
JAVA

JAVA CONTROL STATEMENTS / CONTROL FLOW IN JAVA

- Java compiler executes the code from top to bottom. The statements in the code are executed according to the order in which they appear.
- Java provides statements that can be used to control the flow of Java code. Such statements are called control flow statements. It is one of the fundamental features of Java, which provides a smooth flow of program.
- Java provides three types of control flow statements.
  1. ***Decision Making statements***
  2. ***Loop statements***
  3. ***Jump statements***

DECISION-MAKING STATEMENTS

- Decision-making statements evaluate the Boolean expression and control the program flow depending upon the result of the condition provided.
  - if statements
  - switch statement

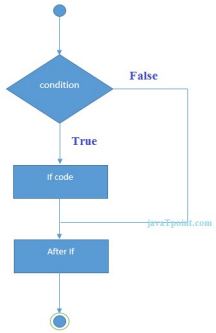
IF STATEMENT

- The "if" statement is used to evaluate a condition. The control of the program is diverted depending upon the specific condition.
- The condition of the If statement gives a Boolean value, either true or false.
  - Simple if statement
  - if-else statement
  - if-else-if ladder
  - Nested if-statement

IF STATEMENT

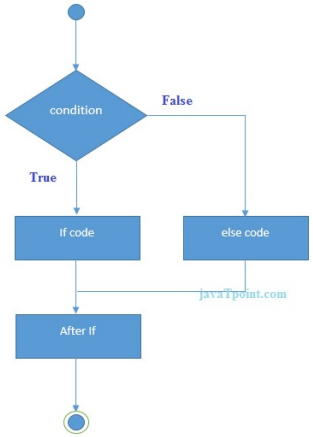
Simple if statement

```
if(condition) {
statement 1; //executes when condition is true
}
```



if-else statement

```
if(condition) {
statement 1; //executes when condition is true
}
else{
statement 2; //executes when condition is false
}
```



IF STATEMENT

if-else-if ladder

```
if(condition 1) {
statement 1; //executes when condition 1 is true
}
else if(condition 2) {
statement 2; //executes when condition 2 is true
}
else {
statement 2; //executes when all the conditions are false

}
```

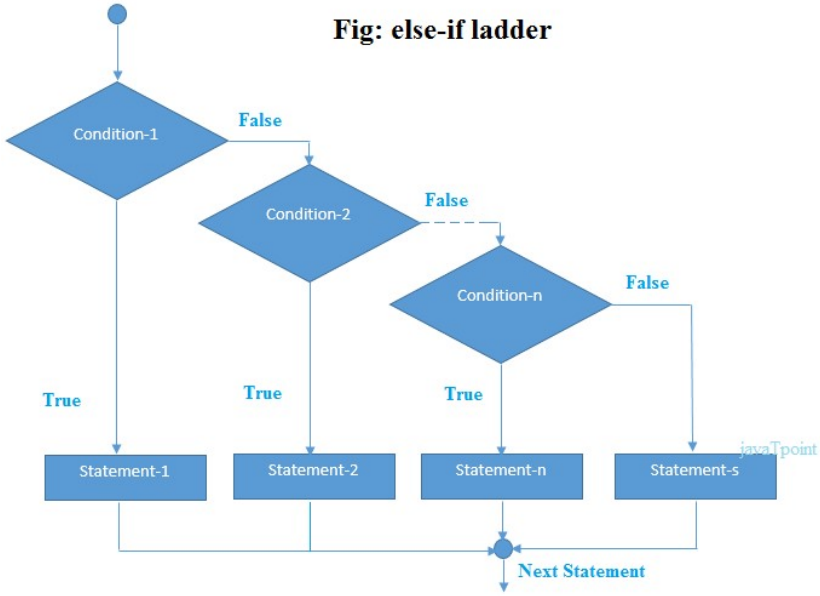
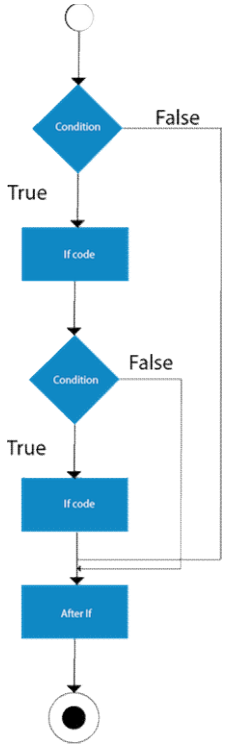


Fig: else-if ladder

IF STATEMENT

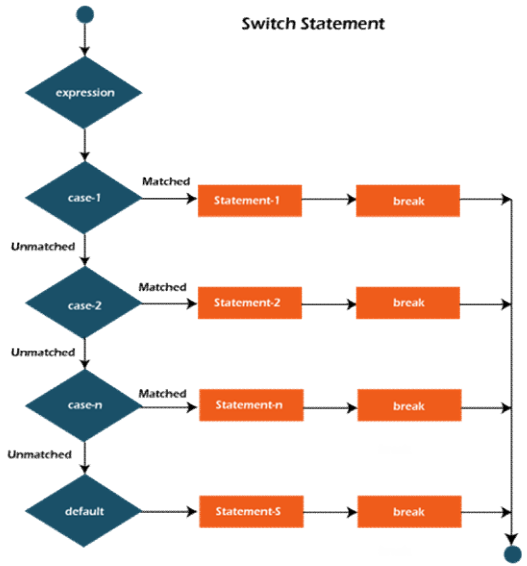
Nested if-statement

```
if(condition 1) {
statement 1; //executes when condition 1 is true
if(condition 2) {
statement 2; //executes when condition 2 is true
}
else{
statement 2; //executes when condition 2 is false
}
}
```



