## While loop:

if we don't know the no of iterations in advance then best loop is while loop:
The argument to the while statement should be Boolean type. If we are using any other type we will get compile time error.

**Example 1:**

```
public class ExampleWhile{
public static void main(String args[]){
while(1)
{
System.out.println("hello");
}}}
```

OUTPUT:
Compile time error.

```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:3: incompatible types
found : int
required: boolean
while(1)
```

Curly braces are optional and without curly braces we can take only one statement which should not be declarative statement.

**Example 2:**

```
public class ExampleWhile{
public static void main(String args[]){
while(true)
System.out.println("hello");
}}
```

OUTPUT:
Hello (infinite times).

Example 3:

```
public class ExampleWhile{
public static void main(String args[]){
while(true);
}}
```

OUTPUT:
No output.

**Example 4:**

```
public class ExampleWhile{
public static void main(String args[]){
while(true)
int x=10;
}}
```

OUTPUT:
Compile time error.

```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:4: '.class' expected
int x=10;
ExampleWhile.java:4: not a statement
int x=10;
```

**Example 5:**

```
public class ExampleWhile{
public static void main(String args[]){
while(true)
{
int x=10;
}}}
```

OUTPUT:
No output.

## Unreachable statement in while:

**Example 6:**

```
public class ExampleWhile{
public static void main(String args[]){
while(true)
{
System.out.println("hello");
}
System.out.println("hi");
}}
```

OUTPUT:

Compile time error.

```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:7: unreachable statement
System.out.println("hi");
```

**Example 7:**

```
public class ExampleWhile{
public static void main(String args[]){
while(false)
{
System.out.println("hello");
}
System.out.println("hi");
}}
```

OUTPUT:

```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:4: unreachable statement
{public class ExampleWhile{
public static void main(String args[]){
int a=10,b=20;
while(a<b)
{
System.out.println("hello");
}
System.out.println("hi");
}}
```

OUTPUT:

Hello (infinite times).

**Example 9:**

```
public class ExampleWhile{
public static void main(String args[]){
final int a=10,b=20;
while(a<b)
{
System.out.println("hello");
}
System.out.println("hi");
}}
```

OUTPUT:

Compile time error.

```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:8: unreachable statement
System.out.println("hi");
```

**Example 10:**

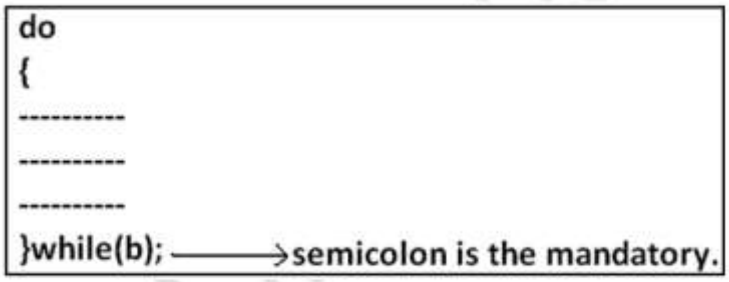
```
public class ExampleWhile{
public static void main(String args[]){
final int a=10;
while(a<20)
{
```

```
System.out.println("hello");
}
System.out.println("hi");
}}
```
OUTPUT:
```
D:\Java>javac ExampleWhile.java
ExampleWhile.java:8: unreachable statement
System.out.println("hi");
```
Note:

 Every final variable will be replaced with the corresponding value by compiler.
 If any operation involves only constants then compiler is responsible to perform that operation.
 If any operation involves at least one variable compiler won't perform that operation. At runtime jvm is responsible to perform that operation.

### Example 11:
```
public class ExampleWhile{
public static void main(String args[]){
int a=10;
while(a<20)
{
System.out.println("hello");
}
System.out.println("hi");
}}
```
OUTPUT:
```
Hello (infinite times).
```
## Do-while:
**If we want to execute loop body at least once then we should go for do-while.**

**Syntax:**


```
do
{
----------
----------
----------
}while(b); ———>semicolon is the mandatory.
```
**Curly braces are optional. Without curly braces we can take only one statement between do and while and it should not be declarative statement**

