**Exercises**

1. Write a code that by given name prints on the console "**Hello, <name>!"** (for example: "**Hello, Peter!**").

using System;

namespace detyra1

{

class Program

{

static void ReturnName(string name)

{

Console.WriteLine("Hello {0}", name);

}

static void Main(string[] args)

{

Console.Write("Enter name: ");

ReturnName(Console.ReadLine());

}

}

}

1. Create a method **GetMax()** with two integer (**int**) parameters, that returns **maximal** of the two numbers. Write a program that reads three numbers from the console and prints the biggest of them. Use the **GetMax()**method you just created. Write a test program that validates that the methods works correctly.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace detyra2

{

class Program

{

static int a;

public static void GetMax(int first, int second)

{

if (first > second) a = first;

else a = second;

}

public static void Main(string[] args)

{

Console.Write("Enter first number: ");

a = Int32.Parse(Console.ReadLine());

Console.Write("Enter second number: ");

int b = Int32.Parse(Console.ReadLine());

Console.Write("Enter third number: ");

int c = Int32.Parse(Console.ReadLine());

GetMax(a, b);

GetMax(a, c);

Console.WriteLine("Biggest number is {0}", a);

}

}

}

3.    Write a method that returns the **English name of the last digit** of a given number. Example: for **512** prints "**two**"; for **1024** à "**four**".

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace detyra3

{

class Program

{

static void GetName(string number)

{

switch (number[number.Length - 1])

{

case '1': Console.WriteLine("One"); break;

case '2': Console.WriteLine("Two"); break;

case '3': Console.WriteLine("Three"); break;

case '4': Console.WriteLine("Four"); break;

case '5': Console.WriteLine("Five"); break;

case '6': Console.WriteLine("Six"); break;

case '7': Console.WriteLine("Seven"); break;

case '8': Console.WriteLine("Eight"); break;

case '9': Console.WriteLine("Nine"); break;

case '0': Console.WriteLine("Zero"); break;

default: Console.WriteLine("Wrong input."); break;

}

}

static void Main(string[] args)

{

Console.Write("Enter a number: ");

string number = Console.ReadLine();

GetName(number);

}

}

}

1. Write a method that finds **how many times certain number can be found in a given array**. Write a program to test that the method works correctly.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace detyra4

{

class Program

{

static int CountNumber(int number, int[] arr)

{

int counter = 0;

for (int i = 0; i < arr.Length; i++) if (number == arr[i]) counter++;

return counter;

}

static void Main(string[] args)

{

Console.Write("Enter array length: ");

int length = Int32.Parse(Console.ReadLine());

int[] arr = new int[length];

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

{

Console.Write("Enter {0} element: ", i);

arr[i] = Int32.Parse(Console.ReadLine());

}

Console.Write("Enter number: ");

int number = Int32.Parse(Console.ReadLine());

Console.WriteLine("{0} was found {1} times.", number, CountNumber(number, arr));

}

}

}

1. Write a method that checks whether an element, from a certain position in an array is **greater than its two neighbors**. Test whether the method works correctly.

using System;

namespace detyra5

{

class Program

{

static void CompareNumber(int pos, int[] arr)

{

if (pos == 0)

{

if (arr[0] < arr[1]) Console.WriteLine("{0} is smaller than it's right.", arr[0]);

else if (arr[0] > arr[1]) Console.WriteLine("{0} is bigger than it's right.", arr[0]);

else Console.WriteLine("{0} is equal to it's right.", arr[0]);

}

else if (pos == arr.Length - 1)

{

if (arr[arr.Length - 1] < arr[arr.Length - 2]) Console.WriteLine("{0} is smaller than it's left.", arr[pos]);

else if (arr[arr.Length - 1] > arr[arr.Length - 2]) Console.WriteLine("{0} is bigger than it's left.", arr[pos]);

else Console.WriteLine("{0} is equal to it's left.", arr[pos]);

}

else

{

if (arr[pos] < arr[pos - 1])

{

if (arr[pos] < arr[pos + 1]) Console.WriteLine("{0} is smaller than it's neighbours.", arr[pos]);

else if (arr[pos] == arr[pos + 1]) Console.WriteLine("{0} is smaller than it's left and equal to it's right.", arr[pos]);

else Console.WriteLine("{0} is smaller than it's left and bigger than it's right.", arr[pos]);

}

else if (arr[pos] == arr[pos - 1])

{

if (arr[pos] < arr[pos + 1]) Console.WriteLine("{0} is euqal to it's left and smaller than it's right.", arr[pos]);

else if (arr[pos] == arr[pos + 1]) Console.WriteLine("{0} is equal to it's neighbours.", arr[pos]);

else Console.WriteLine("{0} is equal to it's left and bigger than it's right.", arr[pos]);

}

else

{

if (arr[pos] < arr[pos + 1]) Console.WriteLine("{0} is bigger than it's left and smaller than it's right.", arr[pos]);

else if (arr[pos] == arr[pos + 1]) Console.WriteLine("{0} is bigger than it's left and equal to it's right.", arr[pos]);

else Console.WriteLine("{0} is bigger than it's neighbours.", arr[pos]);

}

}

}

static void Main(string[] args)

{

Console.Write("Enter array length: ");

int length = Int32.Parse(Console.ReadLine());

int[] arr = new int[length];

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

{

Console.Write("Enter {0} element: ", i);

arr[i] = Int32.Parse(Console.ReadLine());

}

Console.Write("Enter position in array: ");

int pos = Int32.Parse(Console.ReadLine());

