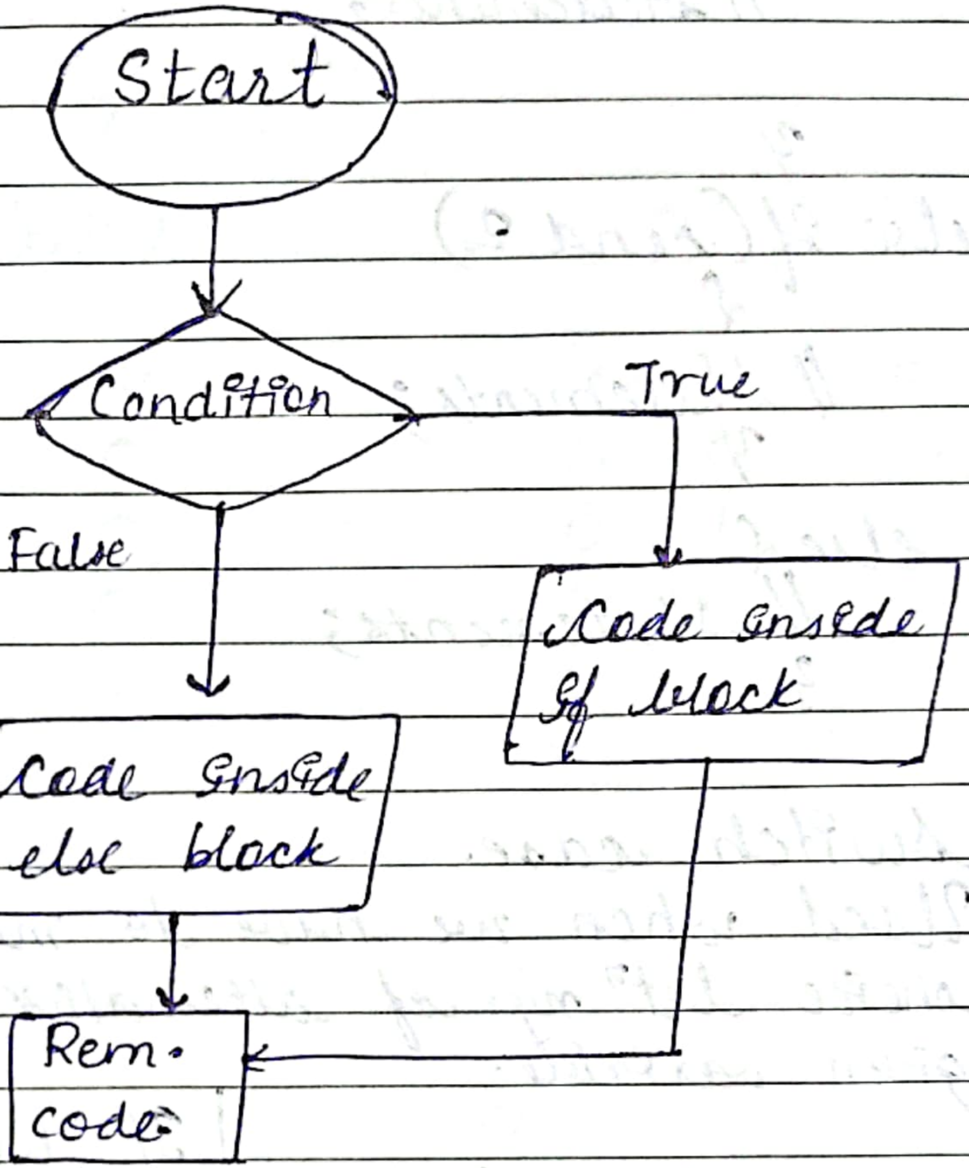
Conditionals



Decision making

```
if (condition to be checked)
{
    /* code */
}
else{
  Code is exec. if condition false
}
```

Else-If

```
If(cond 1) {

        // statements;

    }
else if(cond 2)
        {
    // Statements;
        }
  else {
    // Statements;
    }
```

Switch case.

Used when we have to make a choice betn no. of alternatives for a given variable.

```
Switch(var) {
        ~~Case 1~~
        Case C1:
            // code;
            break;

        Case C2:
            // Code;
            break;

        ~~D.~~ default:
            // code
    }
```

If break st. is not used switch case won't terminate.

Var can be a integer, character or string In Java
A switch can occur within another but in practice its rarely done.

Logical Operators

AND (&&) OR (||)

A	B	R
0	0	0
0	1	0
1	0	0
1	1	1

A	B	R
0	0	0
0	1	1
1	0	1
1	1	1

NOT(!)

