* Static Variables

When a variable is declared as static, then a single copy of the variable is created and shared among all objects at the class level. Static variables are, essentially, global variables. All instances of the class share the same static variable.

* Static Methods

A static method is a method that belongs to a [class](https://blog.hubspot.com/website/what-is-a-class-in-java?_ga=2.242790726.531217907.1666296918-1447620525.1666296918) rather than an instance of a class. This means you can call a static method without creating an object of the class. Static methods are sometimes called class methods.

* Static block

The static block executes when class loader loads the class. A static block is invoked before main () method.

* toString method

If you want to represent any object as a string, **toString() method** comes into existence.

The toString() method returns the String representation of the object.

If you print any object, Java compiler internally invokes the toString() method on the object. So overriding the toString() method, returns the desired output, it can be the state of an object etc. depending on your implementation.

* Passing parameters:

### Types of parameters:

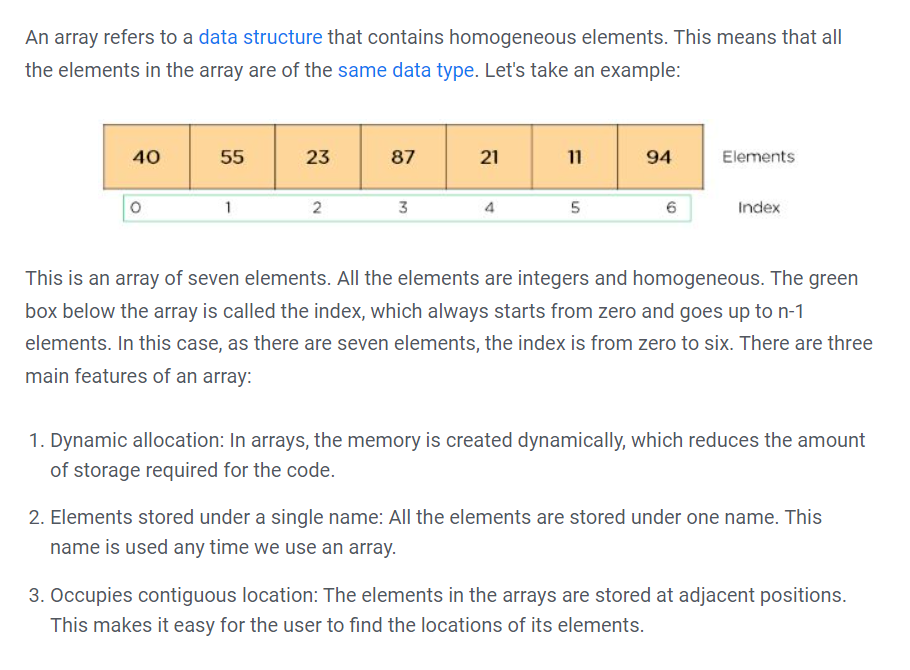
**Formal Parameter:** A variable and its type as they appear in the prototype of the function or method.

