CECS 174 – Lecture 20 – Arrays

Array – An array is a sequence or collection of values of the same type.

Declaring an array-

```
int [] array1 = new int[5];
```

or

```
int [] array2 = {1, 2, 3, 4, 5};
```

array2	
[0]	1
[1]	2
[2]	3
[3]	4
[4]	5

Both of these declarations create space for five integers in an array. The first declaration, array1, reserves the space for these integers and initializes them all to 0. The second declaration, array2, creates the array and initializes each of the integers to the values specified. Any data type or object can be used to create an array.

```
String [] strArray = new String [3];
```

This will create three empty (null) strings in an array.

Creating a Dynamically Allocated Array -

If you do not know how many elements you will have in the array and you want the user to specify how many there will be, you can use a variable to declare the size of the array.

```
int input = in.nextInt();
int [] array3 = new int [input];
```

Whatever the value of the variable input at the time of the array's declaration is what the size of the array will be.

Indexing an array-

To access specific elements of your array you must identify the index of that element.

Placing a value into an element:

```
int val1 = 11;
array1[0] = 5;
array1[1] = 7;
array1[2] = 9;
array1[3] = val1;
```

array1	
[0]	5
[1]	7
[2]	9
[3]	11
[4]	0

Using the contents of an individual element of an array:

```
System.out.println( "The first element of array1 is: " + array1[0]);

int value = array1[2] + 4;
```

Using For loops with arrays:

For loops provide us with an excellent way of accessing each element of an array.

```
for (int i=0; i<5; i++){
      System.out.println("Item " + i + ": " + array1[i]);
}
```

or use the array's built-in length function -

```
for (int i=0; i<array1.length; i++) {
      System.out.println("Item " + i + ": " + array1[i]);
}
```

or use the for each loop –

```
int i = 0;
for (int x : array1){
      System.out.println("Item " + i + ": " + x);
      i++;
}
```

All three versions give the same output:

```
/*Output
Item 0: 5
Item 1: 7
Item 2: 9
Item 3: 11
Item 4: 0
*/
```

Using Arrays in Methods:

Arrays are passed by reference, as such, when you change any of the values of an array from inside a method, the values of the original array are changed as well.

```
//method calls in main
int [] arr2 = newArray();
changeArray(arr2);

public static void changeArray(int [] arr){
      arr[0] = 5;
      arr[1] = 10;
}
public static int[] newArray(){
      int [] a = {1,2,3,4,5};
      return a;
}
```

arr2	
[0]	5
[1]	10
[2]	3
[3]	4
[4]	5