~~A~~	~~B~~	~~R~~	A	R
~~0~~	~~1~~		0	1
			1	0

Enhanced Switch

```
String var="Saurabh";

switch(var){
case "Shubham" -> {
        Sout("....");
      }
case "Saurabh" -> {
          Sout("---.");
        }
```

default → Sout("....");

Practice set.

Q Check whether the student is passing or failing if mark req. in each sub. is 33 & ~~total marks=~~ 40 percentage of marks is 40. Ass. 3 subjects

```
public class Marks{

  p s v m(String [] args){

  Scanner sc = new Scanner (System.in);

  Sout("Subject 1 marks:);
  float s1= sc.nextFloat();

  Sout("Subject 2 marks:);
  float s2= sc.nextFloat();

  Sout("Subject 3 marks:);
  float s3= sc.nextFloat();

  float total=(s1+s2+s3)/3;

   if((s1>=33 || s2>=33 || s3>=33) &&
                                total>=40)
      {
       sout("You have passed");
      }
   else {
       sout("You have failed");
      }       } }
```

```
o/p

Subject 1 marks
 33

Subject 2 marks
 33

Subject 3 marks
 33

You have failed
```

Loops:

While Loop:

→ The while loop in Java is used when we need to execute a block of cod again & again based on the boolean condition.

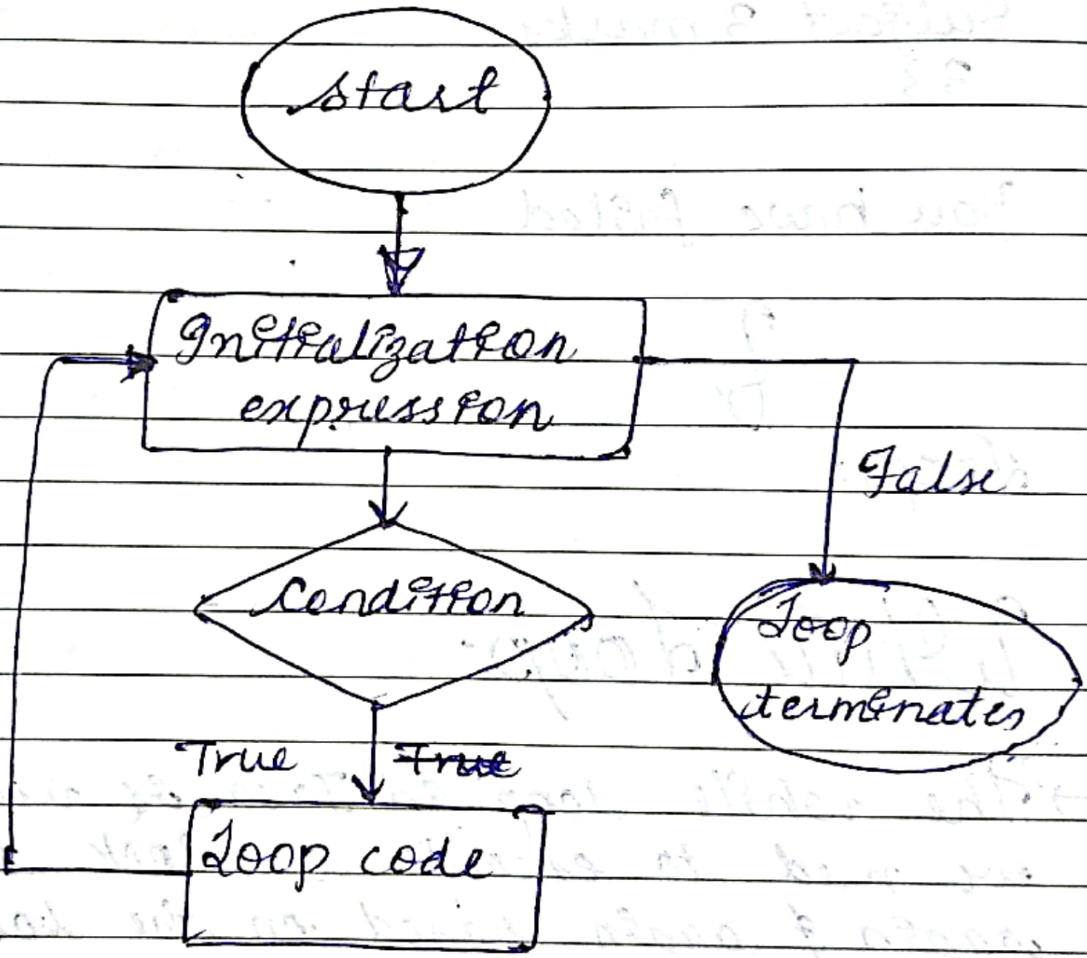
→ Use when exact ~~Itera~~ no. of Iterations are not known.

