1.Write a program, which creates an array of **20 elements of type integer** and initializes each of the elements with a value equals to the index of the element multiplied by 5. Print the elements to the console.

using System;

namespace Detyra1

{

class Program

{

static void Main(string[] args)

{

int[] arr = new int[20];

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

{

arr[i] = i \* 5;

Console.WriteLine(arr[i]);

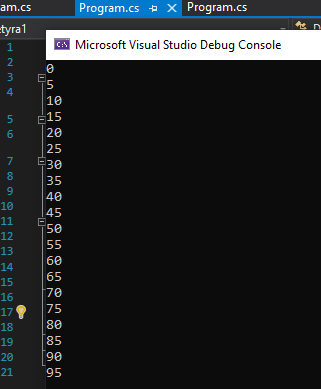
}

Console.ReadKey();

}

}

}



2.    Write a program, which **reads two arrays** from the console and **checks whether they are equal** (two arrays are equal when they are of equal length and all of their elements, which have the same index, are equal).

using System;

namespace Detyra2

{

class Program

{

static void Main(string[] args)

{

bool arraysEqual = true;

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

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

int[] arrA = new int[length];

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

{

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

arrA[i] = int.Parse(Console.ReadLine());

}

Console.Write("\nEnter length of second array: ");

if (length != Int32.Parse(Console.ReadLine())) Console.WriteLine("\nArrays

have different lengths.");

else

{

int[] arrB = new int[length];

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

{

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

arrB[i] = int.Parse(Console.ReadLine());

}

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

{

if (arrA[i] != arrB[i])

{

Console.WriteLine("\nArrays are different.");

arraysEqual = false;

break;

}

}

if (arraysEqual) Console.WriteLine("\nArrays are the same.");

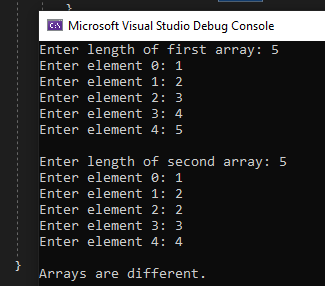
}

Console.ReadKey();

}

}

}



3.    Write a program, which **compares two arrays of type char lexicographically** (character by character) and checks, which one is first in the lexicographical order.

using System;

namespace Detyra3

{

class Program

{

static void Main(string[] args)

{

bool arrayEqual = true;

char[] arrA = new char[5] { 'a', 'b', 'c', 'd', 'e' };

char[] arrB = new char[5] { 'a', 'b', 'c', 'd', 'e' };

if (arrA.Length > arrB.Length) Console.WriteLine("Second array is lexicographicaly first.");

else if (arrA.Length < arrB.Length) Console.WriteLine("First array is lexicographicaly first.");

else

{

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

{

if (arrA[i] < arrB[i])

{

Console.WriteLine("First array is lexicographicaly first.");

arrayEqual = false;

break;

}

if (arrA[i] > arrB[i])

{

Console.WriteLine("Second array is lexicographicaly first.");

arrayEqual = false;

break;

}

}

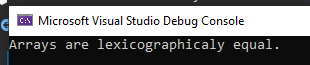
if (arrayEqual) Console.WriteLine("Arrays are lexicographicaly equal.");

}

}

}

}



4.    Write a program, which finds the **maximal sequence** **of consecutive equal elements** in an array. E.g.: {1, 1, 2, 3, **2, 2, 2**, 1} à {2, 2, 2}.

using System;

namespace Detyra4

{

class Program

{

static void Main(string[] args)

{

int count = 1, tempCount = 1, number = 0;

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());

}

for (int i = 0; i < arr.Length - 1; i++)

{

if (arr[i] == arr[i + 1]) tempCount++;

else tempCount = 1;

if (tempCount > count)

{

count = tempCount;

number = arr[i];

}

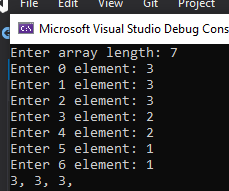
}

for (int i = 0; i < count; i++) Console.Write("{0}, ", number);

}

}

}



5.    Write a program, which finds the **maximal sequence** of consecutively placed **increasing** integers. Example: {3, **2, 3, 4**, 2, 2, 4} à {2, 3, 4}.

using System;

namespace Detyra5

{

class Program

{

static void Main(string[] args)

{

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

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

int[] arr = new int[length];

int sames = 1, bestSames = 1, bestStart = 0, lastElement = 0;

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

{

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

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

}

for (int i = 0; i < arr.Length - 1; i++)

{

if (arr[i] + 1 == arr[i + 1])

