**1 задание**

using System;

namespace циклы

{

class Program

{

static double[] a = new double[1000];

static void Main(string[] args)

{

int n = InputA();

Console.WriteLine("сумма\_F = {0}", sumF(n));

Console.WriteLine("сумма\_W = {0}", sumW(n));

Console.WriteLine("сумма\_D = {0}", sumD(n));

Console.WriteLine("произведение\_F = {0}", multyF(n));

Console.WriteLine("произведение\_W = {0}", multyW(n));

Console.WriteLine("произведение\_D = {0}", multyD(n));

Console.ReadKey();

}

static int InputA()

{

int n;

Console.Write("Ввести кол-во элементов:");

n = Convert.ToInt32(Console.ReadLine());

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

{

Console.Write("a[{0}]=", i);

a[i] = Convert.ToDouble(Console.ReadLine());

}

return n;

}

static double sumF(int n)

{

double s = 0;

for (int k = 0; k < n; k++)

s = s + a[k];

return s;

}

static double sumW(int n)

{

double s = 0;

int k = 0;

while (k < n)

{

s = s + a[k];

k++;

}

return s;

}

static double sumD(int n)

{

double s = 0;

int k = 0;

do

{

s = s + a[k];

k++;

} while (k <= n);

return s;

}

static double multyF(int n)

{

double p = 1;

for (int k = 0; k < n; k++)

p = p \* a[k];

return p;

}

static double multyW(int n)

{

double p = 1;

int k = 0;

while (k < n)

{

p = p \* a[k];

k++;

}

return p;

}

static double multyD(int n)

{

double p = 1;

int k = 0;

do

{

p = p \* a[k];

k++;

} while (k < n);

return p;

}

}

}

**2 задание**

Сходства между алгоритмами вычисления суммы, произведения и циклами:

Оба поддерживают текущую сумму (или произведение), которая обновляется на каждой итерации.

Различия между алгоритмами вычисления суммы и произведения:

Суммирование использует сложение (+), в то время как умножение использует умножение (\*).

Сходства между структурами циклов:

Оба for и while цикла могут использоваться для вычисления суммы и произведения.

Различия между структурами циклов:

For циклы предпочтительны, когда известно количество итераций, в то время как while циклы более гибки для динамических условий.