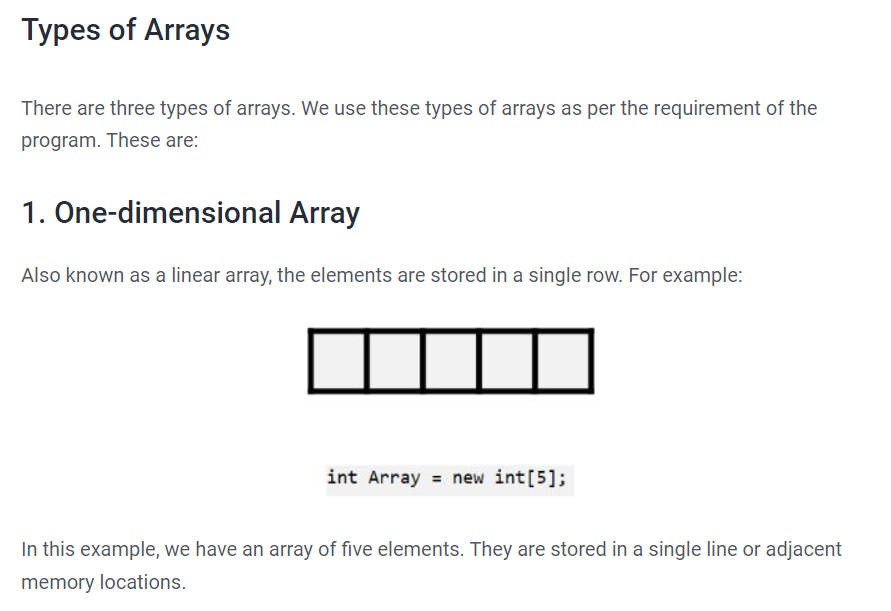
**Syntax:**

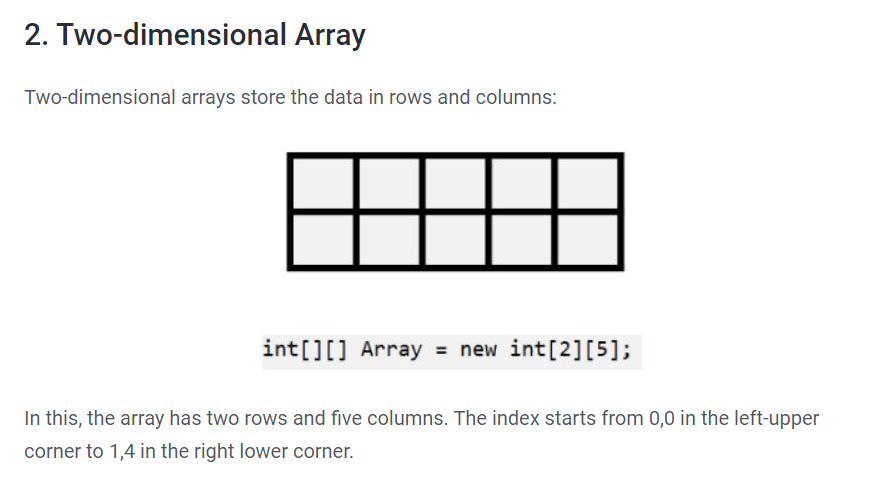
function\_name(datatype variable\_name)

**Actual Parameter:** The variable or expression corresponding to a formal parameter that appears in the function or method call in the calling environment.

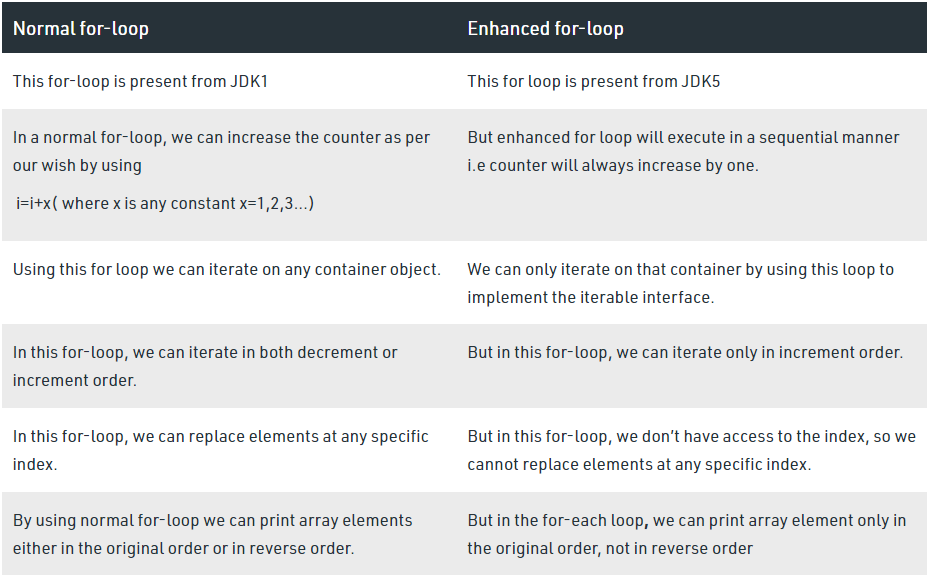
* Arrays:





