2020 г

Оглавление

[1 Метод max(x, y) 3](#_Toc59886257)

[1.1 Формулировка задания 3](#_Toc59886258)

[1.2 Выполнение программы 3](#_Toc59886259)

[1.3 Исходный код 3](#_Toc59886260)

[2 Вычисление значений арифметического выражения 4](#_Toc59886261)

[2.1 Формулировка задания. 4](#_Toc59886262)

[2.2 Выполнение программы. 4](#_Toc59886263)

[2.3 Исходный код. 4](#_Toc59886264)

[3 Вычисление значений тригонометрического выражения 5](#_Toc59886265)

[3.1 Формулировка задания. 5](#_Toc59886266)

[3.2 Выполнение программы. 5](#_Toc59886267)

[3.3 Исходный код. 5](#_Toc59886268)

[4 Работа с массивами 6](#_Toc59886269)

[4.1 Формулировка задания. 6](#_Toc59886270)

[4.2 Выполнение программы. 6](#_Toc59886271)

[4.3 Исходный код. 6](#_Toc59886272)

[ПРИЛОЖЕНИЕ А. Вспомогательные вычислительные модули 8](#_Toc59886273)

# Метод max(x, y)

## Формулировка задания

Написать метод max(x, y), находящий максимальное значение из двух чисел. С его помощью найти максимальное значение из четырёх чисел a, b, c, d.

Определите метод расширения, подсчитывающий количество делителей в исходном целом числе.

## Выполнение программы

7.1.png

## Исходный код

Random rnd = new Random();

Arithmetics arith = new Arithmetics();

// 2. Написать метод max(x, y), находящий максимальное значение из двух чисел.

// С его помощью найти максимальное значение из четырёх чисел a, b, c, d.

int a = rnd.Next(-5, 6);

int b = rnd.Next(-5, 6);

int c = rnd.Next(-5, 6);

int d = rnd.Next(-5, 6);

int m = arith.Max(a, arith.Max(b, arith.Max(c, d)));

Console.WriteLine("\nМаксимум из {0}, {1}, {2}, {3}: {4}", a, b, c, d, m);

// Определите метод расширения, подсчитывающий количество делителей в исходном целом числе.

Console.Write("Введите целое число для подсчёта количества делителей: ");

int num = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("\nколичество делителей в числе {0}: {1}", num, num.DividersCount());

# Вычисление значений арифметического выражения

## Формулировка задания.

Написать метод, вычисляющий значение xn/(n+x). С его помощью вычислить выражение:



Определите метод расширения, вычисляющий дробную часть в выражении.

## Выполнение программы.

7.2.png

## Исходный код.

// 5. Написать метод, вычисляющий значение xn/(n+x). С его помощью вычислить выражение.

double x = rnd.Next(1, 5);

double expression = 0.0;

for(int i = 0; i < 10; i++)

{

expression += arith.Proportion(x, i);

}

Console.WriteLine("\nРезультат вычисления выражения для x = {0}: {1}", x, expression);

// Определите метод расширения, вычисляющий дробную часть в выражении

Console.WriteLine("\nДробная часть выражения: {0}", expression.Fraction());

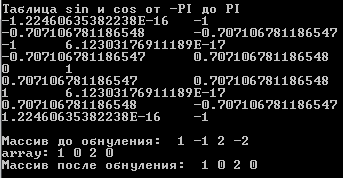
# Вычисление значений тригонометрического выражения

## Формулировка задания.

Написать метод, который вычисляет значения x=sin2(a) и y=cos2(a). Напечатать таблицу значений от –π до π с шагом π/4.

Написать метод расширения, обнуляющий в массиве все отрицательные элементы.

## Выполнение программы.



## Исходный код.

// 11. Написать метод, который вычисляет значения x=sin2(a) и y=cos2(a).

// Напечатать таблицу значений от –π до π с шагом π/4

Console.WriteLine("\nТаблица sin и cos от -PI до PI");

x = -Math.PI;

double step = Math.PI / 4.0;

while (x <= Math.PI)

{

double[] sincos = arith.SinCos(x);

Console.WriteLine("{0}\t{1}", sincos[0], sincos[1]);

x += step;

}

// Написать метод расширения, обнуляющий в массиве все отрицательные элементы.

int[] array = new int[] { 1, -1, 2, -2 };

array.Print("\nМассив до обнуления: ");

array.NullifyNegative();

array.Print("Массив после обнуления: ");

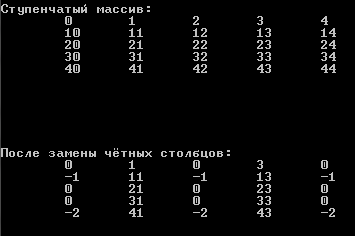
# Работа с массивами

## Формулировка задания.

Дан массив размером n×n, элементы которого целые числа (для хранения массива n×n использовать ступенчатый массив):

* Четные столбцы таблицы заменить на вектор Х.
* Для каждой строки подсчитать количество положительных элементов и записать данные в новый массив.
* . Для каждого столбца найти номер последнего нечетного элемента и записать данные в новый массив.