→ If condition never becomes false, it becomes a ∞ loop.

```
while(Boolean condition)
{
        // code

    }
```

Flowchart



do-while loop

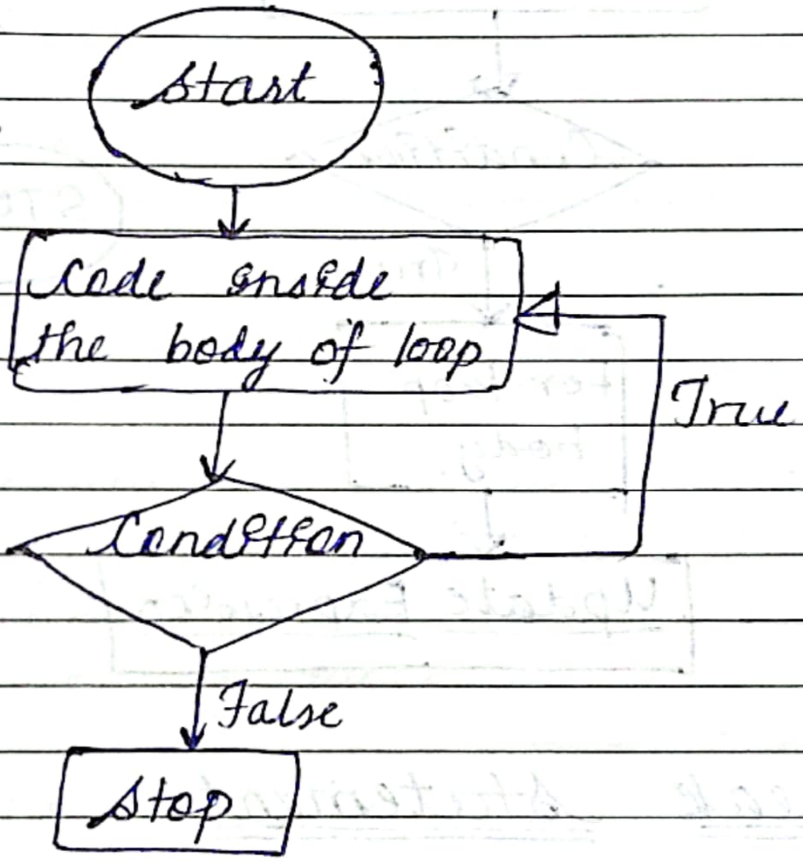
→ Do while loop is similar to a while loop except for the fact that it is guaranteed to execute atleast once.

→ After executing a part of program for once, the rest code gets executed on the base of boolean expression.

```
do {
    // code

  } while (condition);
```

