# CORE JAVA

MANUAL V8.3

**MODULE CODE:**

**ANUDIP FOUNDATION**


ICONS AND THEIR MEANING
HINTS:
Get ready for helpful insites on difficult topics and questions.
STUDENTS:
This icon symbolize important instrcutions and guides for the students.
TEACHERS/TRAINERS:
This icon symbolize important instrcutions and guides for the trainers.

## Module 1: Java Fundamental and Programming Concepts

### Chapter 7

| | |
|---|---|
| **Objective:** After completing this lesson you will be able to :<br>* Get familiar with Java loops<br>* Learn about continue and break statements<br>* Gain an idea about Java user-defined functions | **Materials Required:**<br>1. Computer<br>2. Internet access |
| **Theory Duration:** 60 minutes | **Practical Duration:** 60 minutes |
| **Total Duration:** 120 minutes | |

# Chapter 7

## 7.1 Loops (For, While, Do-While)

In Java, loops are used to repeatedly execute statement sets until a specific condition becomes true. Looping enables the performing of functions and instructions within a Java program.

A loop is used to repeat a block of programming code, till the time when a particular condition is met. Loops are used to avoid rewriting the same code multiple times throughout a program. Performing repetitive tasks becomes easier with the use of Java loops.

**The three types of Java loops are -**

i) For loop

ii) While loop

iii) Do-while loop

**i) For loop -** A For loop is used when the exact number of intended repetitions are known to the programmer. It is a flow control statement that contains the initialization, testing condition and value of increment/decrement in a single line.

The initialization of a for loop occurs only once, followed by the testing condition evaluation for each repetition. A statement is executed if a condition is 'true', and not executed if it is 'false'. Each repetition executed is counted into a loop counter by the increment/decrement part of the code.

The main categories of for loops are -

* Simple for loop
* Enhanced for loop
* Infinitive for loop

**For loop syntax:**

```
for(initialization; condition ; increment/decrement)
{
   statement(s);
}
```

**Example of For loop**

```
class ForLoopExample {
    public static void main(String args[]){
        for(int i=5 i12; i--){
            System.out.println('The value of i is: '+i);
        }
    }
}
```

**Output:**

```
The value of i is: 5
The value of i is: 4
The value of i is: 3
```

**ii) While loop –** A While loop is used when the exact number of intended repetitions are not known to the programmer. This loop is a control flow statement that executes blocks of code based on a provided boolean condition. A while loop is also known as an Entry Control Loop, as the condition is checked first before any execution happens.

A while loop starts with checking the condition first. A statement is executed if the condition is 'true', and not executed if 'false'. Loops are terminated if the condition becomes 'false'. A programming statement is executed as long as (or while) the condition remains 'true'.

* **While loop syntax:**

```
while(condition)
{
   statement(s);
}
```

* **Example of While loop**

```
public class WhileExample {
        public static void main(String[] args) {

        int i=1;

        while(i<=4){

        System.out.println(i);

        i++;

           }

        }

        }
```

**Output:**

```
1
2
3
4
```

**ii) Do-while loop –** A Java do-while loop is a control flow statement capable of executing a program section once or more times. The number of execution repetitions are based on a designated boolean condition.

A do-while loop is preferred when a loop has to be run once or more, but the total number of repetitions is 'not fixed'. This loop has to be executed for a minimum of one time, as a condition is verified after the looping. A do-while loop is also known as an Exit Control Loop, as it first executes one (or more) statements, before checking for a condition.

*** Do-while loop syntax:**

```
do
{
   statements..
}
while (condition);
```

*** Do-while loop example**

```
class DoWhileLoopExample {
   public static void main(String args[]){
        int i=7;
        do{
            System.out.println(i);
            i--;
        }while(i!3);
   }
}
```

**Output:**

```
7
6
```

5

4

**Difference between While and Do-While Loop**

| WHILE | DO-WHILE |
|---|---|
| while ( condition) {<br>statements; //body of loop<br>} | do{<br>.<br>statements; // body of loop.<br>.<br>} while( Condition ); |
| In 'while' loop the controlling condition appears at the start of the loop. | In 'do-while' loop the controlling condition appears at the end of the loop. |
| The iterations do not occur if, the condition at the first iteration, appears false. | The iteration occurs at least once even if the condition is false at the first iteration. |

**7.2 Continue and Break Statement**

**i) Continue statement**

A continue statement in Java is used for continuing a loop. It ensures that a program's flow is constant, overlooking pieces of code based on provided conditions. This statement can be used with a 'while', 'for' and 'do-while' loop. Using a 'continue' statement is ideal in cases where the next repetition has to be executed immediately after the current.

