

Loops

The Loops are used to repeat a block of code. Being able to have your program repeatedly execute a block of code is one of the most basic and useful tasks in programming. Loops are very useful when the number of lines or statements needs to execute more than one time.

There are 3 types of loops in JAVA Language

1. **for** loop
2. **while** loop
3. **do while** loop

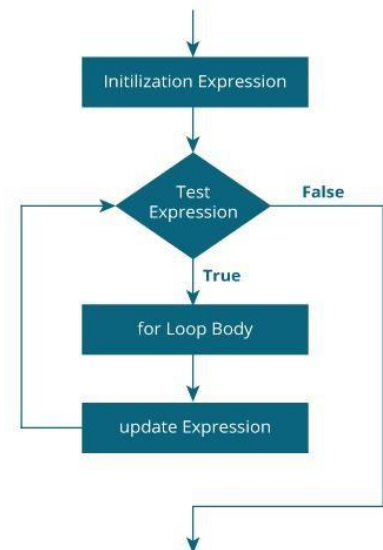
1. for Loop

for loop is used to execute a set of statements repeatedly until **for – loops** through a block of code a specified number of times. We can say it an open ended loop. It is usually preferable when the number of iterations are known as **for** loop.

Syntax: **for** (**initialization**; **test expression**; **update expression**)

```
{  
    Statement(s);  
}
```

Flow Chart:



Explanation:

The initial expression is initialized only once at the beginning of the '**for**' loop. Then, the condition is checked by the program. If the condition is false, **for** loop is terminated. But, if condition is true then, the codes are executed and update expression is updated. Again, the condition is checked. If it is false, loop is terminated and if it is true, the same process repeats until condition is false.

Example

```
int a;  
System.out.println("For Loop Example.. ");  
for ( a=5; a<=50; a+=5) // if a is less than and equals to 50  
    System.out.print(a+", " ); // then it will print value of a, then update value by 5
```

```

Output - test1 (run) X
run:
For Loop Example..
5, 10, 15, 20, 25, 30, 35, 40, 45, 50, BUILD SUCCESSFUL (total time: 0 seconds)

```

2. while loop

While loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The condition is evaluated, and if the condition is true, the code within the block is executed first. This repeats until the condition becomes false. Because **while** loop checks the condition before the block is executed, it is also known as a **pre-test loop**.

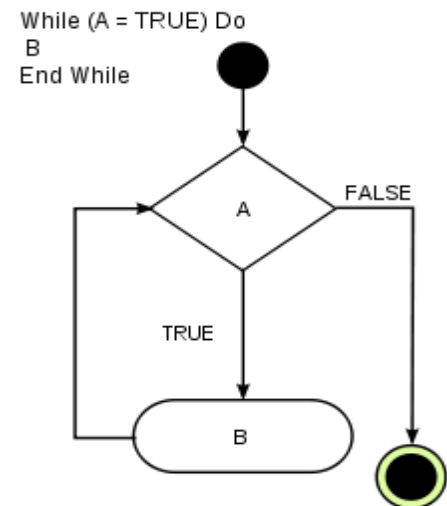
Syntax:

Flow Chart

```

Initialization;
while(condition)
{
    Statement(s);
    Update_Value;
}

```



Explanation:

first checks whether A is TRUE, which it is, so then the {loop body} is entered, where it executes statement(s) that are in the body of loop, after execution of every statement once it checks the condition again if the condition remains TRUE it will execute the body again . Process repeating until the A remains true.

Example:

```

int a=5;
System.out.println("While Loop Example. ");
while( a<=50)    // if a is less than and equals to 50
{
    System.out.print(a+", " );    // then it will print value of a, then update value by 5
    a += 5;
}

```

```

Output - test1 (run) X
run:
While Loop Example...
5, 10, 15, 20, 25, 30, 35, 40, 45, 50, BUILD SUCCESSFUL (total time: 0 seconds)

```

3) do...while loop

In JAVA, **do...while** loop is very similar to **while** loop. Only difference between these two loops is that, in **while** loops, condition is checked at first but, in **do...while** loop code is executed at first then the condition is checked. So, the code is executed at least once in **do...while** loops that's why it is also called **Post Test** loop. **Syntax:**

```

Initialization;
do
{
    Statement(s);
    Update value;
} while (condition);

```

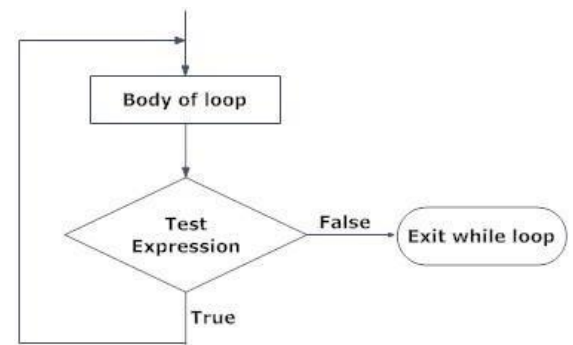


Figure: Flowchart of do...while loop

Notice, there is semicolon in the end of **while** (); in **do...while** loop.