**Example 1:**
```
public class ExampleDoWhile{
public static void main(String args[]){
do
System.out.println("hello");
while(true);
}}
```
Output:
```
Hello (infinite times).
```
**Example 2:**
```
public class ExampleDoWhile{
public static void main(String args[]){
do;
while(true);
}}
```
Output:
```
Compile successful.
```
**Example 3:**
```
public class ExampleDoWhile{
public static void main(String args[]){
```

```
do
int x=10;
while(true);
}}
```
Output:
```
D:\Java>javac ExampleDoWhile.java
ExampleDoWhile.java:4: '.class' expected
int x=10;
ExampleDoWhile.java:4: not a statement
int x=10;
ExampleDoWhile.java:4: ')' expected
int x=10;
```

**Example 4:**
```
public class ExampleDoWhile{
public static void main(String args[]){
do
{
int x=10;
}while(true);
}}
```
Output:
Compile successful.
**Example 5:**
```
public class ExampleDoWhile{
public static void main(String args[]){
do while(true)
System.out.println("hello");
while(true);
}}
```
Output:
Hello (infinite times).
Rearrange the above Example:
```
public class ExampleDoWhile{
public static void main(String args[]){
do
 while(true)
 System.out.println("hello");
while(true);
}}
```
Output:
Hello (infinite times).
Example 6:
```
public class ExampleDoWhile{
public static void main(String args[]){
do
while(true);
}}
```
Output:
Compile time error.
```
D:\Java>javac ExampleDoWhile.java
ExampleDoWhile.java:4: while expected
while(true);
ExampleDoWhile.java:5: illegal start of expression
}
```

## Unreachable statement in do while:
**Example 7:**
```
public class ExampleDoWhile{
public static void main(String args[]){
```

```
do
{
System.out.println("hello");
}
while(true);
System.out.println("hi");
}}
```

Output:

```
Compile time error.
D:\Java>javac ExampleDoWhile.java
ExampleDoWhile.java:8: unreachable statement
System.out.println("hi");
```

**Example 8:**

```
public class ExampleDoWhile{
public static void main(String args[]){do
{
System.out.println("hello");
}
while(false);
System.out.println("hi");
}}
```

Output:

```
Hello
Hi
```

**Example 9:**

```
public class ExampleDoWhile{
public static void main(String args[]){
int a=10,b=20;
do
{
System.out.println("hello");
}
while(a<b);
System.out.println("hi");
}}
```

Output:

```
Hello (infinite times).
```

**Example 10:**

```
public class ExampleDoWhile{
public static void main(String args[]){
int a=10,b=20;
do
{
System.out.println("hello");
}
while(a>b);
System.out.println("hi");
}}
```

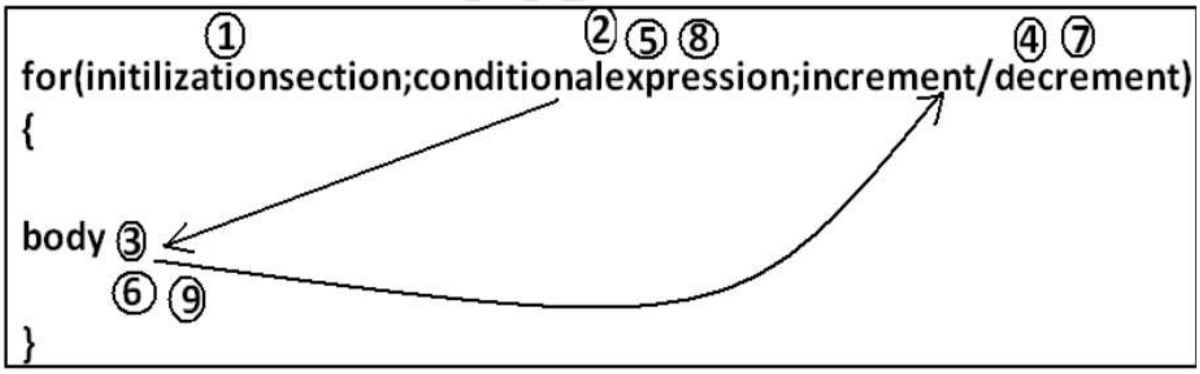
Output:

```
Hello
Hi
```

**Example 11:**

```
public class ExampleDoWhile{
public static void main(String args[]){
final int a=10,b=20;
do
{
System.out.println("hello");
}
while(a<b);
System.out.println("hi");
}}
```
Output:
Compile time error.
```
D:\Java>javac ExampleDoWhile.java
ExampleDoWhile.java:9: unreachable statement
System.out.println("hi");
```
**Example 12:**
```
public class ExampleDoWhile{
public static void main(String args[]){
final int a=10,b=20;
do
{
System.out.println("hello");
}
while(a>b);
System.out.println("hi");
}}
```
Output:
```
D:\Java>javac ExampleDoWhile.java
D:\Java>java ExampleDoWhile
Hello
Hi
```

