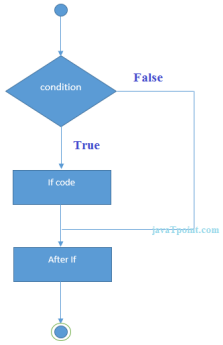
If...else I




```
1  if(condition)
2  {
3        //code to be executed
4  }
```

If...else II

```java
//Java Program to demonstate the use of if statement.
public class IfExample
{
      public static void main(String[] args)
      {
            //defining an 'age' variable
            int age=20;
            //checking the age
            if(age>18)
            {
                  System.out.print("Age is greater than 18");
            }
      }
}
```

if-else statement:

✓ The Java if-else statement also tests the condition. It executes the if block if condition is true otherwise else block is executed.

If...else III

```
if(condition)
{
      //code if condition is true
}
else
{
      //code if condition is false
}
```

```
//A Java Program to demonstrate the use of if-else statement.
//It is a program of odd and even number.
public class IfElseExample
{
      public static void main(String[] args)
      {
            //defining a variable
            int number=13;
            //Check if the number is divisible by 2 or not
            if(number%2==0)
            {
                  System.out.println("even number");
            }
            else
```

If...else IV

```
15              {
16                    System.out.println("odd number");
17              }
18        }
19  }
```

✓ Leap Year Example:

A year is leap, if it is divisible by 4 and 400. But, not by 100.

```
1   public class LeapYearExample
2   {
3         public static void main(String[] args)
4         {
5               int year=2020;
6               if(((year % 4 ==0) && (year % 100 !=0)) || (year % 400==0))
7               {
8                     System.out.println("LEAP YEAR");
9               }
10              else
11              {
12                    System.out.println("COMMON YEAR");
13              }
14        }
```

```
15 }
```

Using Ternary Operator:

✓ We can also use ternary operator (? :) to perform the task of if...else statement. It is a shorthand way to check the condition. If the condition is true, the result of ? is returned. But, if the condition is false, the result of : is returned.

```
1  public class IfElseTernaryExample
2  {
3        public static void main(String[] args)
4        {
5              int number=13;
6              //Using ternary operator
7              String output=(number%2==0)?"even number":"odd number";
8              System.out.println(output);
9        }
10 }
```

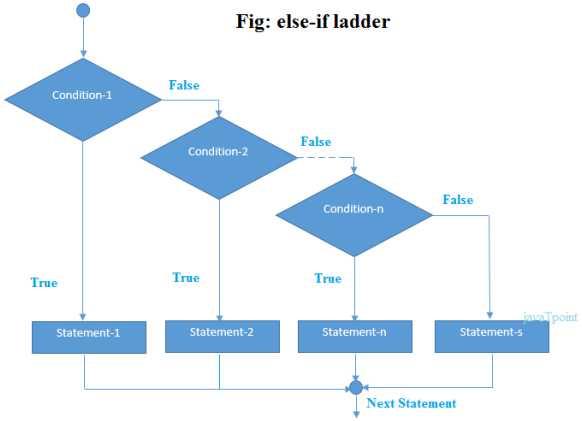
Java if-else-if ladder Statement:

The if-else-if ladder statement executes one condition from multiple statements.

```
if(condition1)
{
      //code to be executed if condition1 is true
}
else if(condition2)
{
      //code to be executed if condition2 is true
}
else if(condition3)
{
      //code to be executed if condition3 is true
}
...
else
{
      //code to be executed if all the conditions are false
}
```

If...else VII

Fig: else-if ladder



If...else VIII

```java
//Java Program to demonstrate the use of If else-if ladder.
//It is a program of grading system for fail, D grade, C grade, B grade, A grade and A+.
public class IfElseIfExample
{
      public static void main(String[] args)
      {
            int marks=65;

            if(marks<50)
            {
                  System.out.println("fail");
            }
            else if(marks>=50 && marks<60)
            {
                  System.out.println("D grade");
            }
            else if(marks>=60 && marks<70)
            {
                  System.out.println("C grade");
            }
            else if(marks>=70 && marks<80)
            {
                  System.out.println("B grade");
```

If...else IX

```
24              }
25              else if(marks>=80 && marks<90)
26              {
27                    System.out.println("A grade");
28              }
29              else if(marks>=90 && marks<100)
30              {
31                    System.out.println("A+ grade");
32              }
33              else
34              {
35                    System.out.println("Invalid!");
36              }
37        }
38  }
```

If...else X

✓ Program to check POSITIVE, NEGATIVE or ZERO:

```
1  public class PositiveNegativeExample
2  {
3        public static void main(String[] args)
4        {
5              int number=-13;
6              if(number>0)
7              {
8                    System.out.println("POSITIVE");
9              }
10             else if(number<0)
11             {
12                   System.out.println("NEGATIVE");
13             }
14             else
15             {
16                   System.out.println("ZERO");
17             }
18       }
19 }
```

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.

