JAVA Arrays Recap

- An array can be thought of as a piece of named memory that is split into a certain number of partitions.
- Arrays hold similar data (i.e. data type), but can have different values
- For example, if I had an array of five (5) people’s names I would do the following (in this case in the main method)

```java
public class ArrayExample {
    public static void main (String [] args) {
        // Below is one way to declare an array as a set
        String [] names = {"Ali", "Bob", "Chris", "Denise", "Eric"};

        // you can also just allocate memory as shown below and then
        // assign the values at a later time.
        String [] names1 = new String [5];

        names[0] = "Ali";
        names [4] = "Eric";
    }
}
```

- The values stored in the Array are called elements
- We access the elements in the array by referring to them through their subscript or index
- From a visual standpoint the array called `names` would look like this:

<table>
<thead>
<tr>
<th></th>
<th>Ali</th>
<th>Bob</th>
<th>Chris</th>
<th>Denise</th>
<th>Eric</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

- Notice the numbers below each element - these are called subscripts or indices (index)
In Java (and many other languages) we start counting at 0. The length would be n-1, where n is the total number of elements in the array.

System.out.println(names.length); would print 5

If we tried to access names[5] we would get an index out of bounds runtime error

We can change values at anytime by referencing the index and assigning the element a new value:

names[2] = “Christine”; // it used to store a value of Chris

**Accessing the Elements in an Array Using Loops**

Loops are the best way to access a range of values in an array. Otherwise, you would have to write many statements instead of a simple loop. A for loop works the best for this application because you can directly define the range in the loop header.

To loop through the **names** loop you could do the following:

```java
for (int i = 0; i < names.length; i++) {
    System.out.println(names[i]);
}
```

This would print each element in that array on a new line. You could, of course, get creative and concatenate other information to your print out.

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

Check out the following reading at Ohio State University for some more details on Arrays.