## For Loop:
This is the most commonly used loop and best suitable if we know the no of iterations in advance.
Syntax:



for(initilizationsection;conditionalexpression;increment/decrement)
{
body
}
①
② ⑤ ⑧
④ ⑦
③
⑥ ⑨

**Curly braces are optional and without curly braces we can take only one statement which should not be declarative statement.**

### Initilizationsection:
This section will be executed only once.
Here usually we can declare loop variables and we will perform initialization.
We can declare multiple variables but should be of the same type and we can't declare different type of variables.

**Example:**
Int i=0,j=0; valid
Int i=0,Boolean b=true; invalid
Int i=0,int j=0; invalid
In initialization section we can take any valid java statement including "s.o.p" also.
Example 1:
```
public class ExampleFor{
public static void main(String args[]){
int i=0;
for(System.out.println("hello u r sleeping");i<3;i++){
System.out.println("no boss, u only sleeping");
}}}
```
Output:
```
D:\Java>javac ExampleFor.java
D:\Java>java ExampleFor
Hello u r sleeping
No boss, u only sleeping
No boss, u only sleeping
No boss, u only sleeping
```

## Conditional check:

We can take any java expression but should be of the type Boolean.
Conditional expression is optional and if we are not taking any expression compiler will place true.
Increment and decrement section:
Here we can take any java statement including s.o.p also.
Example:
```
public class ExampleFor{
public static void main(String args[]){
int i=0;
for(System.out.println("hello");i<3;System.out.println("hi")){
i++;
}}}
```
Output:
```
D:\Java>javac ExampleFor.java
D:\Java>java ExampleFor
Hello
Hi
Hi
Hi
```

**All 3 parts of for loop are independent of each other and all optional.**
**Example:**
```
public class ExampleFor{
public static void main(String args[]){
for(;;){
System.out.println("hello");
}}}
```
Output:
Hello (infinite times).
**Curly braces are optional and without curly braces we can take exactly one statement and it should not be declarative statement.**

## Unreachable statement in for loop:

**Example 1:**
```
public class ExampleFor{
public static void main(String args[]){
for(int i=0;true;i++){
System.out.println("hello");
}
System.out.println("hi");
}}
```
Output:
Compile time error.
```
D:\Java>javac ExampleFor.java
ExampleFor.java:6: unreachable statement
System.out.println("hi");
```
**Example 2:**
```
public class ExampleFor{
public static void main(String args[]){
```

```
for(int i=0;false;i++){
System.out.println("hello");
}
System.out.println("hi");
}}
```

Output:

Compile time error.

```
D:\Java>javac ExampleFor.java
ExampleFor.java:3: unreachable statement
for(int i=0;false;i++){
```

**Example 3:**

```
public class ExampleFor{
public static void main(String args[]){
for(int i=0;;i++){
System.out.println("hello");
}
System.out.println("hi");
}}
```

Output:

Compile time error.

```
D:\Java>javac ExampleFor.java
ExampleFor.java:6: unreachable statement
System.out.println("hi");
```

**Example 4:**

```
public class ExampleFor{
public static void main(String args[]){
int a=10,b=20;
for(int i=0;a<b;i++){
System.out.println("hello");
}
System.out.println("hi");
}}
```

Output:

Hello (infinite times).

