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]);

}

}

}

}

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

int[] arrA = new int[length];

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

{

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

arrA[i] = Int32.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] = Int32.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.");

}

}

}

}

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.

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}.

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}.

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}.

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**.

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]);

}

}

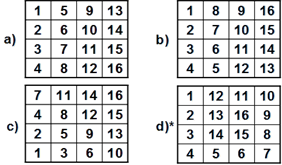
}

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**

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).

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)

using System;

namespace Detyra12

{

class Program

{

static void Main(string[] args)

{

Console.Write("Enter height: ");

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

Console.Write("Enter width: ");

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

int a = 0;

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

{

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

a += i;

for (int j = 1; j < x; j++)

{

a += y;

Console.Write("{0} ", a);

}

a = 0;

Console.WriteLine();

}

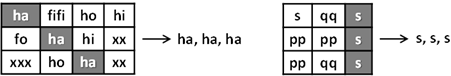
}

}

}

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**.

using System;

namespace Detyra15

{

class Program

{

static void Main(string[] args)

{

char[] alphabet = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z' };

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

char[] word = (Console.ReadLine()).ToCharArray();

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

for (int j = 0; j < alphabet.Length; j++)

if (word[i] == alphabet[j]) Console.Write("{0} ", j);

}

}

}

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.

using System;

namespace Detyra17

{

class Program

{

static public void DoMerge(int[] numbers, int left, int mid, int right)

{

int[] temp = new int[25];

int i, left\_end, num\_elements, tmp\_pos;

left\_end = (mid - 1);

tmp\_pos = left;

num\_elements = (right - left + 1);

while ((left <= left\_end) && (mid <= right))

{

if (numbers[left] <= numbers[mid]) temp[tmp\_pos++] = numbers[left++];

else temp[tmp\_pos++] = numbers[mid++];

}

while (left <= left\_end) temp[tmp\_pos++] = numbers[left++];

while (mid <= right) temp[tmp\_pos++] = numbers[mid++];

for (i = 0; i < num\_elements; i++)

{

numbers[right] = temp[right];

right--;

}

}

static public void MergeSort\_Recursive(int[] numbers, int left, int right)

{

int mid;

if (right > left)

{

mid = (right + left) / 2;

MergeSort\_Recursive(numbers, left, mid);

MergeSort\_Recursive(numbers, (mid + 1), right);

DoMerge(numbers, left, (mid + 1), right);

}

}

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 < length; i++)

{

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

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

}

MergeSort\_Recursive(arr, 0, arr.Length - 1);

for (int i = 0; i < arr.Length; i++) Console.WriteLine(arr[i]);

}

}

}

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)