## Выполнение программы.



## Исходный код.

Console.WriteLine("\nЧасть 2");

int n = 5;

int[][] jagged = new int[n][];

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

{

jagged[i] = new int[n];

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

{

jagged[i][j] = i \* 10 + j;

}

}

arrays.PrintJagged(jagged, n, "\nСтупенчатый массив:");

arrays.ReplaceEvenColumns(jagged, n, new int[] { 0, -1, 0, 0, -2 });

arrays.PrintJagged(jagged, n, "\nПосле замены чётных столбцов:");

int[] posCounts = arrays.CountPositive(jagged, n);

arrays.PrintVector(posCounts, n, "\nКоличество положительных элементов в строках:");

int[] oddIndices = arrays.FindLastOdd(jagged, n);

arrays.PrintVector(oddIndices, n, "\nНомера последнего нечётного элемента в столбцах:");

Console.WriteLine("\nНажмите любую клавишу...");

Console.ReadKey();

# ПРИЛОЖЕНИЕ А. Вспомогательные вычислительные модули

**Arithmetics.cs**

using System;

namespace Tasks

{

public class Arithmetics

{

// методы, находящие максимальное значение из двух чисел (три варианта вызова)

public int Max(int x, int y)

{

if (x <= y)

return y;

return x;

}

public float Max(float x, float y)

{

if (x <= y)

return y;

return x;

}

public double Max(double x, double y)

{

if (x <= y)

return y;

return x;

}

// метод, вычисляющий значение (x в степени n)/(n+x). С его помощью вычислить выражение

public double Proportion(double x, double n = 1.0)

{

return Math.Pow(x, n) / (x + n);

}

// метод, который вычисляет значения x=sin2(a) и y=cos2(a).

public double[] SinCos(double a = Math.PI / 2.0)

{

double[] result = new double[2];

result[0] = Math.Sin(a);

result[1] = Math.Cos(a);

return result;

//return new double[2]{ Math.Sin(a), Math.Cos(a) };

}

}

}

**NumericExtentions.cs**

using System;

namespace lab7

{

public static class NumericExtensions

{

public static int DividersCount(this int num)

{

int result = 1;

for (int i = 2; i <= Math.Abs(num); i++)

if (num % i == 0)

result++;

return result;

}

public static double Fraction(this double d)

{

double d3 = Math.Truncate(d);

double result = d - d3;

return result;

}

public static void NullifyNegative(this int[] array)

{

Console.Write("array:");

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

{

if (array[i] < 0)

{

array[i] = 0;

}

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

}

Console.WriteLine("");

}

public static void Print(this int[] array, string prefix = "")

{

Console.Write(prefix);

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

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

Console.WriteLine("");

}

}

}

**Arrays.cs**

using System;

namespace Tasks

{

public class Arrays

{

// Печать двумерного ступенчатого массива

public void PrintJagged(int[][] jagged, int n, string caption = "")

{

if (caption.Length > 0)

Console.WriteLine(caption);

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

{

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

{

Console.Write("\t{0}", jagged[i][j]);

}

Console.WriteLine("");

}

}

// Печать одномерного массива

public void PrintVector(int[] vector, int n, string caption = "")

{

if (caption.Length > 0)

Console.Write(caption);

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

{

Console.Write("\t{0}", vector[i]);

}

Console.WriteLine("");

}

// 2. Четные столбцы таблицы заменить на вектор Х.

public void ReplaceEvenColumns(int[][] jagged, int n, int[] x)

{

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

{

if (j % 2 == 0)

{

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

{

jagged[i][j] = x[i];

}

}

Console.WriteLine("");

}

}

// Для каждой строки подсчитать количество положительных элементов

// и записать данные в новый массив.

public int[] CountPositive(int[][] jagged, int n)

{

int[] result = new int[n];

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

{

int count = 0;

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

{

if (jagged[i][j] > 0)

{

count++;

}

}

result[i] = count;

Console.WriteLine("");

}

return result;

}

// Для каждого столбца найти номер последнего нечетного элемента

// и записать данные в новый массив

public int[] FindLastOdd(int[][] jagged, int n)

{

int[] result = new int[n];

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

{

// значение по умолчанию предполагает, что в строке нечётных элементов нет

result[i] = -1;

}

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

{

for (int i = n - 1; i >= 0; i--)

{

if (jagged[i][j] % 2 == 1)

{

result[j] = i;

break;

}

}

Console.WriteLine("");

}

return result;

}

}

}