**Example 5:**

```
public class ExampleFor{
public static void main(String args[]){
final int a=10,b=20;
for(int i=0;a<b;i++){
System.out.println("hello");
}
System.out.println("hi");
}}
```

Output:

```
D:\Java>javac ExampleFor.java
ExampleFor.java:7: unreachable statement
System.out.println("hi");
```

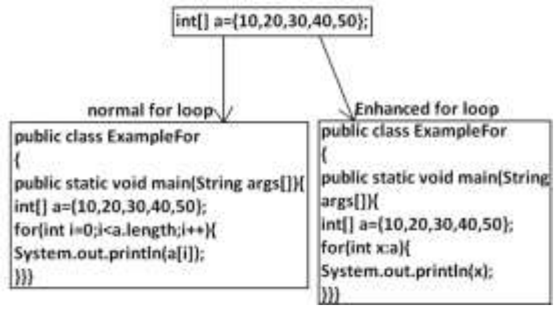
## For each:(Enhanced for loop)

- **For each Introduced in 1.5version.**
- **Best suitable to retrieve the elements of arrays and collections.**

**Example 1:**

**Write code to print the elements of single dimensional array by normal for loop and enhanced for loop.**

**Example:**



```
int[] a={10,20,30,40,50};
```

```
public class ExampleFor
{
public static void main(String args[]){
int[] a={10,20,30,40,50};
for(int i=0;i<a.length;i++){
System.out.println(a[i]);
}}}
```

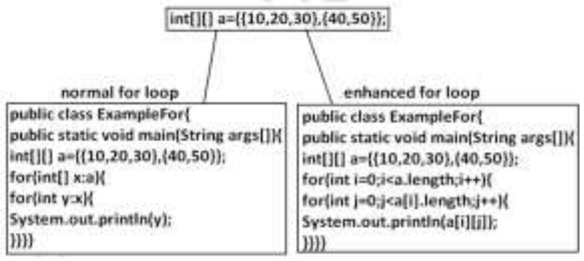
```
public class ExampleFor
{
public static void main(String
args[]){
int[] a={10,20,30,40,50};
for(int x:a){
System.out.println(x);
}}}
```

Output:

```
D:\Java>javac ExampleFor.java
D:\Java>java ExampleFor
10
20
30
40
50
```

**Example 2:**

**Write code to print the elements of 2 dimensional arrays by using normal for loop and enhanced for loop.**



```
int[][] a={{10,20,30},{40,50}};
```

```
public class ExampleFor{
public static void main(String args[]){
int[][] a={{10,20,30},{40,50}};
for(int[] x:a){
for(int y:x){
System.out.println(y);
}}}}
```

```
public class ExampleFor{
public static void main(String args[]){
int[][] a={{10,20,30},{40,50}};
for(int i=0;i<a.length;i++){
for(int j=0;j<a[i].length;j++){
System.out.println(a[i][j]);
}}}}
```

**Example 3:**

**Write equivalent code by For Each loop for the following for loop.**

```
public class ExampleFor{
public static void main(String args[]){
for(int i=0;i<10;i++)
{
System.out.println("hello");
}}}
```

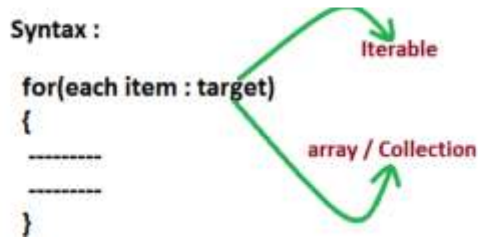
**Output:**

```
D:\Java>javac ExampleFor1.java
D:\Java>java ExampleFor1
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
Hello
```

** We can't write equivalent for each loop.**

** For each loop is the more convenient loop to retrieve the elements of arrays and collections, but its main limitation is it is not a general purpose loop.**

** By using normal for loop we can print elements either from left to right or from right to left. But using for-each loop we can always print array elements only from left to right.**



Syntax :

```
for(each item : target)
{
 ---------
 ---------
}
```

 The target element in for-each loop should be Iterable object.
 An object is set to be iterable iff corresponding class implements java.lang.Iterable interface.
 Iterable interface introduced in 1.5 version and it's contains only one method iterator().
Syntax : public Iterator iterator();
Every array class and Collection interface already implements Iterable interface.
Difference between Iterable and Iterator:
Iterable Iterator
It is related to forEach loop It is related to Collection
The target element in forEach loop should
be Iterable
We can use Iterator to get objects one by
one from the collection
Iterator present in java.lang package Iterator present in java.util package
contains only one method iterator() contains 3 methods hasNext(), next(),
remove()
Introduced in 1.5 version Introduced in 1.2 version

Break statement:
We can use break statement in the following cases.
 Inside switch to stop fall-through.
 Inside loops to break the loop based on some condition.
 Inside label blocks to break block execution based on some condition.