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

If...else XII

✓ Example:

```
1  //Java Program to demonstrate the use of Nested If Statement.
2  public class JavaNestedIfExample
3  {
4        public static void main(String[] args)
5        {
6              //Creating two variables for age and weight
7              int age=20;
8              int weight=80;
9              //applying condition on age and weight
10             if(age>=18)
11             {
12                   if(weight>50)
13                   {
14                         System.out.println("You are eligible to donate blood");
15                   }
16             }
17       }
18
19 }
```

Java while and do-while loop I

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

'condition' must be a boolean value. We cannot pass any other value.

```
public class WhileExample
{
      public static void main(String[] args)
      {
            int i=1;
            while(i<=10)
            {
                  System.out.println(i);
                  i++;
            }
      }
}
```

Java while and do-while loop II

do-while loop:

```
do
{
      //code to be executed
}
while(condition);
```

✓ Example:

```
public class DoWhileExample
{
      public static void main(String[] args)
      {
            int i=1;
            do
            {
                  System.out.println(i);
                  i++;
            }
            while(i<=10);
      }
}
```

Java for loop I

✓ A simple for loop is the same as C/C++. We can initialize the variable, check condition and increment/decrement value. It consists of four parts:

1. Initialization: It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
2. Condition: It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
3. Statement: The statement of the loop is executed each time until the second condition is false.

Java for loop II

4. Increment/Decrement: It increments or decrements the variable value. It is an optional condition.

Syntax:

```
for(initialization; condition; incr/decr)
{
      //statement or code to be executed
}
```

'condition' must be a boolean value. We cannot pass any other value.

Java for loop III

```java
//Java Program to demonstrate the example of for loop
//which prints table of 1
public class ForExample
{
      public static void main(String[] args)
      {
            //Code of Java for loop
            for(int i=1; i<=10; i++)
            {
                  System.out.println(i);
            }
      }
}
```

Java for loop IV

```java
// Calculate the sum of all elements of an array

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

            // an array of numbers
            int[] numbers =
            {
                  3, 4, 5, -5, 0, 12
            }
            ;
            int sum = 0;

            // iterating through each element of the array
            for (int number: numbers)
            {
                  sum += number;
            }

            System.out.println("Sum = " + sum);
      }
```

Java for loop V

```
24 }
```

```
1  public class Main
2  {
3        public static void main(String[] args)
4        {
5
6              char[] vowels =
7              {
8                    'a', 'e', 'i', 'o', 'u'
9              }
10             ;
11
12             // iterating through an array using a for loop
13             for (int i = 0; i < vowels.length; ++ i)
14             {
15                   System.out.println(vowels[i]);
16             }
17       }
18 }
```

Java for loop VI

✓ Using for-each Loop:

```
for(dataType item : array)
{
      ...
}
```

Here,

- array - an array or a collection
- item - each item of array/collection is assigned to this variable
- dataType - the data type of the array/collection

Java for loop VII

```
1  public class Main
2  {
3        public static void main(String[] args)
4        {
5
6              char[] vowels =
7              {
8                    'a', 'e', 'i', 'o', 'u'
9              }
10             ;
11
12             // iterating through an array using the for-each loop
13             for (char item: vowels)
14             {
15                   System.out.println(item);
16             }
17       }
18 }
```

while, do-while and for loop I

Comparison	for loop	while loop	do while loop
Introduction	The Java for loop is a control flow statement that iterates a part of the programs multiple times.	The Java while loop is a control flow statement that executes a part of the programs repeatedly on the basis of given boolean condition.	The Java do while loop is a control flow statement that executes a part of the programs at least once and the further execution depends upon the given boolean condition.
When to use	If the number of iteration is fixed, it is recommended to use for loop.	If the number of iteration is not fixed, it is recommended to use while loop.	If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use the do-while loop.
Syntax	for(init;condition;incr/decr){ // code to be executed }	while(condition){ //code to be executed }	do{ //code to be executed }while(condition);
Example	//for loop for(int i=1;i<=10;i++){ System.out.println(i); }	//while loop int i=1; while(i<=10){ System.out.println(i); i++; }	//do-while loop int i=1; do{ System.out.println(i); i++; }while(i<=10);
Syntax for infinitive loop	for(;;){ //code to be executed }	while(true){ //code to be executed }	do{ //code to be executed }while(true);