1. **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 четырехзначное число. Найти число, образуемое при перестановке второй и третьей цифр заданного числа.

Листинг программы:

namespace Task\_1

{

/// <summary>

/// Реализация решения задачи выполнения в объектах класса Task используя

/// три варианта создания объектов класса Task:

/// Дано четырехзначное число.

/// Найти число, образуемое при перестановке второй и третьей цифр заданного числа.

/// </summary>

class Program

{

/// <summary>

/// Главный метод выполнения.

/// </summary>

/// <param name="args">Аргументы командной строки.</param>

public static void Main(string[] args)

{

Console.Write("Введите четырехзначное число: ");

string num = Console.ReadLine();

Console.WriteLine("TaskOne");

Task task1 = new Task(() => Console.WriteLine(PermutationSecondTherdNumbrs(num)));

task1.Start();

Console.WriteLine("TaskTwo");

Task task2 = Task.Factory.StartNew(() => Console.WriteLine(PermutationSecondTherdNumbrs(num)));

Console.WriteLine("TaskThree");

Task task3 = Task.Run(() => Console.WriteLine(PermutationSecondTherdNumbrs(num)));

}

/// <summary>

/// Метод перестановки второй и третьей цифры в четырехзначном числе.

/// </summary>

/// <param name="number">Параметр четырехзначное число</param>

/// <returns></returns>

public static string PermutationSecondTherdNumbrs(string number)

{

return $"{number[0]}{number[2]}{number[1]}{number[3]}";

}

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1234 | 1324 |

Анализ результатов:

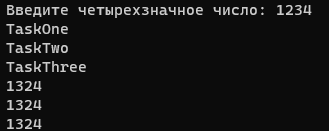


Рисунок 1.2 – Результат работы программы

Источник: собственная разработка

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 2.1 – Задание

Источник: собственная разработка

Листинг программы:

namespace Task\_2

{

/// <summary>

/// Создание массива из 2 задач (объектов класс Task)

/// в каждом объекте выполняется вычисление значения функций

/// </summary>

class Program

{

/// <summary>

/// Главный метод выполнения.

/// </summary>

/// <param name="args">Аргументы командной строки.</param>

public static void Main(string[] args)

{

Console.Write("Введите число: ");

double num = double.Parse(Console.ReadLine());

Task[] tasks1 = new Task[2]

{

new Task(() => Console.WriteLine(FOne(num))),

new Task(() => Console.WriteLine(FTwo(num)))

};

Console.WriteLine("Выполнение двух хадач");

foreach (var t in tasks1)

{

t.Start();

Thread.Sleep(50);

}

}

/// <summary>

/// Метод вычисления уравнения sin2x+sin5x-sin3x/cosx+1-2\*sin^(2)2x.

/// </summary>

/// <param name="x">Параметр числа</param>

/// <returns>Возвращает результат типа double.</returns>

public static double FOne(double x)

{

return (Math.Sin(2 \* x) + Math.Sin(5 \* x) - Math.Sin(3 \* x)) /

(Math.Cos(x) + 1 - 2 \* Math.Sin(Math.Pow(2 \* x, 2)));

}

/// <summary>

/// Метод вычисления уравнения 2sinx.

/// </summary>

/// <param name="x">Параметр числа</param>

/// <returns>Возвращает результат типа double.</returns>

public static double FTwo(double x)

{

return 2 \* Math.Sin(x);

}

}

}

Таблица 2.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 3 | -0,0206… 0,2822… |

Анализ результатов:

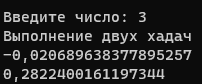


Рисунок 2.2 – Результат работы программы

Источник: собственная разработка

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 трёхзначное число. Найти сумму его первой и второй цифр.

Листинг программы:

namespace Task\_3

{

/// <summary>

/// Создание двух объектов класса Task.

/// Первый объект возвращает результат вычисления,

/// второй объект является задачей продолжения первого объекта и выводит результат первой задачи на консоль.

/// Дано трёхзначное число. Найти сумму его первой и второй цифр.

/// </summary>

class Program

{

/// <summary>

/// Главный метод выполнения.