Inside switch :
We can use break statement inside switch to stop fall-through
Example 1:

```
class Test{
public static void main(String args[]){
int x=0;
switch(x)
{
case 0:
 System.out.println("hello");
 break ;
case 1:
 System.out.println("hi");
}
```

Output:

```
D:\Java>javac Test.java
D:\Java>java Test
Hello
```

Inside loops :
We can use break statement inside loops to break the loop based on some condition.
Example 2:

```
class Test{
```

```
public static void main(String args[]){
for(int i=0; i<10; i++) {
if(i==5)
 break;
 System.out.println(i);
}
}}
```

Output:

```
D:\Java>javac Test.java
D:\Java>java Test
0
1
2
3
4
```

**Inside Labeled block :**

We can use break statement inside label blocks to break block execution based on some condition.

Example:

```
class Test{
public static void main(String args[]){
int x=10;
l1 : {
 System.out.println("begin");
 if(x==10)
 break l1;
 System.out.println("end");
}
System.out.println("hello");
}
}
```

Output:

```
D:\Java>javac Test.java
D:\Java>java Test
begin
hello
```

These are the only places where we can use break statement. If we are using anywhere else we will get compile time error.

Example:

```
class Test{
public static void main(String args[]){
int x=10;
if(x==10)
break;
System.out.println("hello");
}
}
```

Output:

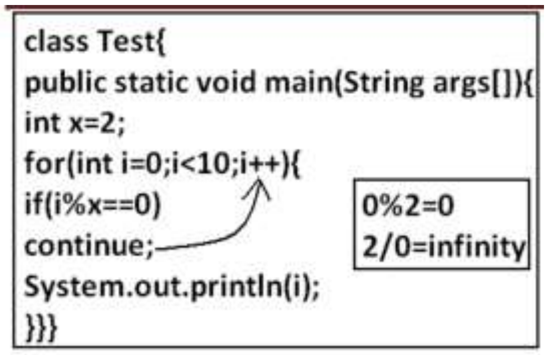
Compile time error.

```
D:\Java>javac Test.java
Test.java:5: break outside switch or loop
break;
```

**Continue statement:**

We can use continue statement to skip current iteration and continue for the next iteration.

Example:



```
class Test{
public static void main(String args[]){
int x=2;
for(int i=0;i<10;i++){
if(i%x==0)
continue;
System.out.println(i);
}}}
```

```
0%2=0
2/0=infinity
```

Output:

```
D:\Java>javac Test.java
D:\Java>java Test
1
3
5
7
9
```

We can use continue only inside loops if we are using anywhere else we will get compile time error saying "continue outside of loop".

Example:

```
class Test
{
public static void main(String args[]){
int x=10;
if(x==10);
continue;
System.out.println("hello");
}
}
```

Output:

Compile time error.

```
D:\Enum>javac Test.java
Test.java:6: continue outside of loop
continue;
```

Labeled break and continue statements:

In the nested loops to break (or) continue a particular loop we should go for labeled break and continue statements.

Example:

```
class Test
{
public static void main(String args[]){
l1:
for(int i=0;i<3;i++)
{
 for(int j=0;j<3;j++)
 {
 if(i==j)
 break;
 System.out.println(i+"........."+j);
 }

}
}
}
```

Break:

```
1.........0
2.........0
2.........1
```

Break l1:

No output.

Continue:

```
0.........1
0.........2
1.........0
1.........2
2.........0
2.........1
```

Continue l1:

```
1.........0
2.........0
2.........1
```

Example 1:

```
class Test
{
public static void main(String args[]){
while(true)
{
System.out.println("hello");
}
System.out.println("hi");
}
}
```

Output:

Compile time error.

```
D:\Enum>javac Test.java
Test.java:8: unreachable statement
System.out.println("hi");
```

Example 2:

```
class Test
{
public static void main(String args[]){
if(true)
{
System.out.println("hello");
}
else
{
System.out.println("hi");
}}}
```

Output:

```
Hello
```