AP Computer Science A: Traversing Arrays

CodeHS

Recap

Arrays store a fixed number of elements of the same type. Once an array has been created, the size cannot change.

```
// Make a new int array and set the values

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

// Make a new int array with default values

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

Accessing Array Values

We can access an array value by referencing its index:

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

System.out.println(scores[0]);
```

Often times arrays can have hundreds, thousands, or even millions of values. We need a systematic way to access all of these values.

Traversing an Array

Systematically accessing all of our elements in an array is called **traversing the array**.

Using loops, we can easily cycle through all of our array values.

Iterating Over an Array

As we step through this loop, the value of `i` changes, allowing us to access each element of the array.

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

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

{

   // This prints out the ith element!

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

}
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 1

Value of i: 0

Value of scores[i]:

Console output:

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Console output:

Loop Iteration: 1

Value of i: 0

Value of scores[i]: 80

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 1

Value of i: 0

Value of scores[i]: 80

Console output:

```
80
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 2

Value of i: 1

Value of scores[i]:

Console output:

```
80
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 2

Value of i: 1

Value of scores[i]: 92

Console output:

```
80
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 2

Value of i: 1

Value of scores[i]: 92

Console output:

```
80
92
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 3

Value of i: 2

Value of scores[i]:

Console output:

```
80
92
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 3

Value of i: 2

Value of scores[i]: 91

Console output:

```
80
92
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 3

Value of i: 2

Value of scores[i]: 91

Console output:

```
80
92
91
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 4

Value of i: 3

Value of scores[i]:

Console output:

```
80
92
91
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 4

Value of i: 3

Value of scores[i]: 68

Console output:

```
80
92
91
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 4

Value of i: 3

Value of scores[i]: 68

Console output:

```
80
92
91
68
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 5

Value of i: 4

Value of scores[i]:

Console output:

```
80
92
91
68
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 5

Value of i: 4

Value of scores[i]: 88

Console output:

```
80
92
91
68
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration: 5

Value of i: 4

Value of scores[i]: 88

Console output:

```
80
92
91
68
88
```

Iterating Over an Array

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

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

{

    // This prints out the ith element!

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

}
```

Loop Iteration:

Value of i: 5

Value of scores[i]:

Console output:

```
80
92
91
68
88
```

Let's Look At Another Example

Given: An Array of integers

Problem: Find the index value where the target number of 91 exists.

Solution With a While Loop

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

int index = 0;

int target = 91;

while (index < scores.length)

{

    if (scores[index] == target)

        break;

    index ++;

}

System.out.println("The target was found at: " + index);
```

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 1

Value of index: 0

Value of scores[i]:

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 1

Value of index: 0

Value of scores[i]: 80

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 1

Value of index: 1

Value of scores[i]: 80

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 2

Value of index: 1

Value of scores[i]:

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 2

Value of index: 1

Value of scores[i]: 92

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 2

Value of index: 2

Value of scores[i]: 92

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 3

Value of index: 2

Value of scores[i]:

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Loop Iteration: 3

Value of index: 2

Value of scores[i]: 91

Console output:

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Console output:

Loop Iteration: 2

Value of index: 2

Value of scores[i]: 92

Iterating Over an Array

```
int[] scores = {80, 92, 91, 68, 88};
int index = 0;
int target = 91;

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

Loop Iteration: 2

Value of index: 2

Value of scores[i]: 92

Console output:

```
The target was found at 2
```

Caution: Beware of Index Out Of Bounds!

In both cases, we looped to a value *less than* the array length:

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

```
while (count < scores.length)
```

If our `array.length` is 5, the last value of our loop will stop at 4 and we would have accessed our last value.

Caution: Beware of Index Out Of Bounds!

If we loop to less than or equal and our `array.length` is 5:

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

```
while (count <= scores.length)
```

The last value in our loop will be 5 which will result in an `ArrayIndexOutOfBoundsException` being thrown.

Now It's Your Turn!

CodeHS

Concepts Learned this Lesson

Term	Definition
Traversing an Array	Traversing an array is the process to loop through an array and access each of the elements. Caution must be taken to avoid looping beyond the valid index values.

Standards Covered

- (LO) VAR-2.B Traverse the elements in a 1D array.
- (EK) VAR-2.B.1 Iteration statements can be used to access all the elements in an array. This is called traversing the array.
- (EK) VAR-2.B.2 Traversing an array with an indexed for loop or while loop requires elements to be accessed using their indices.
- (EK) VAR-2.B.3 Since the indices for an array start at 0 and end at the number of elements − 1, “off by one” errors are easy to make when traversing an array, resulting in an ArrayIndexOutOfBoundsException being thrown.