**14 МНОГОПОТОЧНЫЕ ПРИЛОЖЕНИЯ**

Задание 1. Создать консольное приложение, в котором реализовано 3 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.

Листинг программы:

class Program

{

static void mythread1()

{

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

{

Console.WriteLine("Поток 1 выводит " + i);

}

}

static void mythread2()

{

for (int i = 10; i < 20; i++)

{

Console.WriteLine("Поток 2 выводит " + i);

}

}

static void mythread3()

{

for (int i = 20; i < 30; i++)

{

Console.WriteLine("Поток 3 выводит " + i);

}

}

static void Main(string[] args)

{

Thread thread1 = new Thread(mythread1);

Thread thread2 = new Thread(mythread2);

Thread thread3 = new Thread(mythread3);

thread1.Priority = ThreadPriority.Highest;

thread2.Priority = ThreadPriority.AboveNormal;

thread3.Priority = ThreadPriority.Lowest;

thread1.Start();

Thread.Sleep(10); ;

thread2.Start();

Thread.Sleep(20);

thread3.Start();

Thread.Sleep(30);

Console.ReadLine();

}

Таблица 14.1 – Выходные данные

|  |
| --- |
| Выходные данные |
| Поток 1 выводит 1  Поток 1 выводит 2  Поток 1 выводит 3  Поток 1 выводит 4  Поток 1 выводит 5  Поток 1 выводит 6  Поток 1 выводит 7  Поток 1 выводит 8  Поток 1 выводит 9  Поток 2 выводит 10  Поток 2 выводит 11  Поток 2 выводит 12  Поток 2 выводит 13  Поток 2 выводит 14  Поток 2 выводит 15  Поток 2 выводит 16  Поток 2 выводит 17  Поток 2 выводит 18  Поток 2 выводит 19 |

Окончание таблицы 14.1

|  |
| --- |
| Выходные данные |
| Поток 3 выводит 20  Поток 3 выводит 21  Поток 3 выводит 22  Поток 3 выводит 23  Поток 3 выводит 24  Поток 3 выводит 25  Поток 3 выводит 26  Поток 3 выводит 27  Поток 3 выводит 28  Поток 3 выводит 29 |

Анализ результатов:

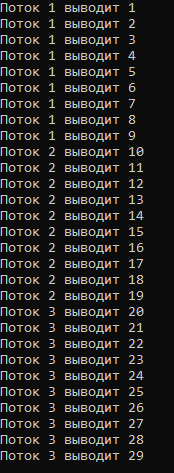


Рисунок 14.1 – Результат работы программы

Источник: собственная разработка

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:

internal class Program

{

static void mythread1()

{

int sum = 0;

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

{

sum += i;

Console.WriteLine("Поток 1 выводит " + sum);

}

}

static void mythread2()

{

int sum1 = 0;

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

{

sum1 += i;

Console.WriteLine("Поток 2 выводит " + sum1);

}

}

static void Main(string[] args)

{

Stopwatch sWatch = new Stopwatch();

Thread thread1 = new Thread(mythread1);

Thread thread2 = new Thread(mythread2);

thread1.Priority = ThreadPriority.Highest;

thread2.Priority = ThreadPriority.AboveNormal;

sWatch.Start();

thread1.Start();

Thread.Sleep(1000);

sWatch.Stop();

thread2.Start();

Console.WriteLine("Время (в мс.) затраченное на первый поток = " + sWatch.ElapsedMilliseconds.ToString());

Console.ReadLine(); ;

}

Таблица 14.2 – Входные и выходные данные

|  |
| --- |
| Выходные данные |
| Поток 1 выводит 0  Поток 1 выводит 1  Поток 1 выводит 3  Поток 1 выводит 6  Поток 1 выводит 10  Поток 1 выводит 15  Поток 1 выводит 21  Поток 1 выводит 28  Поток 1 выводит 36  Поток 1 выводит 45  Время (в мс.) затраченное на первый поток = 1001  Поток 2 выводит 0  Поток 2 выводит 1  Поток 2 выводит 3  Поток 2 выводит 6  Поток 2 выводит 10  Поток 2 выводит 15  Поток 2 выводит 21  Поток 2 выводит 28  Поток 2 выводит 36  Поток 2 выводит 45 |

Анализ результатов:

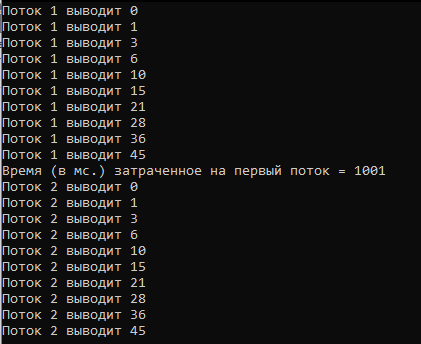


Рисунок 14.2 – Результат работы программы

Источник: собственная разработка

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:

internal class Program

{

static void FirstMethod(int a)

{

double sum = 0;

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

{

sum = sum + Math.Pow(a, i);

Console.WriteLine($"Первый метод: {sum}");

}

}

static void SecondMethod(double a)

{

double multiply = 1;

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

{

multiply = multiply \* Math.Pow(a, j);

Console.WriteLine($"Второй метод: {multiply}");

}

}

static void Main(string[] args)

{

int a;

Console.Write("Введите А: ");

a = Convert.ToInt32(Console.ReadLine());

int n;

Console.Write("Введите N: ");

n = Convert.ToInt32(Console.ReadLine());

Thread thread1 = new Thread(() => FirstMethod(a));

Thread thread2 = new Thread(() => SecondMethod(n));

Thread thread3 = new Thread(() => FirstMethod(a));

thread1.Start();

thread2.Start();

thread3.Start();

Console.ReadLine();

}

}

}

Таблица 14.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Введите А: 14  Введите N: 15 | Первый метод: 14  Второй метод: 15  Первый метод: 14  Первый метод: 210  Второй метод: 3375  Второй метод: 11390625  Второй метод: 576650390625  Первый метод: 210  Первый метод: 2954  Второй метод: 4,378938903808594E+17  Второй метод: 4,987885095119477E+24 |

Окончание таблицы 14.3

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Второй метод: 8,396661731213823E+52  Первый метод: 2954  Первый метод: 41370  Первый метод: 579194  Первый метод: 41370  Первый метод: 8108730  Первый метод: 113522234  Первый метод: 579194  Первый метод: 1589311290  Первый метод: 8108730  Первый метод: 22250358074  Первый метод: 113522234  Первый метод: 1589311290  Первый метод: 22250358074 |

Анализ результатов:

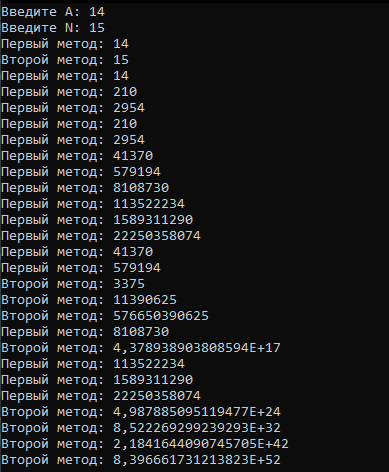


Рисунок 14.3 – Результат работы программы

Источник: собственная разработка