- **Control Statements in C**

C if statement

Syntax :

if(expression)

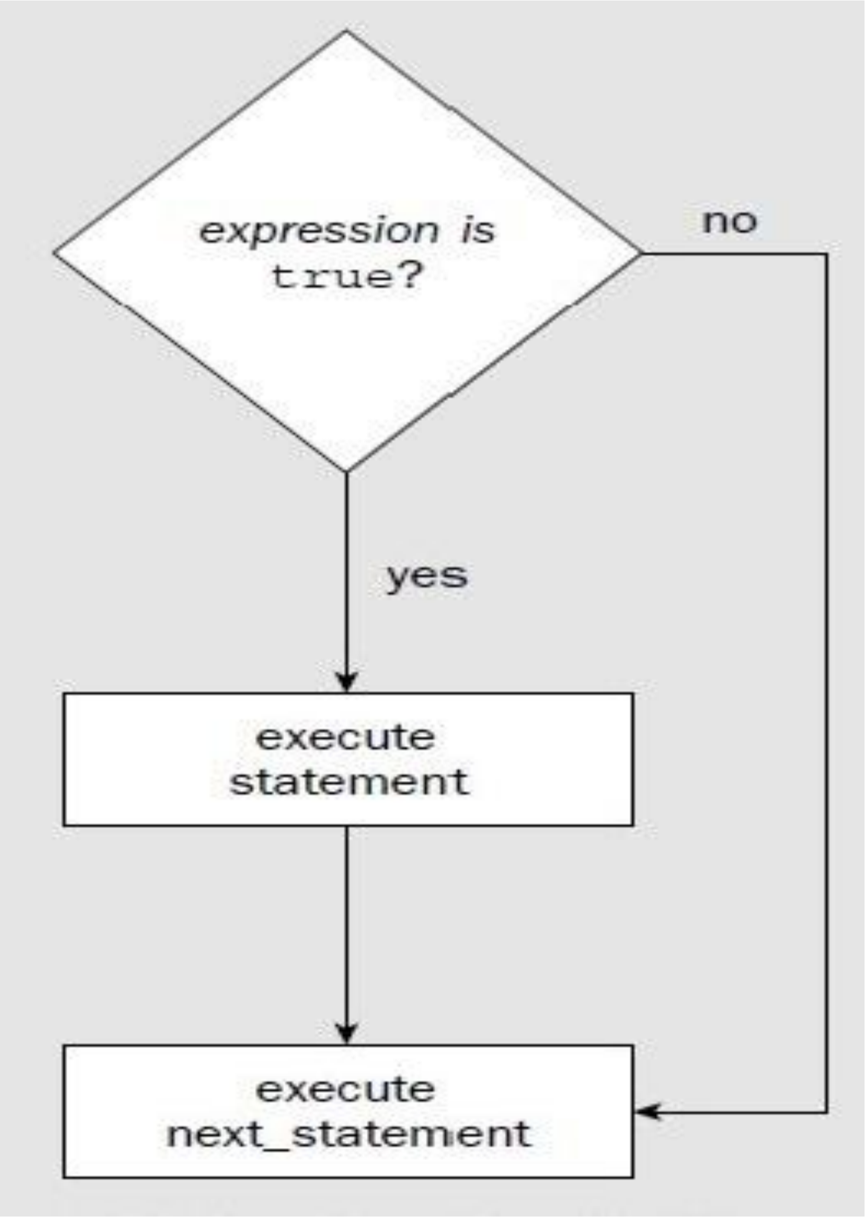
statement1;

Explanation :

- Expression is Boolean Expression
- It may have true or false value

Meaning of If Statement :

- It Checks whether the given Expression is Boolean or not !!
- If Expression is True Then it executes the statement otherwise jumps to next instruction



C if-else statement

- We can use if-else statement in c programming so that we can check any condition and depending on the outcome of the condition we can follow appropriate path. We have true path as well as false path.

