#### Министерство науки и высшего образования Российской Федерации

#### Федеральное государственное бюджетное образовательное учреждение

#### высшего образования

#### «Владимирский государственный университет

#### имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа №3

по дисциплине

" Программирование компьютерной графики"

Выполнил:

ст. гр. ПРИ-117

Емельянов Д.В.

Принял:

Жигалов И.Е.

Владимир, 2020 г.

**Цель работы**

Освоение средств организации многопоточных вычислений в C#.

**Выполнение работы**

**Вариант №5**

Изначальная формула:

Изменённая формула:

При изначальной формуле не было возможности посчитать результат! Также при 0 значении i.

Значения параметров:

a = 0, b = 6, n = 35

Код программы:

В ходе выполнения использовался класс System.Threading.Tasks.Task. Если очень очень очень грубо, то это современный аналог Thread.

Также в данном программе присутствует избыточность многопоточности, это сделано в целях показа умения работы с ней!

using System;

using System.Collections.Generic;

using System.Threading.Tasks;

namespace EmelyanovDmitry\_PRI117\_lab3

{

class Program

{

private static async Task<double> CalcFormula(double xi)

{

double CalcFirst(double x)

{

return 1 / (Math.Pow((x - 2), 2) - x - 1);

}

double CalcSecond(double x)

{

return Math.Log(5 \* x);

}

double CalcThird(double x)

{

return Math.Exp(7 \* Math.Sqrt(x));

}

double CalcFourth(double x)

{

return 0.3 \* (Math.Pow(x, 3) + Math.Pow(x, 2) - 1);

}

var result1Task = Task.Run(() => CalcFirst(xi));

var result2Task = Task.Run(() => CalcSecond(xi));

var result3Task = Task.Run(() => CalcThird(xi));

var result4Task = Task.Run(() => CalcFourth(xi));

return

await result1Task.ConfigureAwait(false) +

await result2Task.ConfigureAwait(false) +

await result3Task.ConfigureAwait(false) +

await result4Task.ConfigureAwait(false);

}

private static double CalcXi(int a, int i, double h)

{

return a + i \* h;

}

static async Task Main(string[] args)

{

var a = 0;

var b = 6;

var n = 35;

var h = (b - a) / (double)n;

var results = await Task.WhenAll(

Enumerable.Range(0, n + 1 )

.Select(i =>

Task.Run(() => CalcFormula(CalcXi(a, i, h)))))

.ConfigureAwait(false);

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

{

Console.WriteLine("Номер итерации: " + i

+ ", Результат вычисления: " + results[i]);

}

Console.ReadKey();

for (int i = 0; i < tasks.Count; i++)

{

var result = await tasks[i].ConfigureAwait(false);

Console.WriteLine("Номер итерации: " + i

+ ", Результат вычисления: " + result);

}

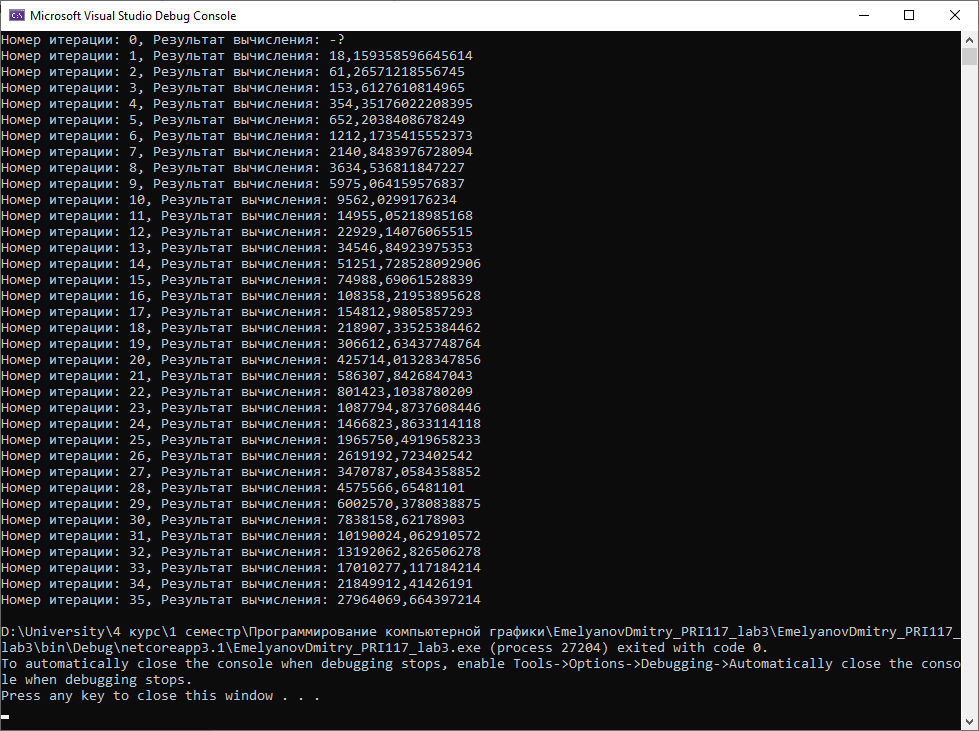
Console.ReadKey();

}

}

}

Выполнение программы:



**Выводы**

Освоил средства организации многопоточных вычислений в C#.