{

sames++;

if (sames > bestSames)

{

bestSames = sames;

lastElement = i + 1;

bestStart = lastElement - bestSames + 1;

}

}

else sames = 1;

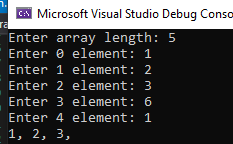
}

for (int i = bestStart; i < bestSames + bestStart; i++) Console.Write("{0}, ", arr[i]);

}

}

}



6.    Write a program, which finds the **maximal sequence of increasing elements** in an array **arr[n]**. It is not necessary the elements to be consecutively placed. E.g.: {9, 6, **2**, 7, **4**, 7, **6**, 5, **8**, 4} à {2, 4, 6, 8}.

using System;

namespace Detyra6

{

class Program

{

static void Main(string[] args)

{

int counter = 0, tempIndex, tempCounter;

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

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

int[] arr = new int[length];

int[] result = new int[length];

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

{

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

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

}

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

{

int[] tempResult = new int[length];

tempIndex = tempCounter = 1;

tempResult[0] = arr[i];

for (int j = i + 1; j < length; j++)

{

if (arr[j] > tempResult[tempIndex - 1])

{

tempResult[tempIndex] = arr[j];

tempIndex++;

tempCounter++;

}

else if (tempIndex > 1 && arr[j] > tempResult[tempIndex - 2] && arr[j] < tempResult[tempIndex - 1]) tempResult[tempIndex - 1] = arr[j];

}

if (counter < tempCounter)

{

counter = tempCounter;

result = tempResult;

}

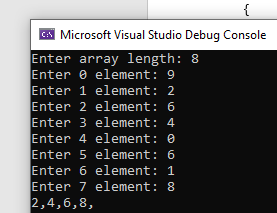
}

for (int i = 0; i < counter; i++) Console.Write("{0},", result[i]);

}

}

}



7.    Write a program, which reads from the console two integer numbers **N** and **K** (K<N) and array of N integers. Find those **K consecutive elements**in the array, which have**maximal sum**.

using System;

namespace Detyra7

{

class Program

{

static void Main(string[] args)

{

int sum = 0;

Console.Write("Enter N: ");

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

Console.Write("Enter K (K < N): ");

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

int[] arr = new int[n];

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

{

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

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

}

Array.Sort(arr, (a, b) => b.CompareTo(a));

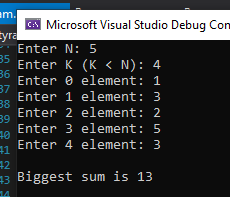
for (int i = 0; i < k; i++) sum += arr[i];

Console.WriteLine("\nBiggest sum is {0}", sum);

}

}

}



8.    **Sorting an array** means to arrange its elements in an increasing (or decreasing) order. Write a program, which sorts an array using the algorithm "**selection sort**".

using System;

namespace Detyra8

{

class Program

{

static void Main(string[] args)

{

int i, j, iMin, temp;

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

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

int[] arr = new int[length];

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

{

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

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

}

for (j = 0; j < length - 1; j++)

{

iMin = j;

for (i = j + 1; i < length; i++) if (arr[i] < arr[iMin]) iMin = i;

if (iMin != j)

{

temp = arr[j];

arr[j] = arr[iMin];

arr[iMin] = temp;

}

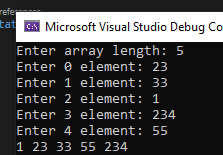
}

for (i = 0; i < length; i++) Console.Write("{0} ", arr[i]);

}

}

}



9.    Write a program, which finds a **subsequence of numbers with maximal sum**. E.g.: {2, 3, -6, -1, **2, -1, 6, 4**, -8, 8} à **11**

using System;

namespace Detyra9

{

class Program

{

static void Main(string[] args)

{

Console.Write("Length:");

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

int[] arr = new int[length];

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

{

Console.Write("Fill array[{0}]:", i + 1);

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

}

int start = 0, end = 0, max = 0, current = 0, index = 0;

for (int i = 1; i < length - 1; i++)

{

if (current > 0)

{

current += arr[i];

}

else

{

current = arr[i];

index = i;

}

if (current > max)

{

max = current;

start = index;

end = i;

}

}

Console.Write("{");

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

{

Console.Write(arr[i] + ", ");

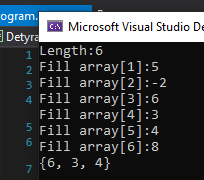
}

Console.WriteLine(arr[end] + "}");