Syntax :

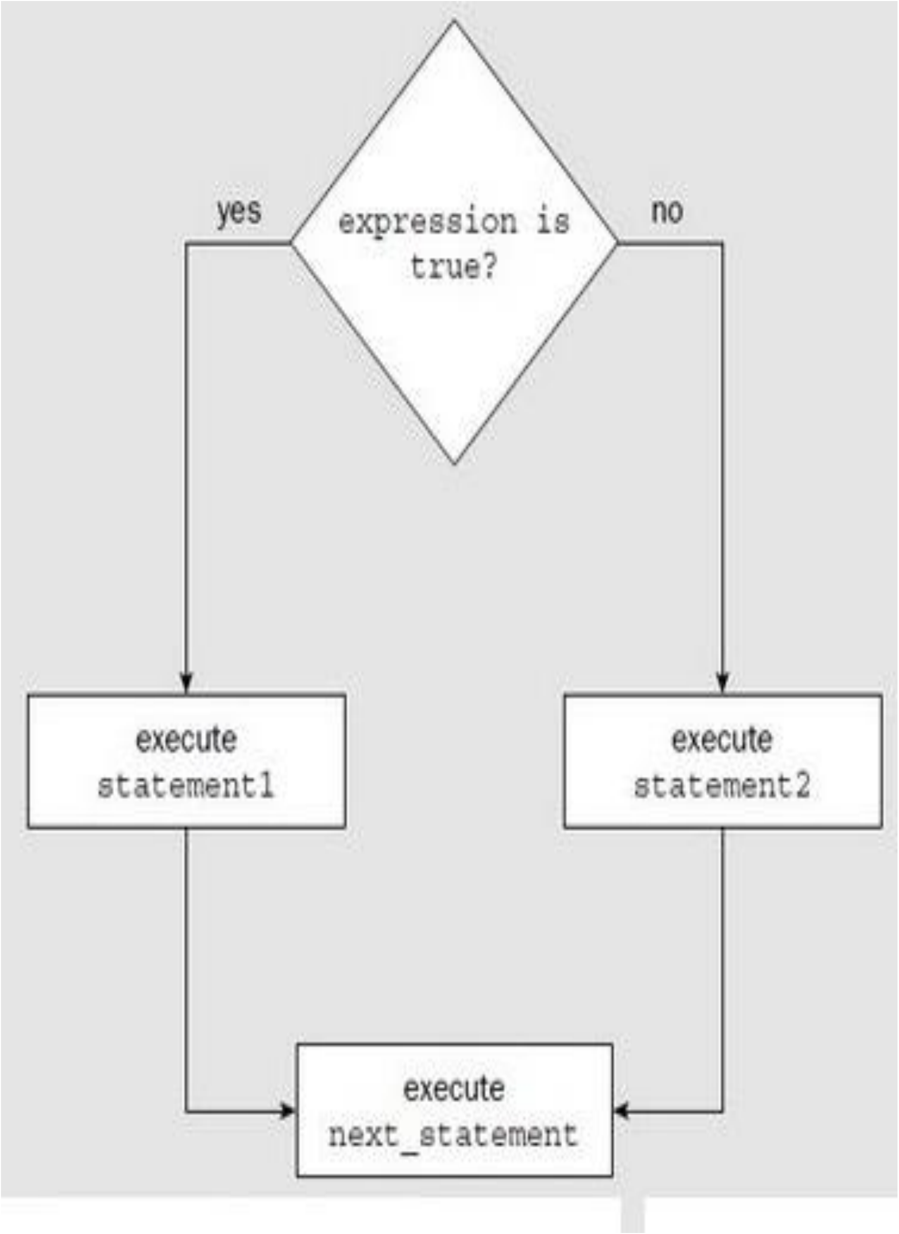
```
if(expression)
 {
  statement1;
  statement2;
}
 else
 {
statement3;
statement4;
 }
next_statement;
```

Explanation :

