AP Computer Science A: Enhanced for Loop for Arrays

CodeHS

Recap: Traversing an Array

To access all the elements in an array, we can use a 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 our values.

Introducing Enhanced For Loops

An Enhanced For Loop is an alternate method to traverse an array instead of using for or while loops

CodeHS

Enhanced For Loop

An Enhanced For Loop is a simplified, but less flexible way to loop through a collection of items, including Arrays.

It is often referred to 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.

Enhanced For Loop

We can rewrite our For Loop or While Loops as an Enhanced For Loop:

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


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

Enhanced For Loop

Structure of an Enhanced For Loop:

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

Enhanced For Loop Variable

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

Enhanced For Loop

Structure of an Enhanced For Loop:

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

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

Enhanced For Loop Variable

Existing Array Variable

Enhanced For Loop

Structure of an Enhanced For Loop:

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

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

Enhanced For Loop Variable - Copy of value from the Array

Enhanced For Loop

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


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

Output

Variable Values:

scores
int object

score

Enhanced For Loop

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


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

Output

Variable Values:

scores
int object

score
80

Enhanced For Loop

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


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

Output

```
80
```

Variable Values:

scores
int object

score
80

Enhanced For Loop

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


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

Variable Values:

scores
int object

score
92

Output

```
80
```

Enhanced For Loop

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


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

Variable Values:

scores
int object

score
92

Output

```
80
92
```

Enhanced For Loop

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


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

Output

```
80
92
```

Variable Values:

```
scores
int object
```

```
score
91
```

Enhanced For Loop

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


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

Output

```
80
92
91
```

Variable Values:

scores
int object

score
91

Enhanced For Loop

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


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

Output

```
80
92
91
```

Variable Values:

scores
int object

score
68

Enhanced For Loop

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


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

Output

```
80
92
91
68
```

Variable Values:

scores
int object

score
68

Enhanced For Loop

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


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

Output

```
80
92
91
68
```

Variable Values:

scores
int object

score
88

Enhanced For Loop

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


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

Output

```
80
92
91
68
88
```

Variable Values:

scores
int object

score
88

Enhanced For Loop

Enhanced for loops provide an efficient way to access objects.

```
Student[] classroom = {julian, larisa, amada, mikka, jay};
```

Standard For Loop:

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

Enhanced For Loop:

```
for(Student student : classroom)
{
    System.out.println(student.getFirstName());
}
```

For Loop vs Enhanced For Loop

Why would you use an **Enhanced For Loop?**

- Enhanced for loops offer a simplified structure and are especially good when using nested loops

- They tend to be easier to write

For Loop vs Enhanced For Loop

Why would you use an **Standard For Loop?**

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

- Since enhanced for loops only make a copy with no reference to the index, they are not optimal if you need to update values in the array

Now It’s Your Turn!

CodeHS

Concepts Learned this Lesson

Term	Definition
Enhanced For Loop	A loop that is an alternate to a for or while loop that accesses each value in an array starting at the first value and proceeding in order.
Enhanced For Loop Variable	Variable created in the enhanced for loop header that contains a copy of the array variable.

Standards Covered

- (LO) VAR-2.C Traverse the elements in a 1D array object using an enhanced for loop.
- **(EK)** VAR-2.C.1 An enhanced for loop header includes a variable, referred to as the enhanced for loop variable.
- (EK) VAR-2.C.2 For each iteration of the enhanced for loop, the enhanced for loop variable is assigned a copy of an element without using its index.
- (EK) VAR-2.C.3 Assigning a new value to the enhanced for loop variable does not change the value stored in the array.
- (EK) VAR-2.C.4 Program code written using an enhanced for loop to traverse and access elements in an array can be rewritten using an indexed for loop or a while loop.