



**Computer Science Virtual Learning**

# **HS Computer Science A**

**May 12th, 2020**



Lesson: **Looping with the For-Each Loop**

**Objective/Learning Target:**

Understanding what a For-Each Loop is and how to apply it using Java

# What is a For-Each Loop?

You will often loop through all of the elements of an array (to get the average or to get each one to display). You will typically do this using a for-each loop. A for-each loop is a loop that can only be used on a collection of items. It will loop through the collection and each time through the loop it will use the next item from the collection. It starts with the first item in the array (the one at index 0) and continues through in order to the last item in the array.

A screenshot of a Java IDE window titled "Original - 1 of 1". It contains a Java class named "Test1" with a "getAvg" method and a "main" method. The "getAvg" method uses a for-each loop to calculate the average of an array of integers. The "main" method creates an array of integers {2, 6, 7, 12, 5} and calls the "getAvg" method. The output of the program is displayed in a text area below the code, showing the value "6.4".

```
1 public class Test1
2 {
3     public static double getAvg(int[] values)
4     {
5         double total = 0;
6         for (int val : values)
7         {
8             total = total + val;
9         }
10        return total / values.length;
11    }
12
13    public static void main(String[] args)
14    {
15        int[] values = {2, 6, 7, 12, 5};
16    }
17 }
```

6.4

Activity: 1 -- ActiveCode (lcaf1)

# What is a For-Each Loop?

The for-each loop is shown on line 6. It says to loop through the array called values and each time through the loop set the variable (val) to the next item in the array. We have to specify the type of val first since this declares a variable. The type must match the type of objects in the array.

Only use the for-each loop when you want to loop through all the values in an array or list. If you only want to loop through part of an array or list use a for loop instead. Also use a for loop instead of a for-each loop if you want to change any of the values in the array or list.

A screenshot of a Java IDE. At the top right, there is a "Run" button and a progress indicator showing "Original - 1 of 1". The code editor contains the following Java code:

```
1 public class Test1
2 {
3     public static double getAvg(int[] values)
4     {
5         double total = 0;
6         for (int val : values)
7         {
8             total = total + val;
9         }
10        return total / values.length;
11    }
12
13    public static void main(String[] args)
14    {
15        int[] values = {2, 6, 7, 12, 5};
```

Below the code editor, a text box displays the output "6.4". At the bottom of the IDE, it says "Activity: 1 -- ActiveCode (lcaf1)".



# How does a For-Each loop affect Object-Oriented Programming?

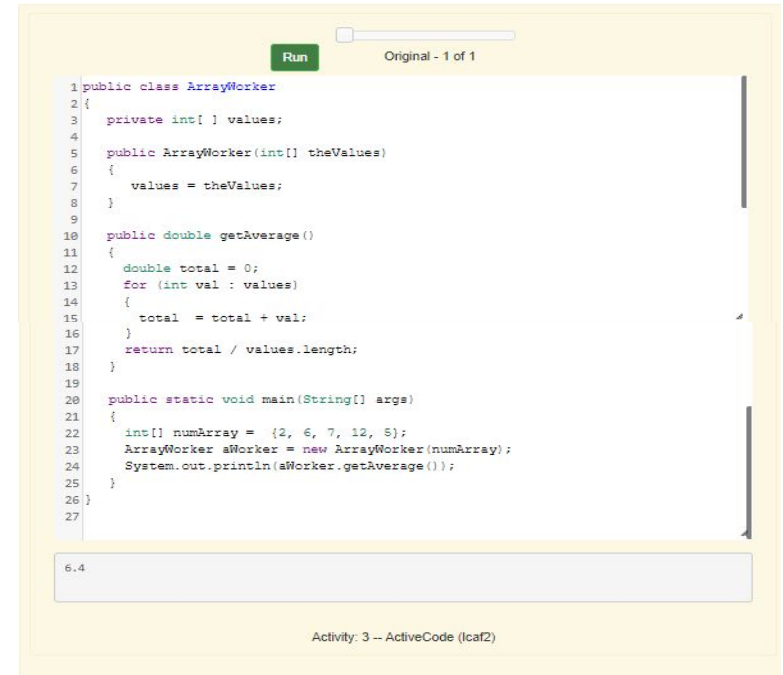
The code in the slides above wasn't object-oriented. You may have noticed that it was declared to be static. This means that it is a class method not an object method. It is a class method since it doesn't operate on any object fields - all data that it needs has been passed in to the method. Class methods can be called using `ClassName.methodName()`. They can also be called on an object of the class. Object methods can only be called on an object of the class.

# Object - Oriented Programming

A more object-oriented way of doing this would be if the array was a field called `values` in the same class as the `getAverage` method. Then you don't need to pass the array values to the method and the method is an object (instance) method since it operates on the fields of the object. You will typically initialize fields in the constructor as shown below.

Notice that we have to create an object of the class now in the main method. Object methods have to be called on an object of the class.

Since `values` is an object field and the method `getAverage` is in the same class it can directly access the field `values`. The code could have also been written as `this.values` to indicate the current object's field called `values`. Every object method is passed the object the method was called on and it can be referenced using the Java keyword `this`.

A screenshot of a Java IDE window titled "Original - 1 of 1". It shows a code editor with Java code for an `ArrayWorker` class. The code includes a constructor, a `getAverage` method, and a `main` method. A green "Run" button is visible above the code. Below the code editor, a text area displays the output "6.4". At the bottom of the IDE window, it says "Activity: 3 -- ActiveCode (lcaf2)".

```
1 public class ArrayWorker
2 {
3     private int[] values;
4
5     public ArrayWorker(int[] theValues)
6     {
7         values = theValues;
8     }
9
10    public double getAverage()
11    {
12        double total = 0;
13        for (int val : values)
14        {
15            total = total + val;
16        }
17        return total / values.length;
18    }
19
20    public static void main(String[] args)
21    {
22        int[] numArray = {2, 6, 7, 12, 8};
23        ArrayWorker aWorker = new ArrayWorker(numArray);
24        System.out.println(aWorker.getAverage());
25    }
26 }
27
```

6.4

Activity: 3 -- ActiveCode (lcaf2)

# Check Your Understanding

1. Given that `a` is an array of integers and `val` is an integer value, which of the following best describes the conditions under which the following code segment will return true?

```
boolean temp = false;
for ( int i = 0; i < a.length; i++)
{
    temp = ( a[i] == val );
}
return temp;
```

- Whenever the first element in `a` is equal to `val`
- Whenever `a` contains any element which equals `val`.
- Whenever the last element in `a` is equal to `val`.
- Whenever only 1 element in `a` is equal to `val`.

3. Given the following field and method, which of the following best describes the contents of `myStuff` after `(int m = mystery(n);)` has been executed?

```
// private field in the class
private int[ ] myStuff;
//precondition: myStuff contains
// integers in no particular order
public int mystery(int num)
{
    for (int k = myStuff.length - 1; k >= 0; k--)
    {
        if (myStuff[k] < num)
        {
            return k;
        }
    }
    return -1;
}
```

- All values in positions `m+1` through `myStuff.length-1` are greater than or equal to `n`.
- All values in position `0` through `m` are less than `n`.
- All values in position `m+1` through `myStuff.length-1` are less than `n`.



## For More Resources and to Check Answers

Go to: <https://runestone.academy/runestone/books/published/apcsareview/ArrayBasics/aForEach.html>