**20.** **Параллельное программирование**

Задание №1

Создайте и реализуйте метод решения задачи, и выполните его в объектах класса Task используя три варианта создания объектов класса Task: Дано трехзначное число. В нем зачеркнули последнюю справа цифру и приписали ее в начале. Найти полученное число.

Листинг программы:

Console.Write("Write number: ");

int num = Convert.ToInt32(Console.ReadLine());

//first

Task<int> ts1 = new Task<int>(() => { return ReverseNum(num); });

ts1.Start();

ts1.Wait();

Console.WriteLine(ts1.Result);

//Second

Task.Run(() => Console.WriteLine(ReverseNum(num)));

//Third

Task task = Task.Factory.StartNew(() => Console.WriteLine(ReverseNum(num)));

int ReverseNum(int num)=>num % 10 \* 100 + num / 10;

Таблица 20.1 – Входные и выходные данные программы задание №1

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | 321 |

Анализ результатов:

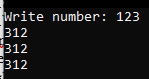


Рисунок 20.1 – Результат выполнения программы задание №1

Задание №2

Создайте массив из 2 задач (объектов класс Task) в каждом объекте выполните вычисление значения функций и выполните условия: Дождитесь выполнения всех задач; Дождитесь выполнения хот бы одной задачи. Замедлить выполнение задачи можно с помощью Thread.Sleep(n) в методе, выполняемом задачей; где n – время в миллисекундах.



Рисунок 20.2 – Функции для задания №2

Листинг программы:

using static System.Math;

Console.Write("Write number a: ");

double a = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Write time sleep for Task: ");

int n = Convert.ToInt32(Console.Read());

Task[] tasks = GetArrTasks();

Console.WriteLine("Waiting for one completed Task from an array:");

foreach (Task t in tasks)

{

t.Start();

}

Task.WaitAny(tasks);

Task[] tasks2 = GetArrTasks();

Console.WriteLine("Waiting for all Tasks from the array: ");

foreach (Task t in tasks2)

{

t.Start();

}

Task.WaitAll(tasks2);

double Func1(double a) =>

(Cos(a) + Sin(a)) / (Cos(a) - Sin(a));

double Func2(double a) =>

Tan(2 \* a) + Sin(2 \* a);

Task[] GetArrTasks() => new Task[]

{

new Task(() => { Thread.Sleep(n); Console.WriteLine($"Func1: {Func1(a):.###}"); }),

new Task(() => { Console.WriteLine($"Func2: {Func2(a):.###}"); })

};

Таблица 20.2 – Входные и выходные данные программы задание №2

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| a: 4  n: 1000 | Write number a: 4  Write time sleep for Task:  1000  Waiting for one completed Task from an array:  Func2: -5,81  Waiting for all Tasks from the array:  Func2: -5,81  Func1: -13,673  Func1: -13,673 |

Анализ результатов:

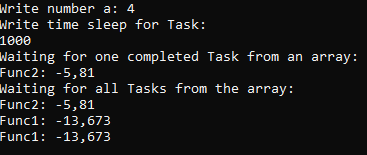


Рисунок 20.1 – Результат выполнения программы задание №3

Задание №3

Создайте два объекта класса Task. Первый объект возвращает результат вычисления, второй объект является задачей продолжения первого объекта и выводит результат первой задачи на консоль. Задания для реализации метода выполняемого в первом объекте класса Task: Дано двузначное число. Получить число, образованное при перестановке цифр заданного числа.

Листинг программы:

Console.Write("Write number: ");

int num = Convert.ToInt32(Console.ReadLine());

Task<int> taskCalculation = new Task<int>(() => { return ReverseNum(num); });

taskCalculation.Start();

taskCalculation.Wait();

Task taskWriteResult = new Task(() => { Console.WriteLine($"Result: {taskCalculation.Result}"); });

taskWriteResult.Start();

int ReverseNum(int num)

{

if((10 < num && num <= 99) || (-99 <= num && num < -10))

{

return num % 10 \* 10 + num / 10;

}

else

{

throw new Exception("Number is not range(11, 99) or range(-99,-11).");

}

}

Таблица 20.3 – Входные и выходные данные программы задание №3

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| num : 45 | 54 |

Анализ результатов:



Рисунок 20.3 – Результат выполнения программы задание №4

Задание №4

Выполните параллельное вычисления значений функции для каждого значения на отрезке [А,B]. Используя метод Parallel.For. Функция Ctg(x), A: -6, B: 4;

Листинг программы:

Parallel.For(-6, 4, Ctg);

void Ctg(int num) => Console.WriteLine(Math.Sin(num) / Math.Cos(num));

Таблица 20.4 – Входные и выходные данные программы задание №4

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| a: -6, b: 4; | -1,1578212823495775  0,29100619138474915  3,380515006246586  0,1425465430742778  2,185039863261519  -1,557407724654902  0  -0,1425465430742778  1,557407724654902  -2,185039863261519 |

Анализ результатов:

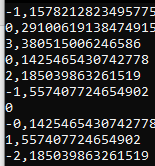


Рисунок 20.4 – Результат выполнения программы задание №5

Задание №5

Выполните прерывание выполнения метода Parallel.Foreach при некотором условии. Функции суммы и произведение чисел от 0 до N где N значения из массива (списка).

Список: 11,12,15,68

Листинг программы:

List<int> nums = new List<int>() { 11, 12, 15, 68 };

Console.WriteLine($"Sum: {Sum(nums)}");

int Sum(List<int> nums)

{

int sum = 0;

Parallel.ForEach(nums, number =>

{

sum += number;

});

return sum;

}

Таблица 20.5 – Входные и выходные данные программы задание №5

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 11,12,15,68 | 106 |

Анализ результатов:



Рисунок 20.5 – Результат выполнения программы задание №6