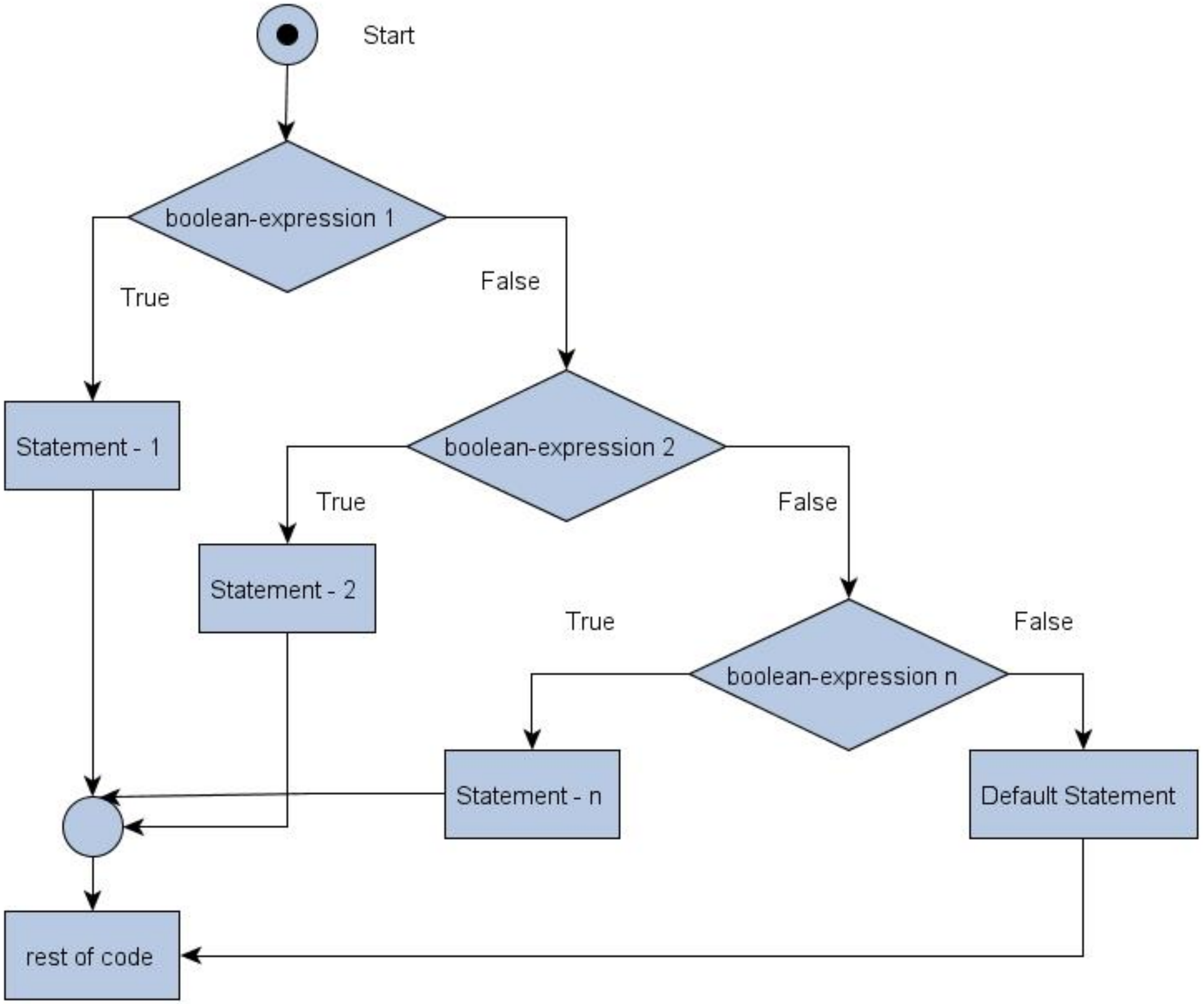
If expression is True then Statement1 and Statement2 are executed

Otherwise

Statement3 and Statement4 are executed.



