ReadLine();

}

}

Таблица 14.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Числа от 1 до 10 | 55 |

Анализ результатов:

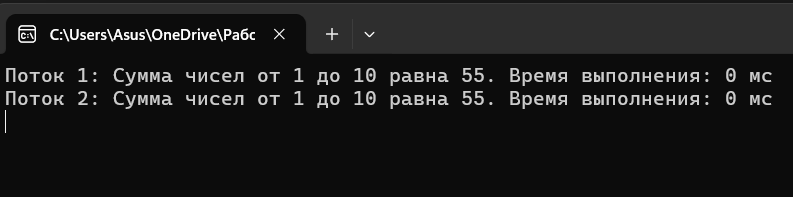


Рисунок 14.2 – Результат работы программы

Задание 3. Создать консольное приложение, в котором будут два метода, один метод будет выполняться двумя потоками одновременно, а другой метод в каждый момент времени будет выполняться одним потоком. (Количество потоков: 2, Метод: 1. A+A 1 +A 2 +A 3 +..+A N , А и N вводятся с клавиатуры; 2. A\*A 1 \*A 2 \*A 3 \*…\*A N , А и N вводятся с клавиатуры ).

Листинг программы:

using System.Threading;

class Program

{

static int CalculateSum(int A, int N)

{

int sum = 0;

for (int i = 1; i <= N; i++)

{

sum += A;

}

return sum;

}

static int CalculateProduct(int A, int N)

{

int product = 1;

for (int i = 1; i <= N; i++)

{

product \*= A;

}

return product;

}

static void Main()

{

Console.Write("Введите число A: ");

int A = int.Parse(Console.ReadLine());

Console.Write("Введите число N: ");

int N = int.Parse(Console.ReadLine());

Thread t1 = new Thread(() =>

{

int sum = CalculateSum(A, N);

Console.WriteLine($"Сумма: {sum}");

});

Thread t2 = new Thread(() =>

{

int product = CalculateProduct(A, N);

Console.WriteLine($"Произведение: {product}");

});

t1.Start();

t2.Start();

t1.Join();

t2.Join();

Console.ReadLine();

}

}

Таблица 14.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 4, 7 | 28, 16384 |

Анализ результатов:

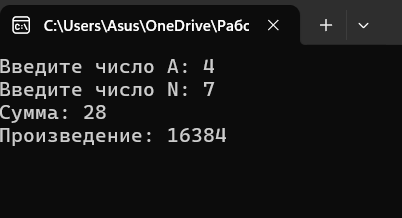


Рисунок 14.3 – Результат работы программы

Задание 4. Дана последовательность натуральных чисел {a 0 …a n–1 }. Создать многопоточное приложение для поиска суммы ∑a i , где a i – четные числа.

Листинг программы:

using System;

using System.Threading;

class Program

{

static int[] numbers = { 5, 12, 8, 23, 6, 10, 15, 18 }; // Пример массива чисел

static int CalculatePartialSum(object data)

{

int partialSum = 0;

int startIndex = (int)data;

for (int i = startIndex; i < numbers.Length; i++)

{

if (numbers[i] % 2 == 0)

{

partialSum += numbers[i];

}

}

return partialSum;

}

static void Main()

{

int numThreads = 4; // Количество потоков

int totalSum = 0;

Thread[] threads = new Thread[numThreads];

int[] partialSums = new int[numThreads];

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

{

int index = i;

threads[i] = new Thread(() =>

{

partialSums[index] = CalculatePartialSum(index);

});

threads[i].Start();

}

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

{

threads[i].Join();

totalSum += partialSums[i];

}

Console.WriteLine($"Общая сумма четных чисел: {totalSum}");

Console.ReadLine();

}

}

Таблица 14.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Массив чисел | 184 |

Анализ результатов:

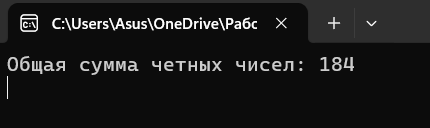


Рисунок 14.4 – Результат работы программы