/// </summary>

/// <param name="args">Аргументы командной строки.</param>

public static void Main(string[] args)

{

Console.Write("Введите трехзначное число: ");

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

Task<int> task1 = new Task<int>(() => SumFerstSecondNumbers(num));

task1.Start();

Task task2 = Task.Run(() => Console.WriteLine(task1.Result));

}

/// <summary>

/// Метод вычисления суммы первой и второй цифры трехзначного числа.

/// </summary>

/// <param name="num">Параметр трехзначное число.</param>

/// <returns>Возвращает сумму типа int.</returns>

public static int SumFerstSecondNumbers(int num)

{

return (num/100) + (num/10%10);

}

}

}

Таблица 3.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 234 | 5 |

Анализ результатов:



Рисунок 3.1 – Результат работы программы

Источник: собственная разработка

Задание 4. Выполните параллельное вычисления значений функции для каждого значения на отрезке [А,B]. Используя метод Parallel.For. Функция Cos(1/x), параметры А = -1, В = 9.

Листинг программы:

namespace Task\_4

{

/// <summary>

/// Выполнение параллельного вычисления значений функции для каждого значения на отрезке.

/// </summary>

class Program

{

/// <summary>

/// Главный метод выполнения.

/// </summary>

/// <param name="args">Аргументы командной строки.</param>

public static void Main(string[] args)

{

Console.Write("Введите начальное число: ");

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

Console.Write("Введите конечное число: ");

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

Parallel.For(start, finish, F);

}

/// <summary>

/// Метод высчитывания функции cos(x).

/// Вывод результата в консоль.

/// </summary>

/// <param name="x">Параметр передаваемого числа.</param>

public static void F(int x)

{

Console.WriteLine(Math.Cos(x));

}

}

}

Таблица 4.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| -1 9 | 1 -0,9899… 0,7539… -0,4161… 0,5403… -0,1455… 0,5403… 0,2836… -0,6536… 0,9601… |

Анализ результатов:

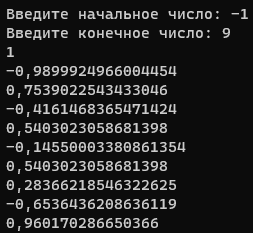


Рисунок 4.1 – Результат работы программы

Источник: собственная разработка

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). Массив значений {34,7,87,12}.

Листинг программы:

namespace Task\_5

{

/// <summary>

/// Выполнения метода Parallel.Foreach.

/// </summary>

class Program

{

/// <summary>

/// Главный метод выполнения.

/// </summary>

/// <param name="args">Аргументы командной строки.</param>

public static void Main(string[] args)

{

int[] array = InputNumInArray();

ParallelLoopResult resultSum = Parallel.ForEach<int>(array, SumNumInArray);

ParallelLoopResult resultMul = Parallel.ForEach<int>(array, MulNumInArray);

}

/// <summary>

///

/// </summary>

/// <returns></returns>

public static int[] InputNumInArray()

{

int[] array = new int[4];

for (int i = 0; i < array.Length; i++)

{

Console.Write("Введите число в массив: ");

array[i] = int.Parse(Console.ReadLine());

}

return array;

}

/// <summary>

/// Метод высчитывания суммы чисел от 0 до введенного значения.

/// </summary>

/// <param name="x">Параметр передаваемого числа.</param>

public static void SumNumInArray(int x)

{

int result = 0;

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

{

result += i;

}

Console.WriteLine($"Сумма значений до {x} = {result}");

}

/// <summary>

/// Метод высчитывания произведения чисел от 0 до введенного значения.

/// </summary>

/// <param name="x">Параметр передаваемого числа.</param>

public static void MulNumInArray(int x)

{

int result = 1;

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

{

result \*= i;

}

Console.WriteLine($"Произведение значений до {x} = {result}");

}

}

}

Таблица 5.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 34 7 87 12 | 21 66 3741 561 0 0 0 0 |

Анализ результатов:

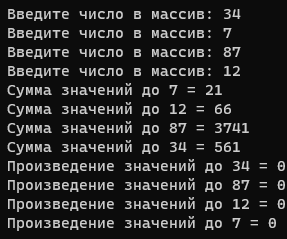


Рисунок 5.1 – Результат работы программы

Источник: собственная разработка