ARRAYS

THEORY

Arrays are a handy way to store many variables of the same type. They work as an ordered list. It’s important to know that you can’t change the number of items stored inside the array once it’s initialized.

You can think of any element of the array as a variable

PRACTICE

To declare an array use the syntax

type[] arrayName;

It’s very similar to the declaration of a standard variable, but after the type you have to write []

To initialize you can use any of those syntaxes

arrayName = new type[N] {variable1, variable2, [...], variableN};

arrayName = new type[] {variable1, variable2, [...], variableN};

arrayName = {variable1, variable2, [...], variableN};

arrayName = new type[N];

The first 3 create an array containing N elements all defined by the values inside the curly brackets, separeted by commas. The last one just creates the array of size N, but doesn’t initialize the elements inside it.

Remember that you can’t recall a variable until you have initialized it.

Now that we have declared our array, we wat to access those elements, to do so

arrayName[index]

The index insed the square brackets is the 0-based position of the element inside the array

e.g.

string[] fruits={“apple”,”peach”,”orange”};

string apple = fruits[0]; //the apple variable has the value “apple”

string peach=fruits[1]; //the peach variable has the value “peach”

fruits[2] = “pear”; //replaces orange with pear

REMEMBER THAT ARRAYS ARE 0-BASED WHILE THE LENGTH IS 1-BASED, SO AN ARRAY CONTAINING 2 ELEMENTS HAS LENGTH 2, BUT YOU CAN RECALL INDEXES 0 AND 1

ADVANCED

To get the number of elements inside and array use the syntax

arrayName.Lenght

why are arrays useful? Why can’t you just use many variables? Well apart from the elegance of the array and the easiness of understanding of the code (that could be a good enough reason), you can iterate through a array.

e.g.

string[] fruits={“apple”,”peach”,”orange”};

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

Console.WriteLine(fruits[i] + “ is a fruit”);

}

The for cycle will work no matter how many fruits we put inside the array.

You can join all the elements of an array into one single string using the method

Array.Join(separator, arrayName)

ASSIGNEMENT

Create a program that writes the first 70 numbers of the Fibonacci sequence, separated by a comma and a space

TIPS:

* The Fibonacci sequence is defined like this: the first 2 elements are 1 and 1, every other element is equal to the sum of the previous two
* You should create an array containing all the numbers, no need to instantiate every element of the array at the start
* Fill the array using a for loop
* Once you filled the array you can write all the elements using the Array.join function