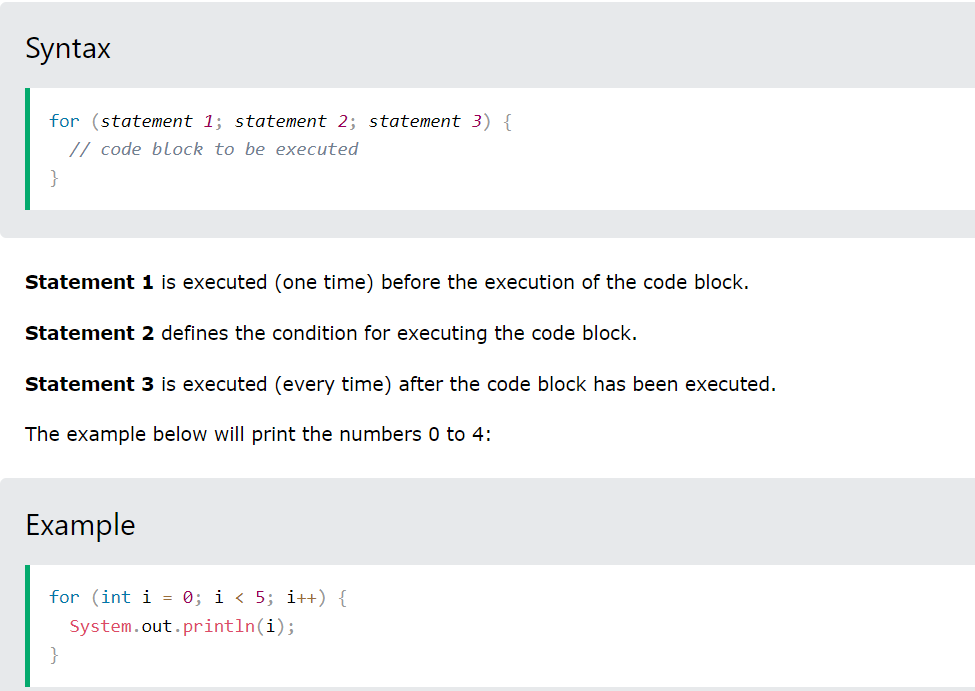


Difference Between for loop and enhanced for loop



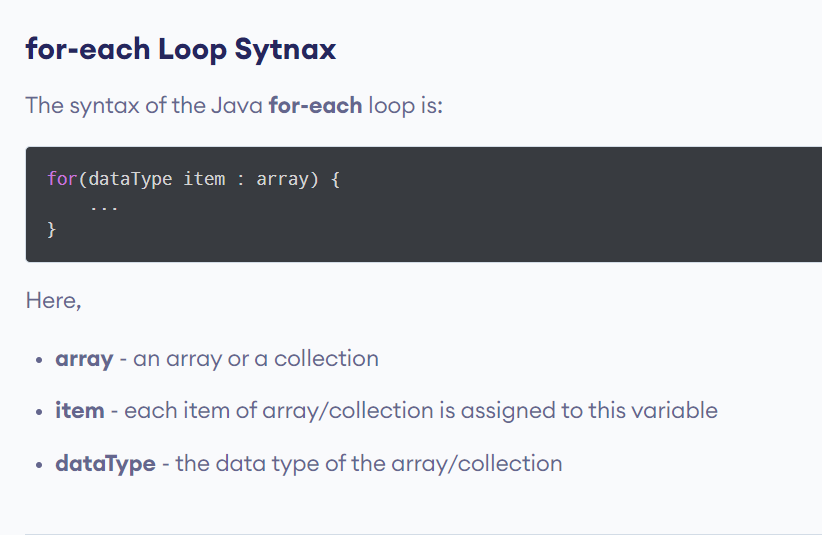
Graphical user interface, text, application

Description automatically generated



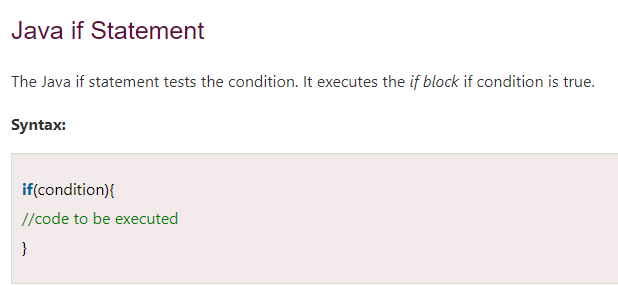
Graphical user interface, text, application, email

Description automatically generated



Text

Description automatically generated



Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email, website

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Logo

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

A picture containing company name

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Logo, company name

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated with medium confidence

Table

Description automatically generated with low confidence

* varArgs:

Let’s suppose you are creating a [Java method](https://www.programiz.com/java-programming/methods). However, you are not sure how many arguments your method is going to accept. To address this problem, Java introduced varargs.

Varargs is a short name for variable arguments. In Java, an argument of a method can accepts arbitrary number of values. This argument that can accept variable number of values is called varargs.