Iteration Statements:

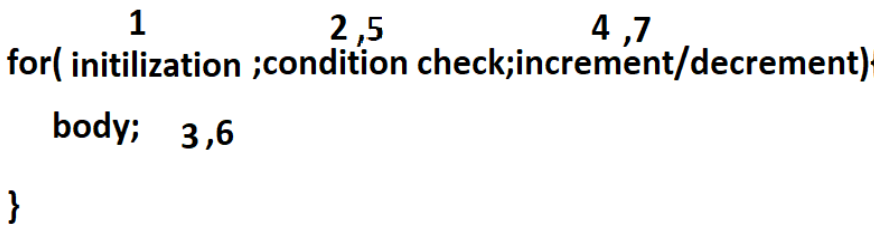
This type of statement is used to execute a statement or set of statements multiple times.

Iteration statement is further classified as:

1) for loop

- This is one of the most widely used loop.
- This loop is the best choice if the number of iterations is already known.

Syntax:



```
               1                 2 ,5                     4 ,7
for( initilization ;condition check;increment/decrement)
    body;   3 ,6

}
```

For loop have four sections:

a) Initialization Section:

- In this section, we declare the loop variable.
- This section is the starting point of for loop and is executed only once.
- We can initialize multiple variables at the same time but should be of same type.
- We can write any valid java statement here, even System.out.println()

For ex.

```
1.    for(int j=10, j=20;…;….){
      //valid
      }
2.    for(int j=10, int j=20;…;….){
      //invalid
```

```
        }
3.      for(int j=10, boolean b=false;…;….){
        //invalid
        }
4.      for(System.out.println("Hi");…;….){
        //valid
        }
```

b) Conditional Check:

- This section of for loop is executed multiple times depending upon the condition used, variable value, increment/decrement, etc.
- As we are checking a condition here so any java statement which ultimately evaluates to boolean result (true/false) can be used here.
- This section is optional and if are not using any statement here then compiler will by default keep it as true and it will be an infinite for loop.
- Till the time conditions check is true, it will execute the for loop body. Once the condition check becomes false, then loop body won't be executed and control will completely come out of for loop.

c) Increment and decrement section:

- This section is used to increase/decrease the value of loop variable.
- Although we can use any valid java statement including s.o.p also.

Note:
All the section of for loop are optional.
Curly braces are optional and without curly braces we can take only one statement which should not be declaration statement.

For ex:

1.
```
for( ; ; ){
        System.out.println("Hi");
}
```

Output: Infinite time Hi.

2.
```
int x=0;
for(System.out.println("In initialization");  x<3 ;
System.out.println("In increment/decrement")){
    System.out.println("In for loop body");
    x++;
}
```

Output:

```
In initialization
In for loop body
In increment/decrement
In for loop body
In increment/decrement
In for loop body
In increment/decrement
```

Unreachable statement in for loop:

There can be a compile time error of unreachable statement while using for loop.
It happens when for loop will be executed infinitely and the next statement after for loop won't execute ever.

For ex.

```
for(int i=0;true;i++){
        System.out.println("In for loop body");
}
System.out.println("Outside for loop");
```

Output:

Here System.out.println("Outside for loop") will give compile time error as the for loop will run indefinitely and this statement will never get a chance to execute.

1. while loop

- While loop is the best choice when the number of iterations are not known in advance.

Syntax:

```
while(condition){
    loop body;
}
```

- The Condition argument should be of Boolean type. Otherwise we will get a compile time error.
- Curly braces are optional here also and without curly braces we can write only one java statement which should not be a declaration statement.

  For ex.

  1)

  ```
  while(rs.next()){
  //body
  }
  ```

Unreachable statement in while loop:

Like for loop, while loop can also have unreachable statement.

For ex:

```
while(true){
        System.out.println("Inside while loop");
}
System.out.println("Outside while loop");
```

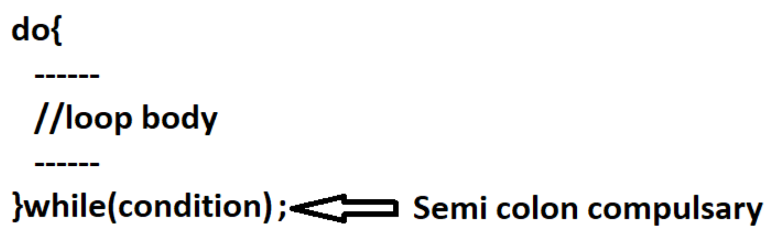
Output:

Here it will be compile time error as Outside while loop statement will never get a chance to execute as while loop will execute infinitely.

do-while loop:

If we want to execute a loop body atleast once, then we use do while loop.

Syntax:



```
do{
  ------
  //loop body
  ------
}while(condition);
```

- *Curly braces are optional.*
- *If we use without curly braces then we should take exactly one statement which should not be declarative statement.*

For ex.

1.
```
do{
        System.out.println ("Hello");
}while(true);
```

Output:

Hello (infinite times)

2.
```
do{
        System.out.println ("Hello");
}while(false);
```

Output:

Hello

Unreachable statement in while loop:

Like for and while loop, do-while loop can also have unreachable statement.

For ex:

```
do{
        System.out.println ("Hello");
}while(true);
System.out.println("Outside do-while loop");
```

Output:

Here it will be compile time error as Outside do-while loop statement will never get a chance to execute as do-while loop will execute infinitely.

Transfer / Jump Statements:

Break statement:

Break statement is used in following cases:

1) Inside loop when you want to break the loop execution under some condition
   - If you don't want to continue the loop to execute if some particular condition is met, then we use break within the loop.

     For ex.

```
for(int i=0; i<10;i++){
        if(i==4)
        break;
        System. out.println("Hello "+i);
}
        System. out.println("Outside for loop");
```

```
Output:
Hello 0
Hello 1
Hello 2
Hello 3
Outside for loop
```

2) Inside switch to stop the fall through within switch

- Within the switch statement if any case is matched from that case onwards all statements will be executed until end of the switch (or) break. This is call "fall-through" inside the switch
- Hence to avoid or stop fall-through we use break in switch.

For ex

1. Switch case without break:

```java
int x=2;
switch(x){
        case 1:
                System.out.println("Case one");
        case 2:
                System.out.println("Case two");
        case 3:
                System.out.println("Case three");
        case 4:
                System.out.println("Case four");
        default:
                System.out.println("Default Case");
}
```

```
Output:
        Case two
        Case three
        Case four
        Default Case
```

2. Switch case with break:

```
int x=2;
switch(x){
    case 1:
        System.out.println("Case one");
        break;
    case 2:
        System.out.println("Case two");
        break;
    case 3:
        System.out.println("Case three");
        break;
    case 4:
        System.out.println("Case four");
        break;
    default:
        System.out.println("Default Case");
}
```

Output:

```
Case two
```

Note:
If you use break statement at any other location apart from mentioned above, then it will be a compile time error.

Continue Statement:

If you want to skip just current iteration and go to next iteration of loop instead of skipping the entire loop execution then we use continue statement.

For ex:

1) Print all even number between 1 to 10.

```
        for(int i=1; i<=10;i++){
                if(i%2!=0)
                continue;
                System.out.println(i);
        }
```

Output:

```
2
4
6
8
10
```

Note:
If you use continue statement at any other location apart from mentioned above, then it will be a compile time error.