if-else Ladder



Else-if Ladder statement flow chart

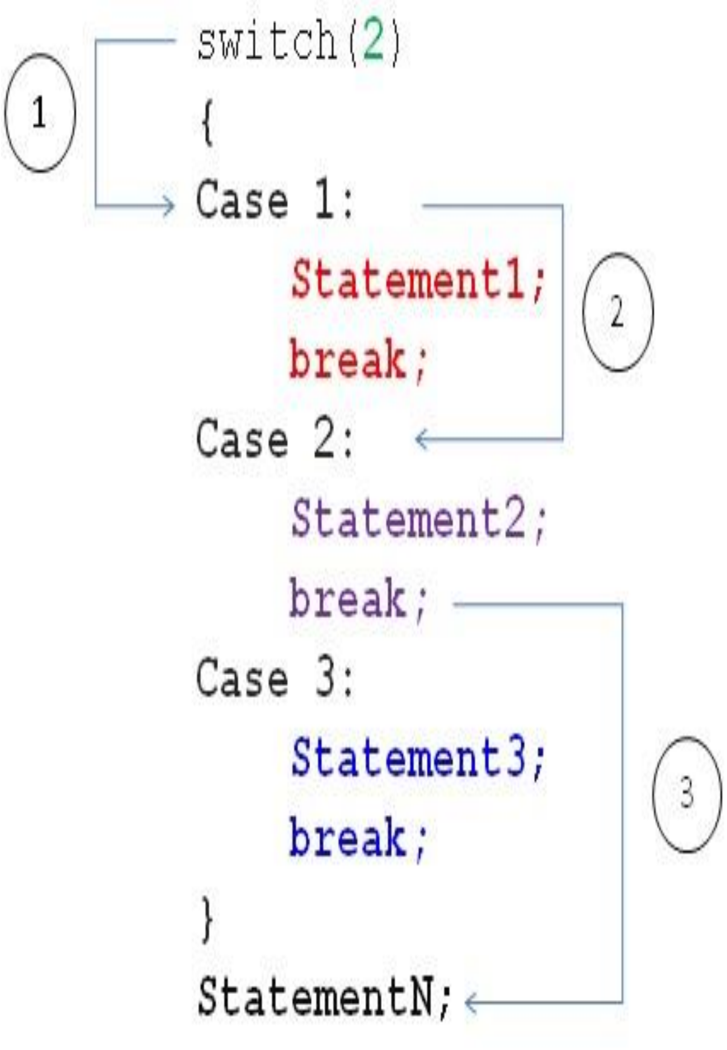
C switch case statement

Why we should use Switch Case

- One of the classic problem encountered in nested if-else / else-if ladder is called problem of Confusion.
- It occurs when no matching else is available for if .
- As the number of alternatives increases the Complexity of program increases drastically.
- To overcome this , C Provide a multi-way decision statement called ‘Switch Statement‘

How it works ?

- Switch case checks the value of expression/variable against the list of case values and when the match is found , **the block of statement associated with that case is executed**
- Expression should be Integer **Expression / Character**
- **Break statement takes** control out of the case.
- Break Statement is **Optional**.



```
switch(2)
{
Case 1:
        Statement1;
        break;
Case 2:
        Statement2;
        break;
Case 3:
        Statement3;
        break;
}
StatementN;
```