If 'continue;' is specified for a particular value or values, it will not be executed. Instead, the loop is continued for executing the next unspecified values. If a continue statement is used for an inner loop, the inner loop is continued.

**Syntax:**

continue;

**Continue statement example –**

```
public class ContinueExample {

   public static void main(String args[]){
          for (int j=1; j<=7; j++)

          {
          if (j==3)
          {
                continue;
             }

          System.out.print(j+' ');
          }
   }
}
```

**Output:**

1 2 4 5 6 7

**ii) Break statement**

In Java, a break statement is used for breaking a switch statement or loop. Conditions have to be specified for a break statement to interrupt a program's flow. A loop is ended immediately when a break statement is found within it. The program control moves forward to the next statement following a loop break. A break statement can be utilised to break for loop, do-while loop and while loop.

**Syntax:**

```
jump-statement;
break;
```

**Break statement example –**

```
public class BreakExample {
    public static void main(String[] args) {
    using for loop


    for(int i=2;i<=8;i++){
    if(i==6){
        break;
      }
      System.out.println(i);
   }
  }
  }
```

**Output:**

```
2
3
4
5
```

## 7.3 User Defined Functions (Return, Non-return type, Parameterize)

### User Defined Functions

Java is an object-oriented programming (OOP) language platform that offers support for most OOP rules. It enables programmers to utilize their own functions apart from predefined Java functions. Functions can be defined by users, or created from the basic functions offered by Java.

### User defined functions rules -

* Function names cannot contain Java reserved keywords
* Existing function names cannot be overloaded

* Names must be within 128 characters
* Names can only contain numbers, letters and underscores, and start with letters.

**returnType** - A method can return a value of native data types (float, double, int, etc.), native objects (Map, List, String etc.), and other default and user-defined objects. Return type is considered void (non returnType) if the method does not produce a value.

**Parameterize** - Parameterized or parameterized type in Java is a type that utilises other types to produce an output.

### Practical (60 minutes)

**a)** See the example programme for Java for-loop below. Write the same programme for i=10 and i16. Show the resulting output.

```
class ForLoopExample {
   public static void main(String args[]){
      for(int i=5 i12; i--){
```

```
    System.out.println('The value of i is: '+i);
      }
  }
}
```

**b)** See the example programme for Java do-while loop below. Write the same programme for i = 11 and 13. Show the resulting output.

```
class DoWhileLoopExample {
  public static void main(String args[]){
      int i=7;
      do{
          System.out.println(i);
          i--;
      }while(i13);
  }
}
```

**Instructions: The progress of students will be assessed with the exercises mentioned below.**

**MCQ**

1. Loops are used to repeatedly _____ sets of statements.

a) execute

b) enumerate

c) embolden

d) None of the mentioned

2. What is used to repeat a block of programming code?

a) A value

b) An array

c) A loop

d) None of the mentioned

3. The three loop types in Java are for loop, do-while loop and _____ loop.

a) whole

b) when

c) while

d) None of the mentioned

4. A for loop initialization occurs only _____.

a) infinite times

b) thrice

c) twice

d) once

5. Which loop is used when the exact number of repetitions are known?

a) for loop

b) do-while loop

c) while loop

d) None of the mentioned

6. A while loop is also known as an _____ Control Loop.

a) Entry

b) Exit

c) Intermediate

d) None of the mentioned

7. A Java do-while loop is a _____ flow statement.

a) connector

b) console

c) control

d) None of the mentioned

8. What loop type has to be executed at least once even if repetitions are not fixed?

a) do-while

b) for

c) while

d) None of the mentioned

9. A _____ statement is used to ensure the constant flow of a program.

a) concatenate

b) collaborate

c) continue

d) None of the mentioned

10. A break statement is used for breaking a _____ statement.

a) Switch

b) switcher

c) selector

d) None of the mentioned