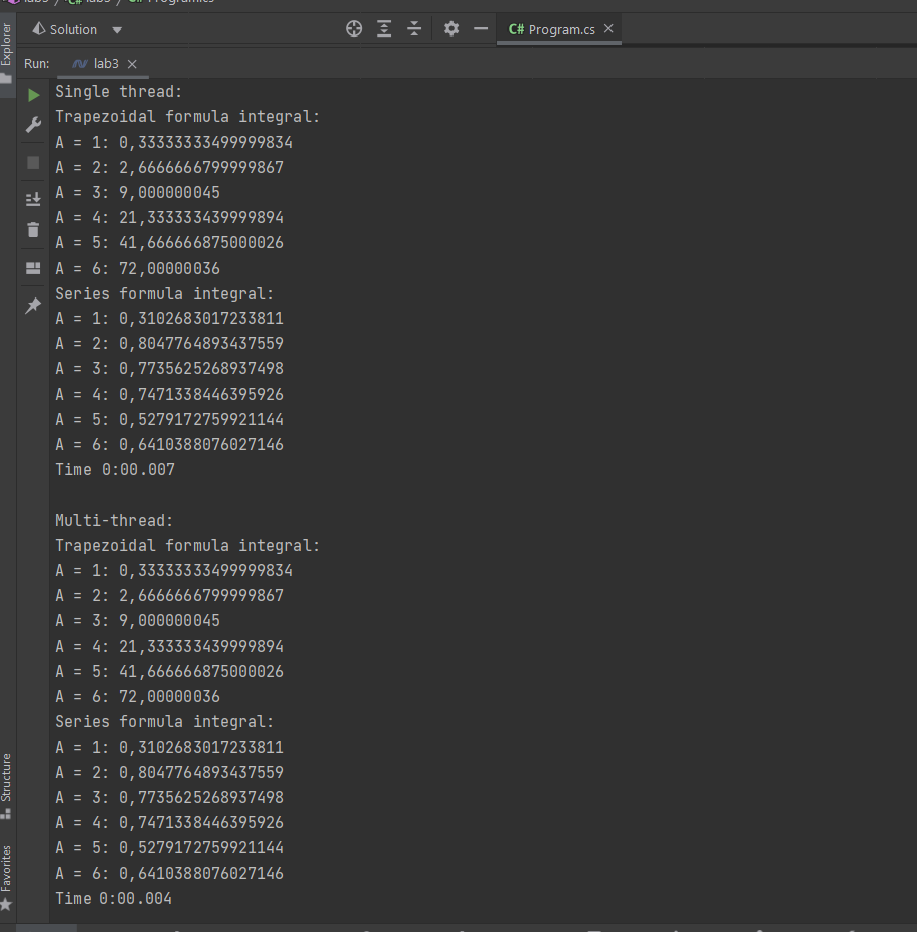
# Лабораторная работа 3

**Цель:** программа вычисляет интеграл ∫ sin x² dx используя формулу трапеций и формулу разложения в ряд.



**Код программы:**

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Threading.Tasks;

double Sin2X(double x) => x \* x;

Main();

double TrapezoidalIntegral(Func<double, double> func, double a, double b, int n)

{

var result = (func(a) + func(b)) / 2;

var h = (b - a) / n;

for (var i = 1; i < n; i += 1)

{

result += func(a + i \* h);

}

return result \* h;

}

double GetElementIntegralIncrement(double a, double i)

{

if (i == 0)

{

return Math.Pow(a, 3) / 3;

}

return -Math.Pow(a, 4) \* (4 \* i - 1) / (2 \* i \* (2 \* i + 1) \* (4 \* i + 3));

}

double SeriesIntegral(double x)

{

var i = 0.0;

var currentElement = GetElementIntegralIncrement(x, i);

var previousElement = 0.0;

var sum = currentElement;

while (Math.Abs(currentElement - previousElement) > double.Epsilon)

{

i += 1;

previousElement = currentElement;

currentElement \*= GetElementIntegralIncrement(x, i);

sum += currentElement;

}

return sum;

}

void RunTrapezoidalIntegral(List<double> upperBoundOfIntegrationValues)

{

const double lowerBoundOfIntegration = 0.0;

const int n = 10000;

Console.WriteLine("Trapezoidal formula integral:");

foreach (var a in upperBoundOfIntegrationValues)

{

Console.WriteLine($"A = {a}: {TrapezoidalIntegral(Sin2X, lowerBoundOfIntegration, a, n)}");

}

}

void RunSeriesIntegral(List<double> upperBoundOfIntegrationValues)

{

Console.WriteLine("Series formula integral:");

foreach (var a in upperBoundOfIntegrationValues)

{

Console.WriteLine($"A = {a}: {SeriesIntegral(a)}");

}

}

void SingleThreadIntegrals(List<double> upperBoundOfIntegrationValues)

{

Console.WriteLine("\nSingle thread:");

var stopwatch = new Stopwatch();

stopwatch.Start();

RunTrapezoidalIntegral(upperBoundOfIntegrationValues);

RunSeriesIntegral(upperBoundOfIntegrationValues);

stopwatch.Stop();

var timeTaken = stopwatch.Elapsed;

Console.WriteLine("Time " + timeTaken.ToString(@"m\:ss\.fff"));

}

void MultiThreadIntegrals(List<double> upperBoundOfIntegrationValues)

{

const int THREADS\_COUNT = 2;

Console.WriteLine("\nMulti-thread:");

var stopwatch = new Stopwatch();

stopwatch.Start();

Parallel.For(

(long) 0,

THREADS\_COUNT,

i =>

{

if (i % 2 == 0)

{

RunTrapezoidalIntegral(upperBoundOfIntegrationValues);

}

else

{

RunSeriesIntegral(upperBoundOfIntegrationValues);

}

});

stopwatch.Stop();

var timeTaken = stopwatch.Elapsed;

Console.WriteLine("Time " + timeTaken.ToString(@"m\:ss\.fff"));

}

void Main()

{

List<double> upperBoundOfIntegrationValues = new() {1, 2, 3, 4, 5, 6};

SingleThreadIntegrals(upperBoundOfIntegrationValues);

MultiThreadIntegrals(upperBoundOfIntegrationValues);

}