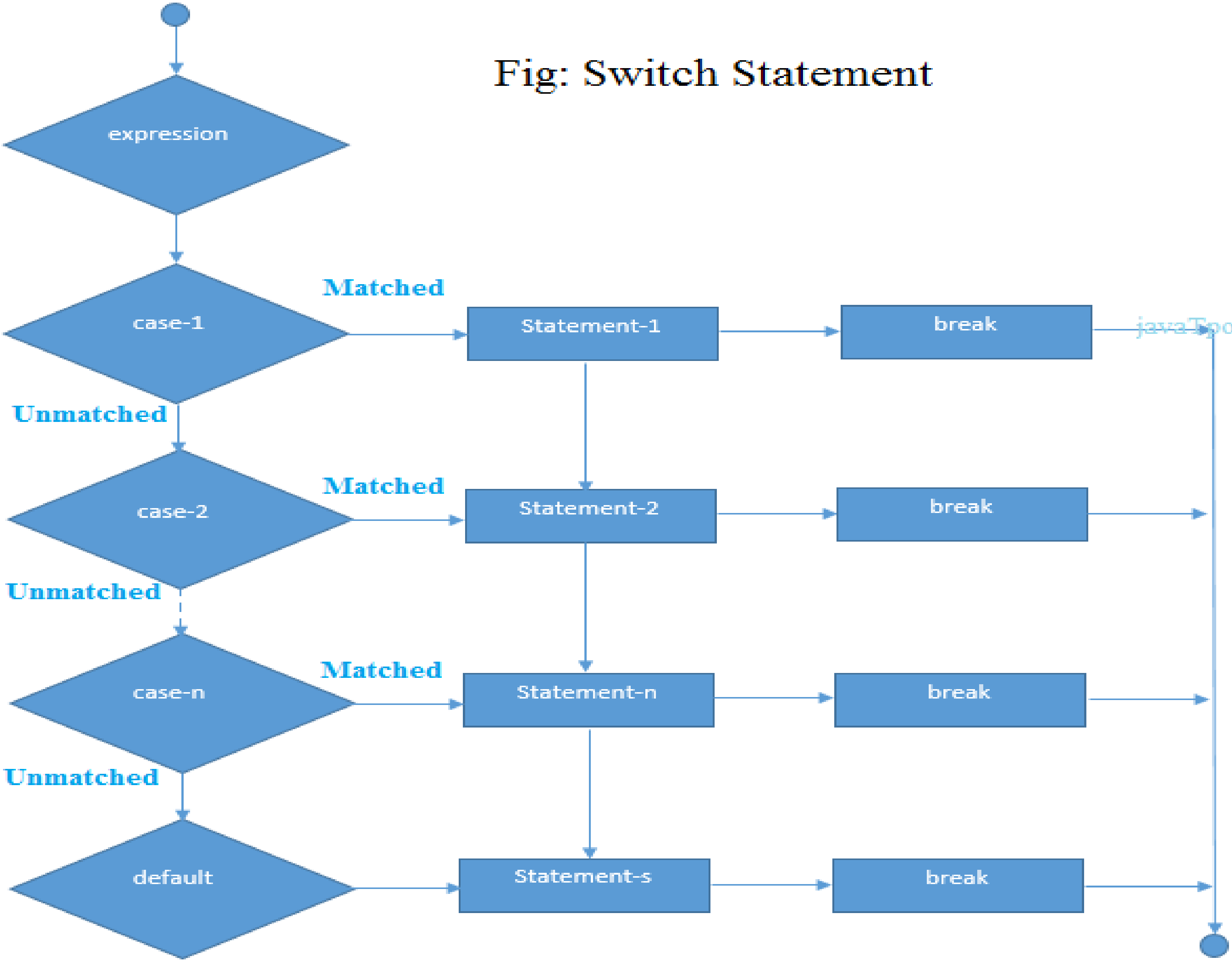


Fig: Switch Statement

Rules of Using Switch Case in C Programming

- Case Label **must be unique**
- Case Labels must **ends with Colon**
- Case labels must **have constants / constant expression**
- Case label must be of **integral Type (Integer,Character)**
- Case label should not be ‘floating point number ‘
- Switch case should have at most one default label
- Default label is Optional
- Default **can be placed anywhere in the switch**
- Break Statement takes control out of the switch
- **Two or more cases** may share **one break statement**
- **Nesting (switch within switch)** is allowed.
- Relational Operators are not allowed in Switch Statement.
- Const Variable is allowed in switch Case Statement.
- Empty Switch case is allowed.