Explanation:

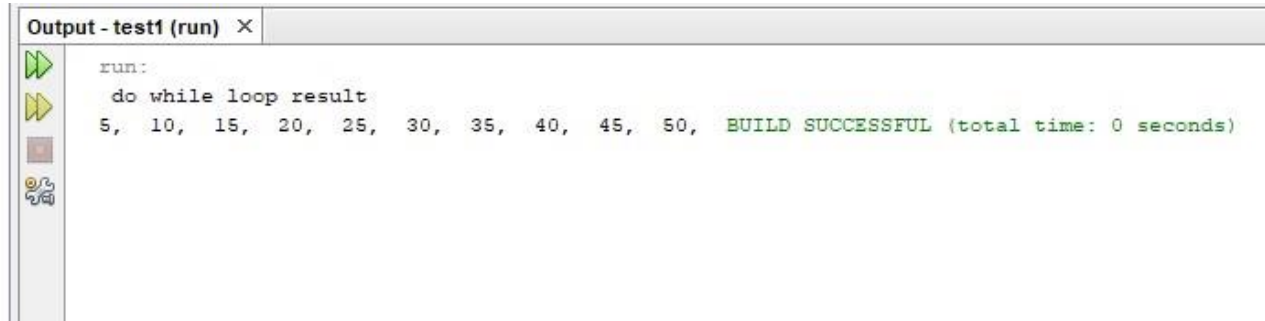
At first codes inside body of **do** is executed. Then, the test expression is checked. If it is true, code/s inside body of **do** are executed again and the process continues until test expression becomes false (zero).

Example:

```

int a;
System.out.println( "do while loop result ");
a=5;
do
{   System.out.print( a+ ", ");    // then it will print value of a, then update value by 5
    a +=5;           // increment or update by 5 in variable a
} while ( a<=50); // if a is less than and equals to 50

```



```

Output - test1 (run) X
run:
do while loop result
5, 10, 15, 20, 25, 30, 35, 40, 45, 50, BUILD SUCCESSFUL (total time: 0 seconds)

```

➤ Nested Loop

Like nested if statement, if loop uses within a loop then this terminology is called “Nested loop”

Syntax:

```

for (initialization; condition; increment/decrement) // Outer Loop
{
    // Outer loop start here
    for (initialization; condition; increment/decrement) // Inner Loop
    {
        // Inner loop start here
        Statement's; // execute this statement
    }
}

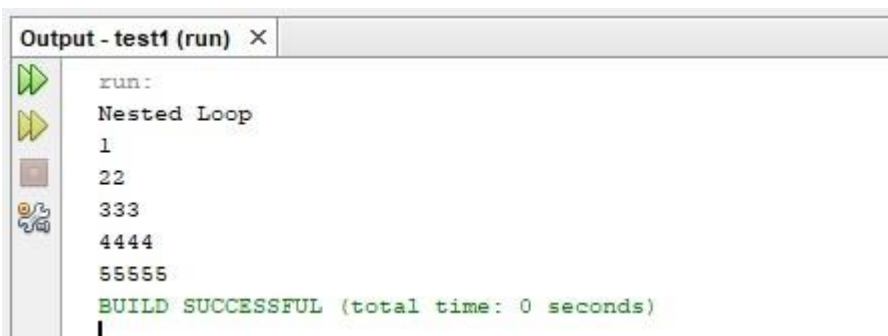
```

Example:

```

int a,b;
System.out.println( "Nested Loop ");
for ( a=1; a<=5; a++)
{
    for ( b=1; b<= a; b++)
    {
        System.out.print( a);
    }
    System.out.println( "");
}

