

Professional Skills
Agile Fundamentals
Jira
Git
Databases Introduction
Java Beginner
<input checked="" type="radio"/> What is Java?
<input checked="" type="radio"/> Installation
<input checked="" type="radio"/> Hello World Example
<input checked="" type="radio"/> Data Types
<input checked="" type="radio"/> Packages
<input checked="" type="radio"/> Naming Conventions Cheat Sheet
<input checked="" type="radio"/> Flow of Control
<input checked="" type="radio"/> Class Members
<input checked="" type="radio"/> Operators
<input checked="" type="radio"/> Conditionals
<input checked="" type="radio"/> Iteration
<input checked="" type="radio"/> Arrays
<input checked="" type="radio"/> ArrayList
<input checked="" type="radio"/> Enhanced For Loops
<input type="radio"/> String Manipulation
<input type="radio"/> Class Constructors
<input type="radio"/> Access Modifiers
<input type="radio"/> Installing Java & Maven To PATH
<input type="radio"/> Object-Oriented Programming Principles
<input type="radio"/> Encapsulation
<input type="radio"/> Inheritance
<input type="radio"/> Polymorphism
<input type="radio"/> Abstraction
<input type="radio"/> Interfaces
<input type="radio"/> Type Casting
<input type="radio"/> Static
<input type="radio"/> Final
<input type="radio"/> Garbage Collection
<input type="radio"/> Input With Scanner
<input type="radio"/> Pass by Value/Reference
<input type="radio"/> JUnit

Enhanced For Loops

Contents

- [Overview](#)
- [Tutorial](#)
 - [Syntax](#)
 - [Example](#)
- [Exercises](#)

Overview

In Java, you have two types of **for** loops; basic **for** loops and enhanced **for** loops, otherwise known as **for-each** loops. Enhanced **for** loops are better suited to iterating through arrays and collections of data. The reason we use enhanced **for** loops, as opposed to basic **for** loops for looping through arrays and collections, is because it makes our code more readable, therefore making maintaining it much easier. Enhanced **for** loops are also better used when iterating through an **entire** data set whereas basic **for** loops are better when we want to partially iterate over a data set due to being able to access the index.

Tutorial

Syntax

```
for(DataType item : array) {  
  
}
```

The above example shows the syntax of an enhanced for loop. We declare the **for** loop the same way we would a basic **for** loop, however the syntax within the parentheses changes slightly. We first specify the data type that the array or collection holds, and then give it a reference variable name that it can put the value at the current iteration into. We then use a colon ":" and specify the array or collection that we want to iterate through.

Example

```
public void printArray(String[] stringArray) {  
    for(String str : stringArray) {  
        System.out.println(str);  
    }  
}
```

In the above example, we iterate through **stringArray** pass the value to **str**, and then execute the body of the loop, in this case, print the value stored in **str**. The variable **str** only stores the value at the current iteration of the loop, once the body of the loop has been executed, the next value in **stringArray** will be stored for the next execution of the method body.

Exercises

- Create an array of strings and iterate through it, printing each value to console, using an enhanced **for** loop.
- Create an array of integers 1-20 and iterate through it, using an enhanced **for** loop, square, and then print each value.

<div><div></div><div>Test Driven Development</div></div> <div><div></div><div>UML Basics</div></div> <div><div></div><div>JavaDoc</div></div> <div><div></div><div>Peer Programming</div></div> <div><div></div><div>Code Reviews</div></div>
Maven
Testing (Foundation)
Java Intermediate
HTML
CSS
Javascript
Spring Boot
Selenium
Sonarqube
Advanced Testing (Theory)
Cucumber
MongoDB
Express
NodeJS
React
Express-Testing
Networking
Security
Cloud Fundamentals
AWS Foundations
AWS Intermediate
Linux
DevOps
Jenkins Introduction
Jenkins Pipeline
Markdown
IDE Cheatsheet

3. Create a method that returns a boolean and accepts an integer as a parameter, if the integer is even, return true, if not then return false.
4. Using the array of integers from exercise 2 and the method created in exercise 3; iterate through the array using an enhanced for loop, calling the method from exercise 3 in the body.

◦ If the value is even, cube it, then print it to console.

◦ If the value is odd, square it, then print it to console.