	If-else	switch
Definition	Depending on the condition in the 'if' statement, 'if' and 'else' blocks are executed.	The user will decide which statement is to be executed.
Expression	It contains either logical or equality expression.	It contains a single expression which can be either a character or integer variable.
Evaluation	It evaluates all types of data, such as integer, floating-point, character or Boolean.	It evaluates either an integer, or character.
Sequence of execution	First, the condition is checked. If the condition is true then 'if' block is executed otherwise 'else' block	It executes one case after another till the break keyword is not found, or the default statement is executed.
Default execution	If the condition is not true, then by default, else block will be executed.	If the value does not match with any case, then by default, default statement is executed.
Editing	Editing is not easy in the 'if-else' statement.	Cases in a switch statement are easy to maintain and modify. Therefore, we can say that the removal or editing of any case will not interrupt the execution of other cases.
Speed	If there are multiple choices implemented through 'if-else', then the speed of the execution will be slow.	If we have multiple choices then the switch statement is the best option as the speed of the execution will be much higher than 'if-else'.

C Loops

- The looping can be defined as repeating the same process multiple times until a specific condition satisfies.

Why use loops in C language?

- The looping simplifies the complex problems into the easy ones. It enables us to alter the flow of the program so that instead of writing the same code again and again, we can repeat the same code for a finite number of times.
- For example, if we need to print the first 10 natural numbers then, instead of using the printf statement 10 times, we can print inside a loop which runs up to 10 iterations.

Advantage of loops in C

1) It provides code reusability.

2) Using loops, we do not need to write the same code again and again.

3) Using loops, we can traverse over the elements of data structures (array or linked lists).

Types of C Loops

There are three types of loops in C language that is given below:

- for
- while
- do while

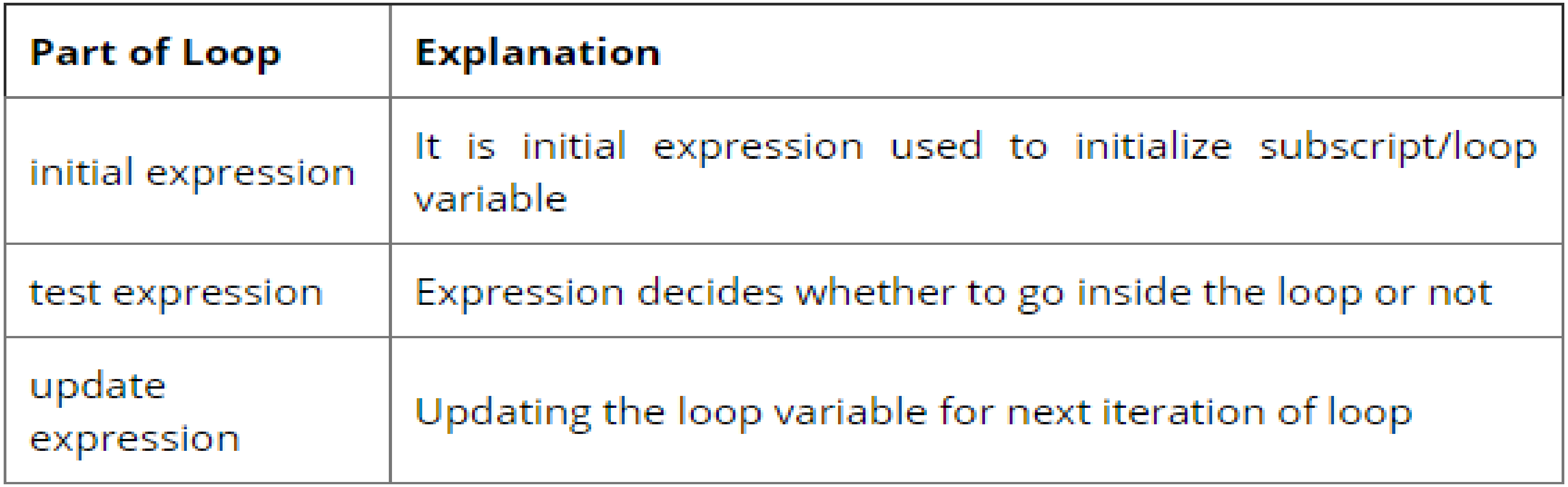
for loop

- Whenever we need to execute certain action multiple times, we need to wrap programming statement in the for loop body.
- The for loop is also called as a per-tested loop. It is better to use for loop if the number of iteration is known in advance.