}

}

}



10.   Write a program, which finds the **most frequently occurring** element in an array. Example: {**4**, 1, 1, **4**, 2, 3, **4**, **4**, 1, 2, **4**, 9, 3} à 4 (5 times).

using System;

namespace Detyra10

{

class Program

{

static void Main(string[] args)

{

int bestCount = int.MinValue;

int bestNumber = 0;

Console.Write("Length of array: ");

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

int[] arr = new int[length];

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

{

Console.Write("Fill array[{0}]: ", i + 1);

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

}

Array.Sort(arr);

int count = 1;

int number = 0;

for (int i = 0; i < length - 1; i++)

{

if (arr[i] == arr[i + 1])

{

count++;

number = arr[i];

if (bestCount < count)

{

bestCount = count;

bestNumber = number;

}

}

else

{

count = 1;

number = 0;

}

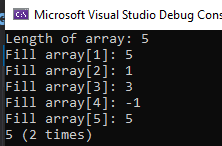
}

Console.WriteLine("{0} ({1} times)", bestNumber, bestCount);

}

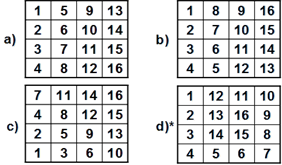
}

}



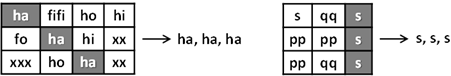
11.   Write a program to find a sequence of neighbor numbers in an array, which has a **sum of certain number S**. Example: {4, 3, 1, **4, 2, 5**, 8}, S=**11** à {4, 2, 5}.

12.   Write a program, which creates **square matrices** like those in the **figures below** and prints them formatted to the console. The size of the matrices will be read from the console. E.g. matrices with size of 4 x 4:

[](https://introprogramming.info/wp-content/uploads/2013/07/clip_image0257.png)

13.   Write a program, which creates a rectangular array with size of **n** by **m**elements. The dimensions and the elements should be read from the console. Find a **platform with size of (3, 3) with a maximal sum**.

14.   Write a program, which finds the **longest sequence of equal** **string** elements in a matrix. A sequence in a matrix we define as a set of neighbor elements **on the same row, column or diagonal**.

[](https://introprogramming.info/wp-content/uploads/2013/07/clip_image0277.png)

15.   Write a program, which creates an array containing **all Latin letters**. The user inputs **a word** from the console and as result the program prints to the console the **indices of the letters from the word**.

16.   Write a program, which uses a **binary** **search** in a **sorted**array of integer numbers to find a certain element.

17.   Write a program, which sorts an array of integer elements using a **"merge** **sort"** algorithm.

18.   Write a program, which sorts an array of integer elements using a "**quick** **sort"**algorithm.

19.   Write a program, which finds **all prime numbers** in the range [1…10,000,000].

20.   \* Write a program, which checks whether there is a **subset**of given array of **N** elements, which has a **sum S**. The numbers **N**, **S** and the array values are read from the console. Same number can be used many times.

Example: {2, **1, 2**, 4, 3, **5**, 2, **6**}, **S** = **14** à yes (1 + 2 + 5 + 6 = 14)

21.   Write a program which by given **N** numbers, **K** and **S**, finds **K** elements out of the **N**numbers, the sum of which is exactly **S** or says it is not possible.

Example: {3, **1,**2, **4**, **9**, 6}, **S** = **14**, **K** = **3** à yes (1 + 2 + 4 = 14)

22.   Write a program, which reads an array of integer numbers from the console and **removes a minimal number of elements** in such a way that **the remaining array is sorted** in an increasing order.

Example: {6, **1,** 4, **3**, 0, **3**, 6, **4**,**5**} à {1, 3, 3, 4, 5}

23.   Write a program, which reads the integer numbers **N** and **K** from the console and prints **all variations of K elements of the numbers in the interval** **[1…N]**. Example: N = 3, K = 2 à {1, 1}, {1, 2}, {1, 3}, {2, 1}, {2, 2}, {2, 3}, {3, 1}, {3, 2}, {3, 3}.

24.   Write a program, which reads an integer number **N** from the console and prints **all** **combinations of K elements of numbers in range** **[1** **…** **N]**. Example: N = 5, K = 2 à {1, 2}, {1, 3}, {1, 4}, {1, 5}, {2, 3}, {2, 4}, {2, 5}, {3, 4}, {3, 5}, {4, 5}.

25.   \*Write a program, which finds in a given matrix the **largest area of equal numbers**. We define an **area** in the matrix as a set of neighbor cells (by row and column). Here is one example with an area containing 13 elements with equal value of 3:

[](https://introprogramming.info/wp-content/uploads/2013/07/clip_image0297.png)