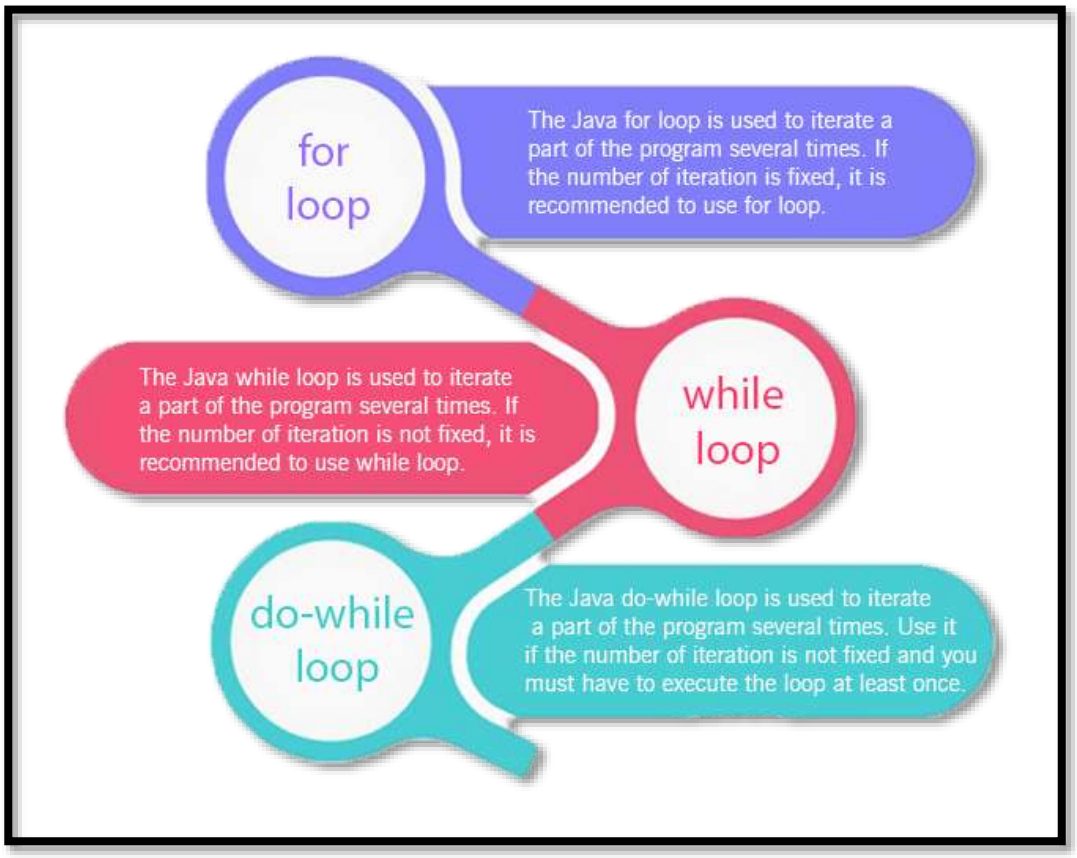
## Loops in Java

The Java for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop.

There are three types of for loops in Java.




## Java Simple for Loop

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

**Syntax:**

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

## ForExample.java

//Java Program to demonstrate the example of for loop

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

**PyramidExample.java**

```
public class PyramidExample
{
public static void main(String[] args)
 {
     for(int i=1;i<=5;i++)
       {
            for(int j=1;j<=i;j++)
             {
                 System.out.print("* ");
             }
            System.out.println();
       }
}
}
```

```
//PyramidExample2.java
public class PyramidExample2
{
    public static void main(String[] args)
   {
        int term=6;
          for(int i=1;i<=term;i++)
           {
              for(int j=term;j>=i;j--)
                {
                      System.out.print("* ");
                 }
                       System.out.println();
            }
       }
}
```

## Java for-each Loop

The for-each loop is used to traverse array or collection in Java. It is easier to use than simple for loop because we don't need to increment value and use subscript notation.

It works on the basis of elements and not the index. It returns element one by one in the defined variable.

**Syntax:**

```
for(data_type variable : array_name)
{
//code to be executed
}
```

**Example:**

ForEachExample.java

//Java For-each loop example which prints the  elements of the array

```
public class ForEachExample
 {
public static void main(String[] args)
 {
   int arr[]={12,23,44,56,78};
   for(int i:arr)
    {
       System.out.println(i);
    }
}
}
```

**Java Break Statement**

**Example**

```
public class ForExample
 {
public static void main(String[] args)
{
      for(int i=1;i<=10;i++)
   {
      if(i==5)
      {
         break;
      }
```

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

JAVA CONTINUE STATEMENT

EXAMPLE:

```
public class ForExample
{
public static void main(String[] args)
{
   for(int i=1;i<=10;i++)
   {
      if(i==5)
      {
         continue;
      }
      System.out.println(i);
   }
}
}
```

## Java While Loop

The Java while loop is used to iterate a part of the program repeatedly until the specified Boolean condition is true. As soon as the Boolean condition becomes false, the loop automatically stops.

The while loop is considered as a repeating if statement. If the number of iteration is not fixed, it is recommended to use the while loop.

**Syntax:**

```
while (condition)
{
//code to be executed
I ncrement / decrement statement
}
```

**WhileExample.java**

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

**BreakWhileExample.java**

//Java Program to demonstrate the use of break statement inside the while loop.

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

```
{
   int i=1;
   while(i<=10)
{
      if(i==5)
{
         i++;
         break;
      }
      System.out.println(i);
      i++;
   }
}
}
```

## Java Continue Statement in while loop

ContinueWhileExample.java

//Java Program to demonstrate the use of continue statement  inside the while loop.

```
public class ContinueWhileExample
 {
public static void main(String[] args)
 {
   int i=1;
   while(i<=10)
{
      if(i==5)
```

```
{


        i++;
        continue;
      }
      System.out.println(i);
      i++;
   }
}
}
```

## Java do-while Loop

The Java do-while loop is used to iterate a part of the program repeatedly, until the specified condition is true. If the number of iteration is not fixed and you must have to execute the loop at least once, it is recommended to use a do-while loop.

Java do-while loop is called an exit control loop. Therefore, unlike while loop and for loop, the do-while check the condition at the end of loop body. The Java do-while loop is executed at least once because condition is checked after loop body.

**Syntax:**

```
Do
{
//code to be executed / loop body
//update statement
}while (condition);
```

**DoWhileExample.java**

```
public class DoWhileExample
```

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

## BreakDoWhileExample.java

//Java Program to demonstrate the use of break statement  inside the Java do-while loop.

```
public class BreakDoWhileExample
{
        public static void main(String[] args)
        {
              int i=1;
           do
        {
              if(i==5)
        {
                i++;
                break;
              }
```

```
            System.out.println(i);
            i++;
        }while(i<=10);
    }
    }
```

## Java Continue Statement in do-while Loop

ContinueDoWhileExample.java

//Java Program to demonstrate the use of continue statement  inside the Java do-while loop.

```
public class ContinueDoWhileExample
{
    public static void main(String[] args)
    {
        int i=1;
        do
            {
            if(i==5)
                {
                    i++;
                continue;
            }
            System.out.println(i);
            i++;
        }while(i<=10);
    }
    }
```