Flowchart

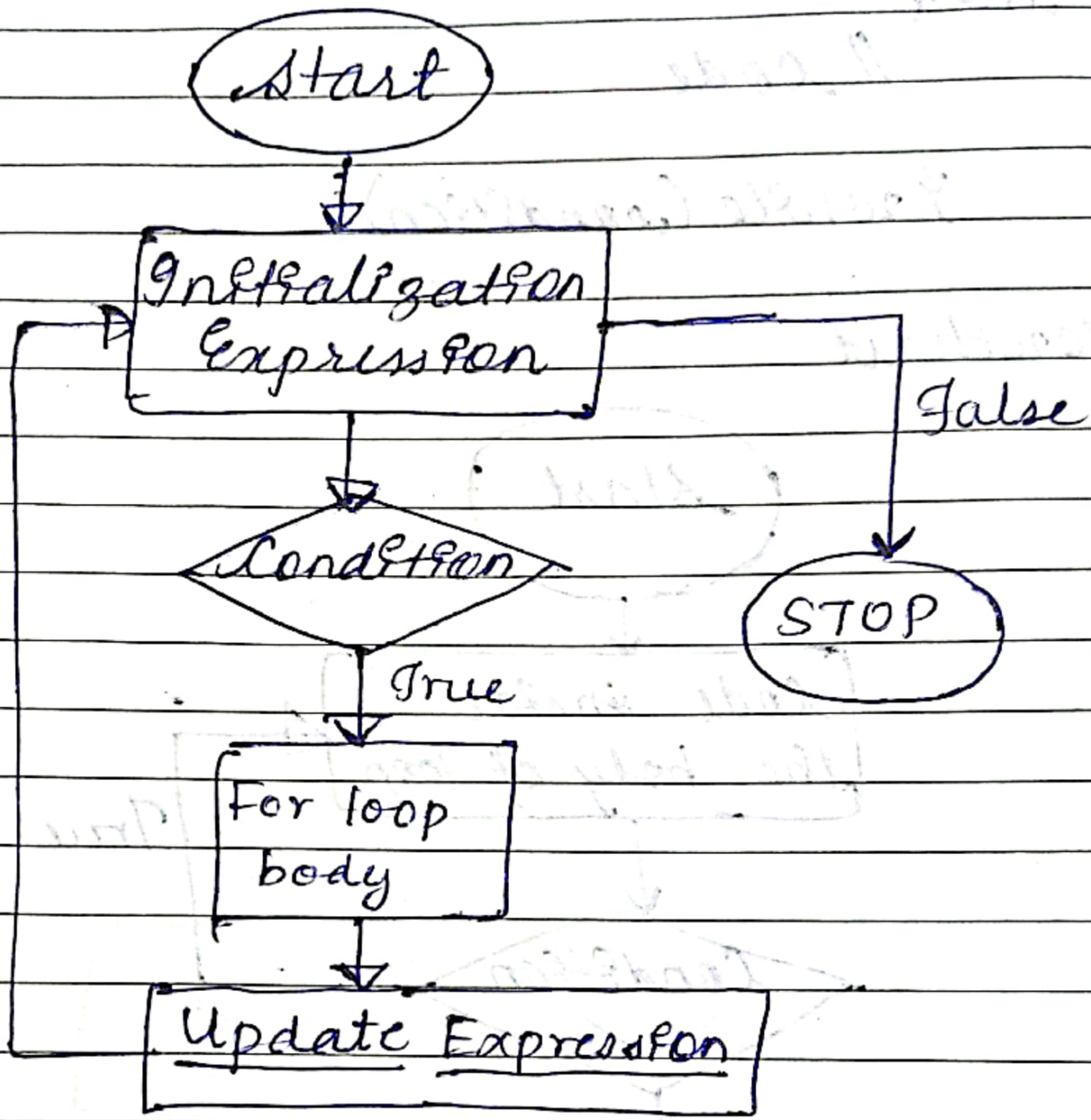


~~For~~ For Loop

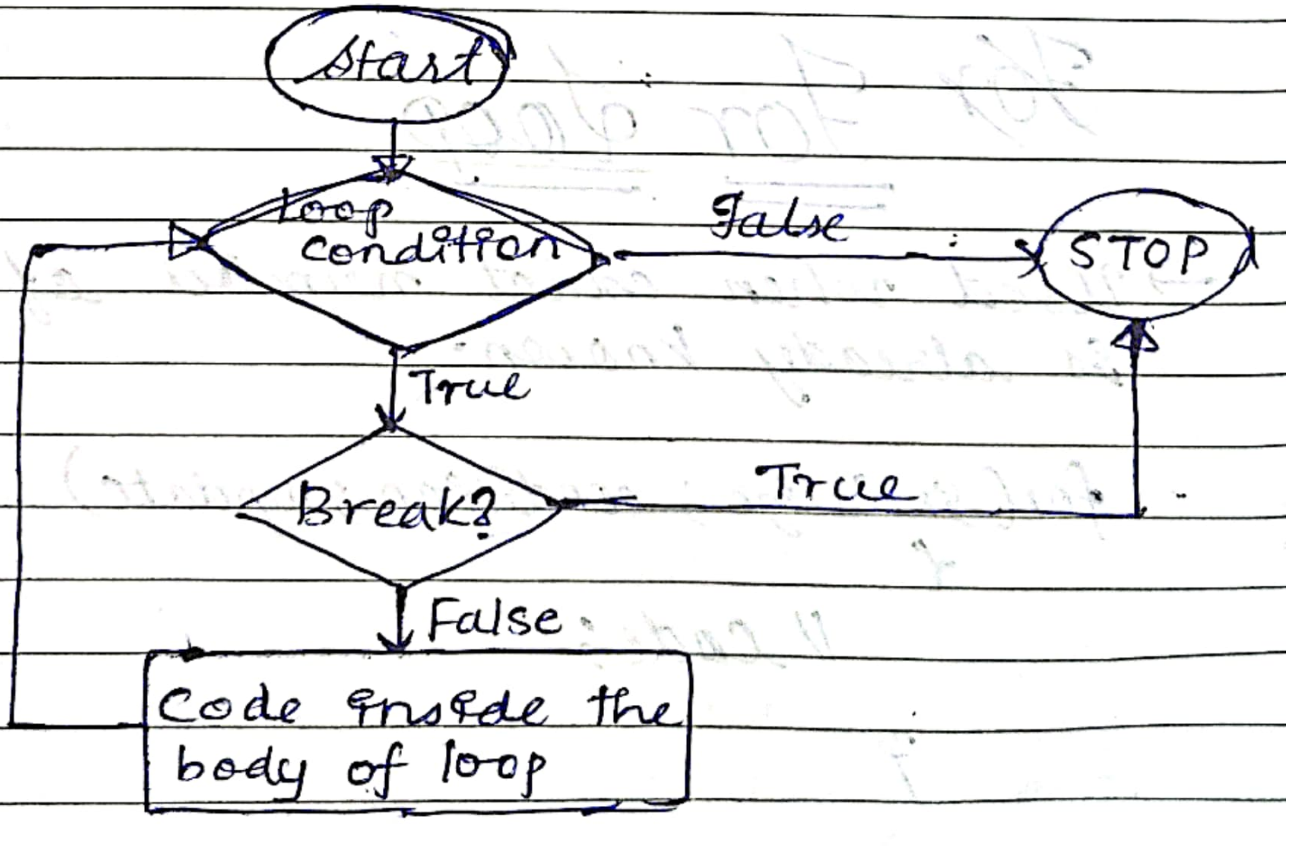
→ Used when exact number of iterations is already known.

```
for( initialize ; condition ; update)
  {
       // code;

  }
```



