# Java Control Statements

## 1) Simple if statement:

It is the most basic statement among all control flow statements in Java. It evaluates a Boolean expression and enables the program to enter a block of code if the expression evaluates to true.

**Syntax** of if statement is given below.

```
if(condition)
{
statement 1;
}
```

PROGRAM 1-

```
public class Student
 {
        public static void main(String[] args)
        {
            int x = 10;
            int y = 12;
            if(x+y > 20)
              {
                  System.out.println("x + y is greater than 20");
               }
        }
  }
```

PROGRAM 2-

```
public class Student
 {
```

```
        public static void main(String[] args)
        {
            int x = 10;
            if(x%2==0)
              {
                  System.out.println("Number is Even");
               }
        }
 }
```

## 2) if-else statement

The if-else statement is an extension to the if-statement, which uses another block of code, i.e., else block. The else block is executed if the condition of the if-block is evaluated as false.

## Syntax:

```
if(condition)
{
statement 1; //executes when condition is true
}
else
{
statement 2; //executes when condition is false
}
```

Program 1-

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

```
  {
        int x = 10;
        int y = 12;
        if(x+y < 10)
       {
              System.out.println("x + y is less than      10");
        }
 else
      {
          System.out.println("x + y is greater than 20");
       }
     }
}
```

## 3) if-else-if ladder:

The if-else-if statement contains the if-statement followed by multiple else-if statements. In other words, we can say that it is the chain of if-else statements that create a decision tree where the program may enter in the block of code where the condition is true. We can also define an else statement at the end of the chain.

**Syntax of if-else-if statement is given below.**

```
if(condition 1)
 {
statement 1; //executes when condition 1 is true
}
else if(condition 2)
{
```

```
statement 2; //executes when condition 2 is true
}
else
 {
statement 2; //executes when all the conditions are false
 }
```

**Program -**

```
public class Student
 {
        public static void main(String[] args)
        {
            String city = "Delhi";
            if(city == "Meerut")
               {
                    System.out.println("city is meerut");
                }
           else if (city == "Noida")
                {
                        System.out.println("city is noida");
                 }
          else if(city == "Agra")
                 {
                    System.out.println("city is agra");
                  }
          else
            {
```

```
                    System.out.println(city);
            }
        }
}
```

## Java Nested if statement

The nested if statement represents the if block within another if block. Here, the inner if block condition executes only when outer if block condition is true.

**Syntax:**

```
if(condition){
    //code to be executed
        if(condition){
            //code to be executed
    }
}
```

### Example:

//Java Program to demonstrate the use of Nested If Statement.

```
public class JavaNestedIfExample
{
public static void main(String[] args)
{
   int age=20;
   int weight=80;
   if(age>=18)
{
```

```
      if(weight>50)
{
         System.out.println("You are eligible to donate blood");
      }
   }
}
}
```

## Example 2:

//Java Program to demonstrate the use of Nested If Statement.

```
public class JavaNestedIfExample2
{
public static void main(String[] args)
{
      int age=25;
   int weight=48;
      if(age>=18)
{
      if(weight>50)
{
         System.out.println("You are eligible to donate blood");
      } else
{
         System.out.println("You are not eligible to donate blood");
      }
   } else
{
```

```
    System.out.println("Age must be greater than 18");
  }
}
}
```