Control Statements

The statement of a java program will be executed in sequential order. If we don't want to execute the statements in sequential order i.e. if we want to execute the statements in random order according to programmer choice, then we take the help of control statements.

The control statements will help the programmer to control the flow of execution in a java program. The control statements are classified as following

1. Conditional statements
   - 1.1 if else statement
   - 1.2 switch statement
2. Iterating statements
   - 2.1 for loop
   - 2.2 while loop
   - 2.3 do-while loop
   - 2.4 for each loop
3. Transfer statements
   - 3.1 break
   - 3.2 continue
   - 3.3 return

Conditional Statements:

If-else statement: if else statement can be used for executing a group of statements based on a condition.

Syntax :

```
if (condition)  {
   statements1;       if block
}
else  {
   statements2;      else block
}
```

If the condition is satisfied then it will be executing if block and if the condition is not satisfied then it will be executing else block.

Switch statement: The switch statement can be used for executing a group of statements based on a value.

Syntax:

```
switch (argument)  {
   case label1:  statements1;
                 break;
   case label2:  statements2;
                 break;
   :
   default:  default statements;
}
```

Iterating Statements:

For Loop: This loop can be used for executing the statements multiple times. A for loop has to be used when we know the exact number of iterations.

Syntax:

```
          1            2  5  8…       4  7  10…
for (initialization ; condition ; increment/decrement)  {
       statements;  3  6  9…
}
```

While loop: This loop can be used for executing the statements multiple times. A while loop has to be used when we do not know the exact number of iterations.

Syntax:

```
while(condition)  {
    statements;
}
```

Do-While loop: This loop can be used for executing the statements multiple times. A do-while loop has to be used when we do not know there exact number of iterations.

Syntax:

```
do  {
    statements;
} while(condition);
```

Difference between while loop and do-while loop:

1) In a while loop the statements will execute after evaluating the condition, whereas in a do-while loop the statements will execute before evaluating the condition.
2) In a while loop if the condition is false for the first time then the statements will not execute, whereas in a do-while loop if the condition is false for the first time then the statemens will execute for one time.
3) In a while loop the statements will execute for 0 or more times, whereas in a do-while loop the statements will execute for 1 or more times.

Nested Loop: If we specify a loop inside another loop then it is called as nested loop. Any loop can be specified inside any other loop any number of times.

Transfer Statements:

Break: break is a transfer statement which can be used either inside switch statement or inside a loop.

- The break statement, when used in switch will transfer the control from inside the switch to outside the switch, so that we skip the execution of removing cases.
- The break statement, when used in loop, will transfer the control from inside the loop to outside the loop, so that we will skip the execution of remaining iterations.

Note: When we are specifying the break statement in a loop we are recommended to specify the break statement along with condition.

Continue: continue is a transfer statement which has to be used only in loops. The continue statements will skip the current iteration and continues with the remaining iterations.

Note: When we are specifying a continue statement in a loop, then we are recommended to specify the continue statement along with a condition.