```



```

Output - test1 (run) X
run:
Nested Loop
1
22
333
4444
55555
BUILD SUCCESSFUL (total time: 0 seconds)
|

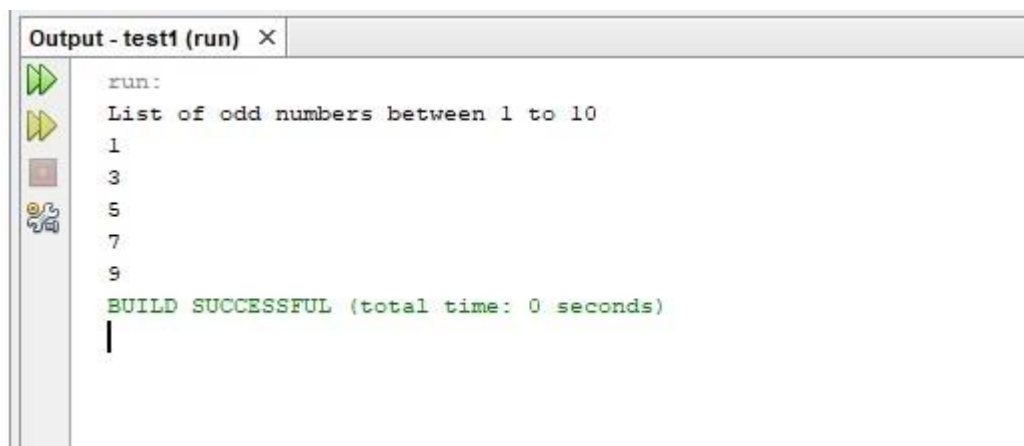
```

The **continue** keyword.

Sometimes a situation arises where we want to take the control to the beginning of the loop (for example **for**, **while**, **do while** etc.) skipping the rest statements inside the loop which have not yet been executed. The keywords **continue** allow us to **do** this. When the keywords **continue** executed inside a loop the control automatically passes to the beginning of loop. Continue is usually associated with the **if**.

In the following example, the lists of odd numbers between 1 to 10 have printed. In the **while** loop we test the remainder (here $x \% 2$) of every number, if the remainder is 0 then it becomes an even number and to avoid printing of even numbers **continue** statement is immediately used and the control passes to the beginning of the loop.

```
int x=1;
System.out.println( "List of odd numbers between 1 to 10 ");
while ( x<=10)
{
    if (( x % 2)==0)
    {
        x++;
        continue;
    }
    else
    {
        System.out.println( x+ " ");
        x++;
    }
}
```



```
Output - test1 (run) x
run:
List of odd numbers between 1 to 10
1
3
5
7
9
BUILD SUCCESSFUL (total time: 0 seconds)
```

Exercise

Theory Questions

1. Write down the syntax of Nested while and Nested do while loops.
2. Describe the purpose of a “counter” variable when executing a loop.
3. Which type of post-test loop and pre-test loop?
3. What is loop and how many types of loop in Java language?

Practical Questions

1. Write a program to print table of a number entered by user using while loop and do while loop.
2. Write a program to print following output using nested for loop

```

1
12
123
1234
12345

```

3. Write down the code to display series from 0 to 5 and their sum numbers, using loop of your choice.

Count	Total
0	0
1	1
2	3
3	6
4	10
5	15

4. Write down the code to display following output

```

1 2 3 4 5
2 4 6 8 10
3 6 9 12 15
4 8 12 16 20
5 10 15 20 25

```

4. Write a program series of prime number from 1 to 25.

Objective MCQ's:

- Each repetition of a looping statement is called a(n) _____.
 - Recurrence
 - Iteration
 - Duplication
 - Re-execution
- Which of the following is the correct syntax for a for loop statement?
 - `for(i=0; i <=10; ++)`
`System.out.print("display from a for statement ");`
 - `for(i=0, i <=10, ++)`
`System.out.print("display from a for statement ");`
 - `for(i=0; i <10);`
`System.out.print("display from a for statement ");`
 - `for`
{
`System.out.print("display from a for statement ");`
`}while (i=0; i<10; i++);`
- Counter variable _____.
 - Can only be increment.
 - Can only be decrement
 - Can be incremented or decremented
 - Do not change
- Which of the following is the correct syntax for while statement?
 - `While (i <10 , i++){ System.out.print(" i ");`
}
 - `While (i <10){`
`System.out.print(" i ");`
`i++;`
}
 - `While (i <10 , i++) System.out.print(" i "); i++;`
 - `While (i <10 , i++){ System.out.print(" i ");`
}
- These keywords are used in the loops, if depend on requirement.
 - continue
 - break;
 - a and b, both
 - case