More about Arrays

Traversing an Array with a For Loop

To access all the elements in an array, we can use a for loop:

```
int[] scores = {80, 92, 91, 68. 88};

for (int i = 0; i < scores.length; i++ ){

    System.out.println(scores[i]);

}
```

With this loop, we access each element by using its index value. As i increments, we are able to go through all of the values.

Traversing an Array with a While Loop

To access all the elements in an array, we can use a while loop:

```
int[] scores = {80, 92, 91, 68. 88};

int i = 0;
while (i < scores.length){

    System.out.println(scores[i]);

    i++;
}
```

Break Loop

Given an array of integers. Find the index value where the target number is 91. When you find it, print the index.

```
int[] scores = {80, 92, 91, 68, 88};

int target = 91;


int i = 0;

while (i < scores.length){
    if (scores[i] == target)
        break;
    i ++;
}
System.out.println("The target was found at: " + i);
```

Enhanced For Loops

It is an alternate method to traverse an array instead of using for or while loops.

It is a simplified, but less flexible way to loop through a collection of items, such as Arrays.

It is referred as a For-Each loop and it starts with the first element of the array and continues through in order to the last element of the array.

Structure of an Enhance For Loop

```
int[] scores = {80, 92, 91, 68, 88}


for (int score: scores)
{
    System.out.println(score);
}
```

For Loops vs. Enhanced For Loop

Why would you use a Standard For Loop?

- A for loop uses a counter variable which sometimes needed in your loop.

Why would you use an Enhanced For Loop?

- Simplified structures, especially good when using nested loops.
- Easier to write.

Print an array

```
int[] scores = {80, 92, 91, 68, 88};

System.out.println(scores);
```

What is printed?

```
[I@439f5b3d
```

How would you write a method to print the scores array like this:

```
"[80, 92, 91, 68, 88]"
```

Print an array

```
import java.util.Arrays;

int[] scores = {80, 92, 91, 68, 88};

Arrays.toString(scores); => "[80, 92, 91, 68, 88]"
```

Create and Return - No variable declaration needed

Method Signature

```
public int[] getNumbers() {
   return new int[] {1, 2, 4};
}
```