Break Statement



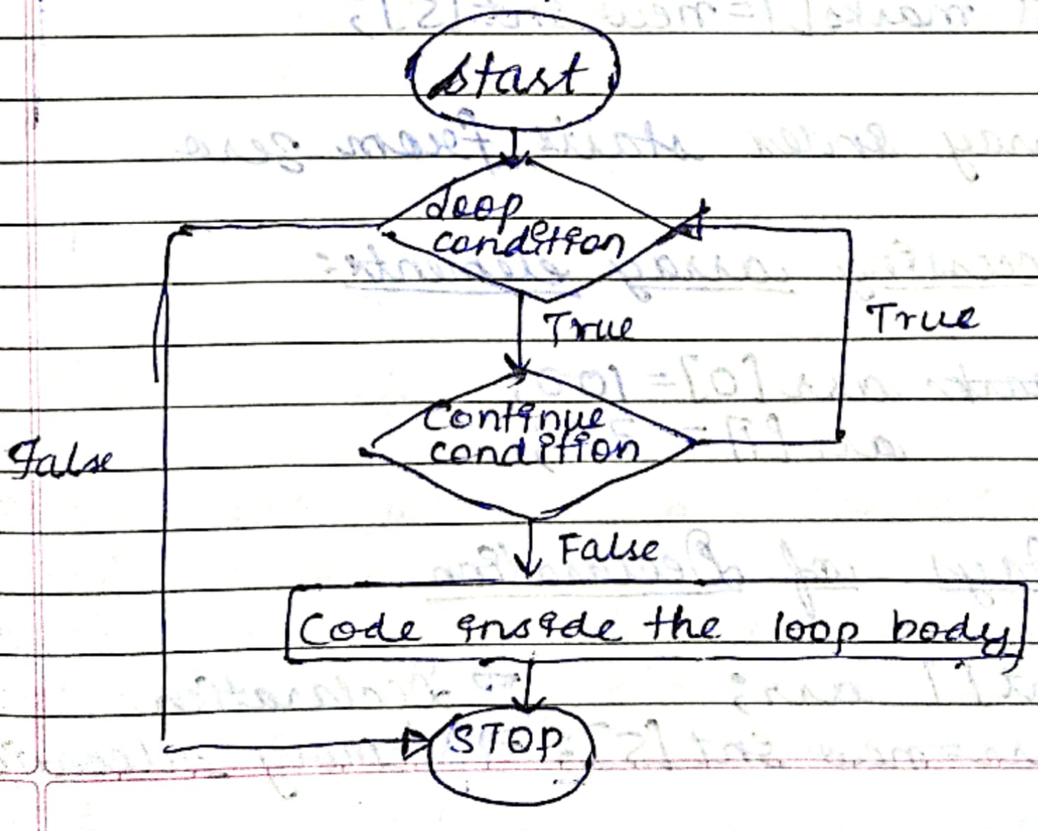
- Break statement is used to exit the loop irrespective of whether condition is true or false
- Whenever a 'break' statement is encountered control is sent outside the loop.

Syntax:

```
break;
```

Continue Statement

- The continue statement is used to immediately move to the next iteration of the loop
- The control is taken to the next iteration skipping everything below 'continue' inside the loop for that iteration.



Syntax:

```
continue;
```

In a nutshell

→ break statement completely exits the loop

→ continue statement skips the particular iteration of the loop.