SWITCH STATEMENT

- In Java, Switch statements are similar to if-else-if statements.
- The switch statement contains multiple blocks of code called cases and a single case is executed based on the variable which is being switched.



SWITCH STATEMENT

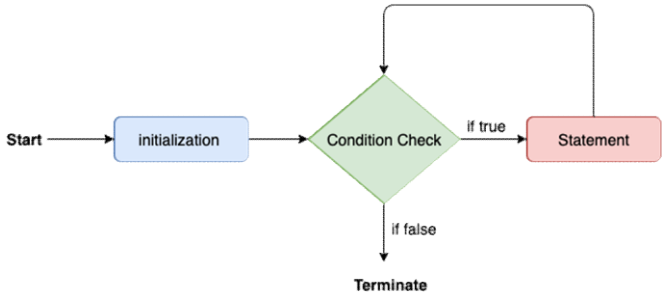
```
switch(expression){
case value1:
 //code to be executed;
 break;  //optional
case value2:
 //code to be executed;
 break;  //optional
.....

default:
  code to be executed if all cases are not matched;
}
```

LOOPS IN JAVA

- Loop statements are used to execute the set of instructions in a repeated order. The execution of the set of instructions depends upon a particular condition.
  - for loop
  - while loop
  - do-while loop

JAVA FOR LOOP

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

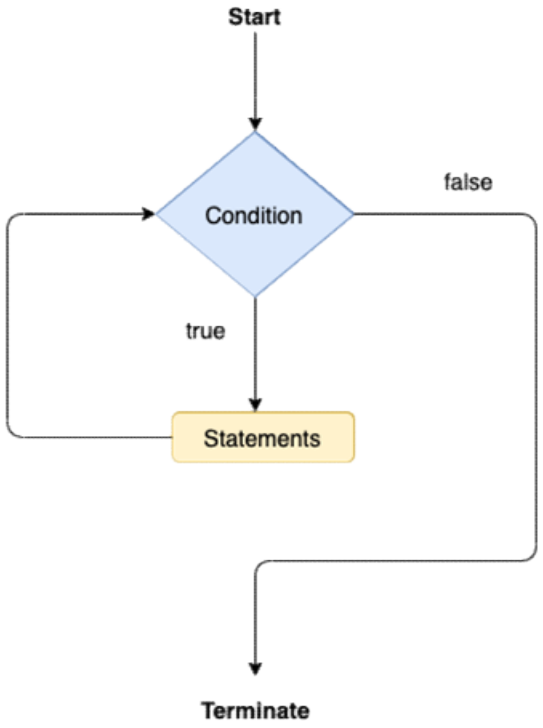


- **Initialization**: It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
- **Condition**: It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
- **Increment/Decrement**: It increments or decrements the variable value. It is an optional condition.
- **Statement**: The statement of the loop is executed each time until the second condition is false.

JAVA WHILE LOOP

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.

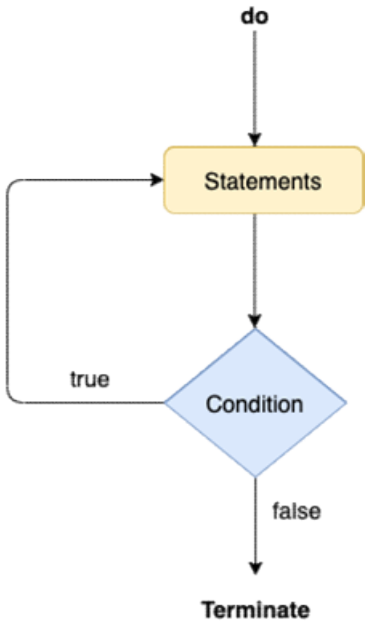
```
while (condition){
//code to be executed
}
```



JAVA DO-WHILE LOOP

- Java do-while loop is called an **exit control loop**.
- The Java *do-while loop* is executed at least once because condition is checked after loop body.

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



JUMP STATEMENTS

- Jump statements are used to transfer the control of the program to the specific statements. In other words, jump statements transfer the execution control to the other part of the program. There are two types of jump statements in Java, i.e., break and continue.

JAVA ***BREAK & CONTINUE*** STATEMENT

- ***break*** statement is used to break the current flow of the program and transfer the control to the next statement outside a loop or switch statement.
- The break statement cannot be used independently in the Java program, i.e., it can only be written inside the loop or switch statement.

- ***continue*** statement doesn't break the loop, whereas, it skips the specific part of the loop and jumps to the next iteration of the loop immediately.