#!markdown

<span style="font-size:16px;">

<h1 style="font-family:Copperplate">Arrays</h1>

<code>Arrays</code> are C# data structures that store multiple instances of the same type of item: e.g., <code>ints</code>: <kbd>[1, 2, 3]</kbd>, <code>doubles</code>: <kbd>[1.2, 0.144579, 63.0]</kbd>, or <code>strings</code>: <kbd>["Hello", "World!"]</kbd>.

</span>

#!markdown

<span style="font-size:16px;">

<h4 style="font-family:Copperplate;margin-bottom:6pt;">Declaring an Array</h4>

Declaring an array is similar to declaring a variable. First, declare the type of values that the array stores. Then, denote that it is an array with <code>[]</code> like so:<br />

<kbd>int[] arrayOfIntegers;</kbd>

</span>

#!markdown

<span style="font-size:16px;">

<h4 style="font-family:Copperplate;margin-bottom:6pt;">Initializing an Array</h4>

An array’s contents can be initialized by assigning a value to each of its items:<br />

<kbd>int[] arrayOfIntegers = {1, 2, 3, 4};</kbd><br />

Alternatively, the <code>new</code> keyword specifies an array’s size and initializes its elements to a type-dependent default value. For example, unspecified <code>integers</code> are initialized to 0.<br />

<kbd>int[] arrayOfIntegers = new int[4];</kbd><br /><br />

Below are some examples.

</span>

#!csharp

// Create an array of two strings

string[] arrayOfStrings = {"Hello", "World!"};

// Create an array of four integers, and add values later

int[] arrayOfIntegers = new int[4];

// You can also use the "new" keyword like this:

double[] arrayOfDoubles = new double[3] {1.2, 0.144579, 63.0};

// Or this:

char[] arrayOfCharacters = new char[] {'a', 'Z', '.'};

//This prints each array:

Console.WriteLine($"The array of strings: \t\t[{string.Join(", ", arrayOfStrings)}]\nThe array of integers: \t\t[{string.Join(", ", arrayOfIntegers)}]\nThe array of doubles: \t\t[{string.Join(", ", arrayOfDoubles)}]\nThe array of characters: \t[{string.Join(", ", arrayOfCharacters)}]\n");

#!markdown

<span style="font-size:16px;">

<h4 style="font-family:Copperplate;margin-bottom:6pt;">Accessing Array Values</h4>

<code>Array</code> elements are accessed using <i>indexing</i>. Each of an array’s value is assigned an index. Its first element is assigned 0, and the succeeding elements are assigned indexes in <i>ascending order</i>.<br />

In C#, array elements are accessed using <code>[i]</code> where i is the index of the element to access. Given the array <kbd><code>AgeOfCows</code> = [15, 4, 8, 9, 3]</kbd>, the following code will print 8 to the console:<br />

<kbd>Console.Write(AgeOfCows[2]);</kbd><br /><br />

Indexing can also be used to alter an array’s elements. The following code changes the value 9 to 10.<br />

<kbd>AgeOfCows[3] = 10;</kbd>

</span>

#!markdown

Reading Excercises here;

Split new file below

#!markdown

<span style="font-size:16px;">

<h4 style="font-family:Copperplate;margin-bottom:6pt;">Arrays in Memory</h4>

<img src="https://raw.githubusercontent.com/FLAMINGxFURY/TextbookTemp/main/Chapter%201%20-%20Array%20and%20Array%20Algorithms/Assets/ArraysInMemory.png" style="width:90%;"/><br />

<a href="https://codeforwin.org/2017/10/c-arrays-declare-initialize-access.html" style="font-size:11px;">Source: Codeforwin.org</a>

</span>

#!markdown

<span style="font-size:16px;">

Content tbd

</span>