CompareNumber(pos, arr);

}

}

}

1. Write a method that returns the position of **the first occurrence** of an element from an array, such that it is greater than its two neighbors simultaneously. Otherwise the result must be **-1**.

using System;

namespace detyra6

{

class Program

{

static int number = int.MinValue;

static void CompareNumber(int[] arr)

{

for (int i = 1; i < arr.Length - 1; i++)

if (arr[i] > arr[i - 1] && arr[i] > arr[i + 1])

{

number = arr[i];

break;

}

}

static void Main(string[] args)

{

Console.Write("Enter array length: ");

int length = Int32.Parse(Console.ReadLine());

int[] arr = new int[length];

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

{

Console.Write("Enter {0} element: ", i);

arr[i] = Int32.Parse(Console.ReadLine());

}

CompareNumber(arr);

if (number == int.MinValue) Console.WriteLine("-1");

else Console.WriteLine("{0} is bigger than it's neighbours", number);

}

}

}

1. Write a method that prints the digits of a given decimal number in a reversed order. For example **256**, must be printed as **652**.

using System;

namespace detyra7

{

class Program

{

public static string Reverse(string number)

{

char[] charArray = number.ToCharArray();

Array.Reverse(charArray);

return new string(charArray);

}

static void Main(string[] args)

{

Console.Write("Enter number: ");

string number = Console.ReadLine();

Console.WriteLine(Reverse(number));

}

}

}

//another try

//static void ReverseNumber ( int num)

//{

// int reverse = 0;

// while (num ! =0)

// { int lastNum = num % 10;

// reverse = reverse \* 10 + lastNum;

// num = num / 10;

// }

// Console.WriteLine("the reversed number is {0}", reverse);

//}

//static void Main(string[] args)

//{

// Console.Write("write the number:");

// ReverseNumberNumber(Convert.ToInt32(Console.ReadLine()));

//}

1. Write a method that calculates the **sum of two very long positive integer numbers**. The numbers are represented as **array digits** and the last digit (the ones) is stored in the array at index 0. Make the method work for all numbers with length up to 10,000 digits.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace detyra8

{

class Program

{

static void Main(string[] args)

{

int length;

int[] Array1 = new int[10000];

int[] Array2 = new int[10000];

Console.Write("Enter first number: ");

String s1 = Console.ReadLine();

length = s1.Length;

for (int i = 0; i < s1.Length; i++)

{

Array1[i] = Convert.ToInt32(s1.Substring(s1.Length - 1 - i, 1));

}

Console.Write("Enter second number: ");

String s2 = Console.ReadLine();

if (s2.Length > length)

{

length = s2.Length;

}

for (int i = 0; i < s2.Length; i++)

{

Array2[i] = Convert.ToInt32(s2.Substring(s2.Length - 1 - i, 1));

}

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

{

Array1[i] += Array2[i];

if (Array1[i] >= 10)

{

Array1[i] -= 10;

Array1[i + 1]++;

}

}

if (Array1[length] == 1)

{

length++;

}

Console.WriteLine();

Console.Write("Result is: ");

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

{

Console.Write(Array1[length - 1 - i]);

}

Console.ReadLine();

}

}

}

9.    Write a method that finds **the biggest element of an array**. Use that method to implement **sorting in descending order**. using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace detyra9

{

class Program

{

public static int GetMax(int[] array, int start, int end)

{

int maxNum = array[start];

for (int i = start + 1; i < end; i++)

{

if (array[i] > maxNum) maxNum = array[i];

}

return maxNum;

}

public static void Main(string[] args)

{

Console.Write("Enter array length: ");

int length = Int32.Parse(Console.ReadLine());

int[] arr = new int[length];

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

{

Console.Write("Enter {0} element: ", i);

arr[i] = Int32.Parse(Console.ReadLine());

}

Console.Write("\nEnter start index: ");

int startIndex = Int32.Parse(Console.ReadLine());

Console.Write("Enter end index: ");

int endIndex = Int32.Parse(Console.ReadLine());

Console.WriteLine("Biggest number in interval {0}-{1} is {2}.", startIndex, endIndex, GetMax(arr, startIndex, endIndex));

}

}

}

10.   Write a program that calculates and prints the **n!** for any **n** in the range **[1**…**100]**. using System;

namespace detyra10

{

class Program

{

static double Factorial(double number)

{

if (number <= 1)

return 1;

else

return number \* Factorial(number - 1);

}

public static void Main(string[] args)

{

for (int i = 1; i < 101; i++)

{

Console.WriteLine("{0}! = {1}", i, Factorial(i));

}

Console.ReadLine();

}

}

}

11.   Write a program that solves the following tasks:

-     Put the digits from an integer number into a reversed order.

-     Calculate the average of given sequence of numbers.

-     Solve the linear equation **a \* x + b = 0**.

Create appropriate **methods** for each of the above tasks.

Make the program show a **text menu** to the user. By choosing an option of that menu, the user will be able to choose which task to be invoked.

Perform validation of the input data:

-     The integer number must be a positive in the range [1…50,000,000].

-     The sequence of numbers cannot be empty.

-     The coefficient **a** must be non-zero.

12.   Write a method that calculates the sum of two polynomials with integer coefficients, for example **(3x2 + x - 3) + (x - 1) = (3x2 + 2x - 4)**.

13.   \* Write a method that calculates the product of two polynomials with integer coefficients, for example **(3x2 + x - 3) \* (x - 1) = (3x3 - 2x2 - 4x + 3)**