Лабораторная работа №1:

1-й уровень:

№4:

using System;

using static System.Console;

using static System.Math;

namespace LaboratoryWorkNo1

{

class Program

{

static void Main(string[] args)

{

Write("Введите значение X: ");

double x = double.Parse(ReadLine());

WriteLine("s = " + FunctionValue(x).ToString());

}

static double FunctionValue(double x)

{

double argument = x;

double numerator = Cos(argument);

double denominator = 1;

double value = numerator / denominator;

for (int i = 2; i < 9; i++)

{

argument += x;

denominator \*= x;

value += numerator / denominator;

}

return value;

}

}

}

№9:

using System;

using static System.Console;

namespace Task09

{

class Program

{

static void Main(string[] args)

{

WriteLine("s = " + FunctionValue().ToString());

}

static double FunctionValue()

{

int firstGrade = -1;

int secondGrade = 5;

int factorial = 1;

double numerator = (double) firstGrade \* secondGrade;

double result = numerator / factorial;

for (int i = 2; i < 7; i++)

{

firstGrade \*= -1;

secondGrade \*= 5;

factorial \*= i;

numerator = (double) firstGrade \* secondGrade;

result += numerator / factorial;

}

return result;

}

}

}

№15:

using System;

using static System.Console;

namespace Task15

{

class Program

{

static void Main(string[] args)

{

WriteLine("5-й член последовательности: " + FindInOrder(5));

}

static string FindInOrder(int n)

{

if (n == 1)

return "1/1";

int currentNum = 2, currentDen = 1;

int previousNum = 1, previousDen = 1;

for (int i = 3; i <= n; i++)

{

int newNumerator = currentNum + previousNum;

int newDenometator = currentDen + previousDen;

previousNum = currentNum;

previousDen = currentDen;

currentNum = newNumerator;

currentDen = newDenometator;

}

return currentNum.ToString() + '/' + currentDen.ToString();

}

}

}

№16:

using System;

using static System.Console;

namespace Task16

{

class Program

{

static void Main(string[] args)

{

WriteLine("Чтобы покрыть доску понадобится:");

double totalGramms = GrammsInChessField();

WriteLine(totalGramms.ToString("N20") + " граммов зерна");

WriteLine(InKilograms(totalGramms).ToString("N20") + " килограммов зерна");

WriteLine(InTonns(totalGramms).ToString("N20") + " тонн зерна");

WriteLine("\nТочность для типа double = " + double.Epsilon);

}

static double GrammsInChessField()

{

double gramsInRoot = 1d / 15d;

for (int i = 1; i < 64; i++)

gramsInRoot \*= 2;

return gramsInRoot;

}

static double InKilograms(double gramms)

{

return gramms / 1000d;

}

static double InTonns(double gramms)

{

return gramms / 1000000d;

}

}

}

№18:

using System;

using static System.Console;

namespace Task18

{

class Program

{

static void Main(string[] args)

{

ShowAmoebaCount(10);

}

static void ShowAmoebaCount(int startCount)

{

int elapsedHours = 0;

WriteLine("0ч: " + startCount);

long totalCount = startCount;

while (elapsedHours < 24)

{

elapsedHours += 3;

totalCount \*= 2;

WriteLine(elapsedHours + "ч: " + totalCount);

}

}

}

}

2-й уровень:

№2:

using System;

using static System.Console;

namespace L2\_task02

{

class Program

{

static void Main(string[] args)

{

WriteLine("Наибольшее значение сомножителя для L = 30000: " + GetMaxSubmultipler(30000).ToString());

}

static int GetMaxSubmultipler(int L)

{

int result = 1, multiper = 1, previousResult = 1;

while (result < L)

{

previousResult = result;

multiper += 3;

result \*= multiper;

}

return previousResult;

}

}

}

№4:

using System;

using static System.Console;

using static System.Math;

namespace L2\_task04

{

internal class Program

{

static void Main(string[] args)

{

Write("Введите значение X: ");

double x = double.Parse(ReadLine());

double epsilon = 0.0001;

WriteLine("s = " + FunctionValue(x, epsilon) + ", при eplison = " + epsilon.ToString());

}

static double FunctionValue(double x, double epsilon)

{

double currentValue = 1, result = currentValue;

while (Abs(currentValue) >= epsilon)

{

currentValue \*= x \* x;

result += currentValue;

}

return result;

}

}

}

№7, 8:

using System;

using static System.Console;

namespace L2\_task07

{

internal class Program

{

static void Main(string[] args)

{

ShowSportsmenStats(10, 10);

}

static void ShowSportsmenStats(double baseRunRange, double upPercentage)

{

double days = 1;

double currentDayRange = baseRunRange;

double runnedRange = baseRunRange;

bool aSolved = false, bSolved = false, cSolved = false;

while (!aSolved || !bSolved || !cSolved)

{

days++;

currentDayRange = currentDayRange \* (1 + upPercentage / 100d);

runnedRange += currentDayRange;

if (days == 7 && !aSolved)

{

WriteLine("a) Суммарный путь за 7 дней = " + runnedRange.ToString() + " км.");

aSolved = true;

}

if (runnedRange >= 100 && !bSolved)

{

WriteLine("б) 100 км. спортсмен пробежал за " + days.ToString() + " дней");

bSolved = true;

}

if (currentDayRange > 20 && !cSolved)

{

WriteLine("в) Больше 20 км. в день спортсмен будет пробегать через " + days + " дней");

cSolved = true;

}

}

}

}

}

3-й уровень:

№1:

using System;

using static System.Console;

using static System.Math;

namespace L3\_task01

{

internal class Program

{

static void Main(string[] args)

{

ShowFunctionValues(0.1, 1, 0.1);

}

static void ShowFunctionValues(double start, double end, double delta)

{

for (double x = start; x <= end; x += delta)

{

WriteLine(

" x = " + x.ToString() + ":" +

" y = " + Cos(x).ToString() +

" s = " + FunctionValue(x, 0.0001).ToString());

}

}

static double FunctionValue(double x, double epsilon)

{

double result = 0, currentValue = 1, i = 0;

while (Abs(currentValue) >= epsilon)

{

result += currentValue;

currentValue \*= -1 \* x \* x / ++i / ++i;

}

return result;

}

}

}

№2:

using System;

using static System.Console;

using static System.Math;

namespace L3\_task02

{

class Program

{

static void Main(string[] args)

{

ShowFunctionValues(0.1, 1, 0.1);

}

static void ShowFunctionValues(double start, double end, double delta)

{

for (double x = start; x <= end; x += delta)

{

WriteLine(

" x = " + x.ToString() + ":" +

" y = " + BaseFunctionValue(x).ToString() +

" s = " + FunctionValue(x, 0.0001).ToString());

}

}

static double BaseFunctionValue(double x)

{

double numerator = x \* Sin(PI / 4d);

double denominator = 1 - 2 \* x \* Cos(PI / 4d) + x \* x;

return numerator / denominator;

}

static double FunctionValue(double x, double epsilon)

{

double result = 0, multipler = x, i = 1;

while (Abs(multipler) >= epsilon)

{

result += multipler \* Sin(i \* PI / 4d);

multipler \*= x;

i++;

}

return result;

}

}

}