**14 Многопоточные приложения**

Задание 1. 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:

namespace Space

{

class Program

{

static void Main(string[] args)

{

// создаем два потока, каждый из которых будет выполнять метод "CalculateSum"

Thread thread1 = new Thread(new ThreadStart(CalculateSum));

thread1.Name = "Поток №1";

Thread thread2 = new Thread(new ThreadStart(CalculateSum));

thread2.Name = "Поток №2";

// запоминаем время начала работы программы

DateTime startTime = DateTime.Now;

// запускаем оба потока

thread1.Start();

thread2.Start();

// ожидаем завершения выполнения обоих потоков

thread1.Join();

thread2.Join();

// запоминаем время окончания работы программы

DateTime endTime = DateTime.Now;

// вычисляем время, затраченное на выполнение программы

TimeSpan duration = endTime - startTime;

Console.WriteLine("Общее время работы: " + duration.TotalMilliseconds + " миллисекунд");

}

// объявляем метод "CalculateSum", который будет выполняться в каждом из потоков

private static void CalculateSum()

{

int sum = 0;

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

{

sum += i;

}

// выводим на экран информацию о выполненной работе текущего потока

Console.WriteLine("Поток " + Thread.CurrentThread.Name + " с суммой: " + sum + ", время потрачено: " + Thread.CurrentThread.ManagedThreadId + " миллисекунд");

}

}

}

Анализ результатов:

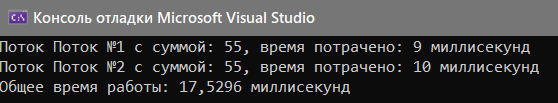


Рисунок 14.1 – Результат работы программы

Задание 2. 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:

namespace Space

{

class Program

{

static void Main(string[] args)

{

Console.Write("Введите A: ");

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

Console.Write("Введите N: ");

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

Thread thread1 = new Thread(() => Method1(A, N));

thread1.Name = "Поток №1";

Thread thread2 = new Thread(() => Method1(A, N));

thread2.Name = "Поток №2";

Thread thread3 = new Thread(() => Method2(A, N));

thread3.Name = "Поток №3";

DateTime startTime = DateTime.Now;

thread1.Start();

thread2.Start();

thread3.Start();

thread1.Join();

thread2.Join();

thread3.Join();

DateTime endTime = DateTime.Now;

TimeSpan duration = endTime - startTime;

Console.WriteLine("Общее время работы: " + duration.TotalMilliseconds + " миллисекунд");

}

private static void Method1(int A, int N)

{

int result = 0;

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

{

result += A + i;

}

Console.WriteLine("Метод 1 - " + Thread.CurrentThread.Name + " результат: " + result);

}

private static void Method2(int A, int N)

{

int result = 1;

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

{

result \*= A;

}

Console.WriteLine("Метод 2 - " + Thread.CurrentThread.Name + " результат: " + result);

}

}

}

Таблица 14.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 8  2 | 19  64  11,1382 |

Анализ результатов:

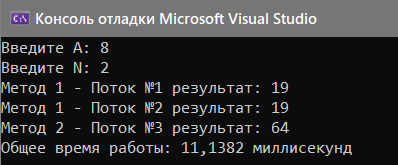


Рисунок 14.2 – Результат работы программы