Syntax

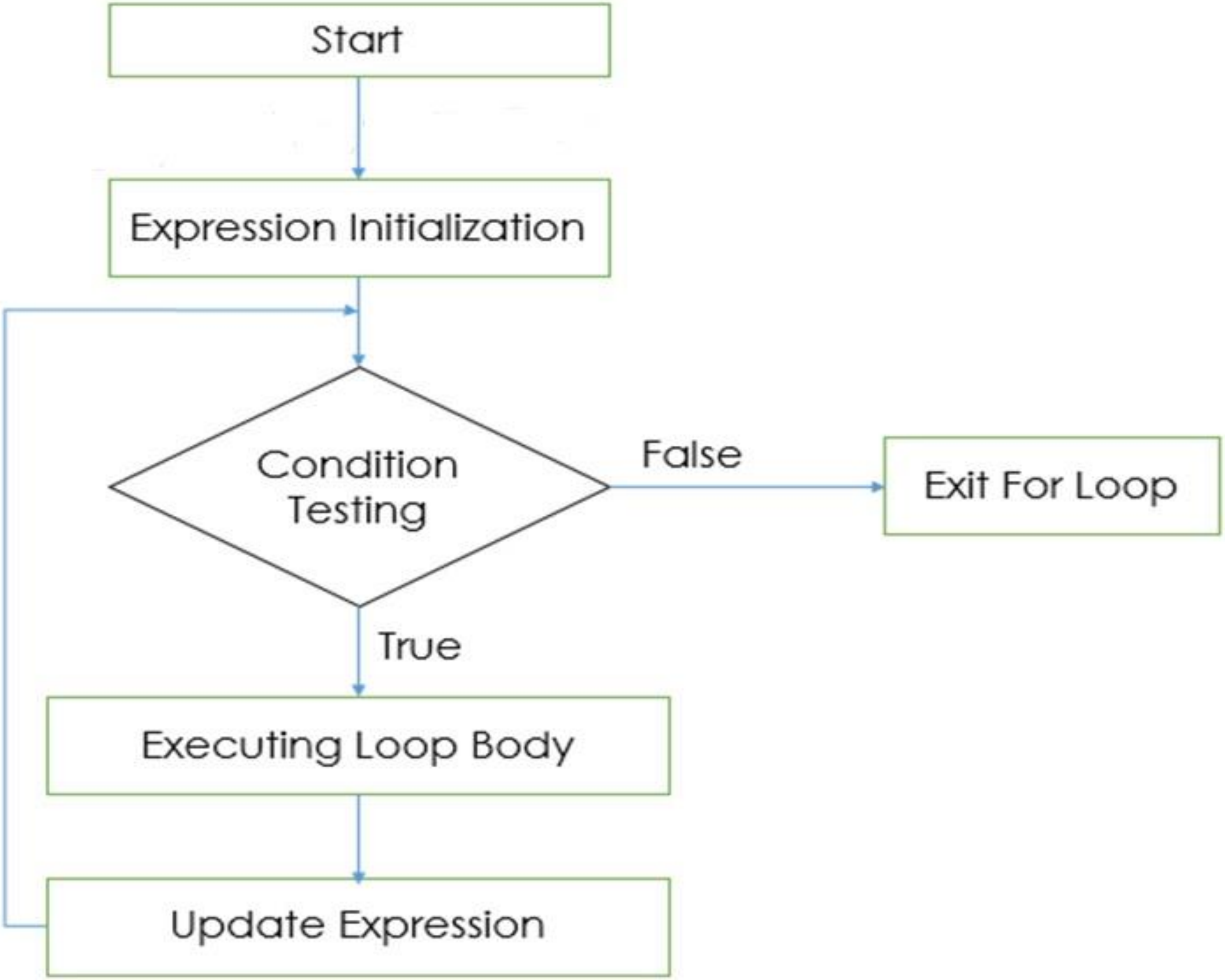
- **for**(initial expression; test expression; update expression)

```
{
        body of loop;
 }
```



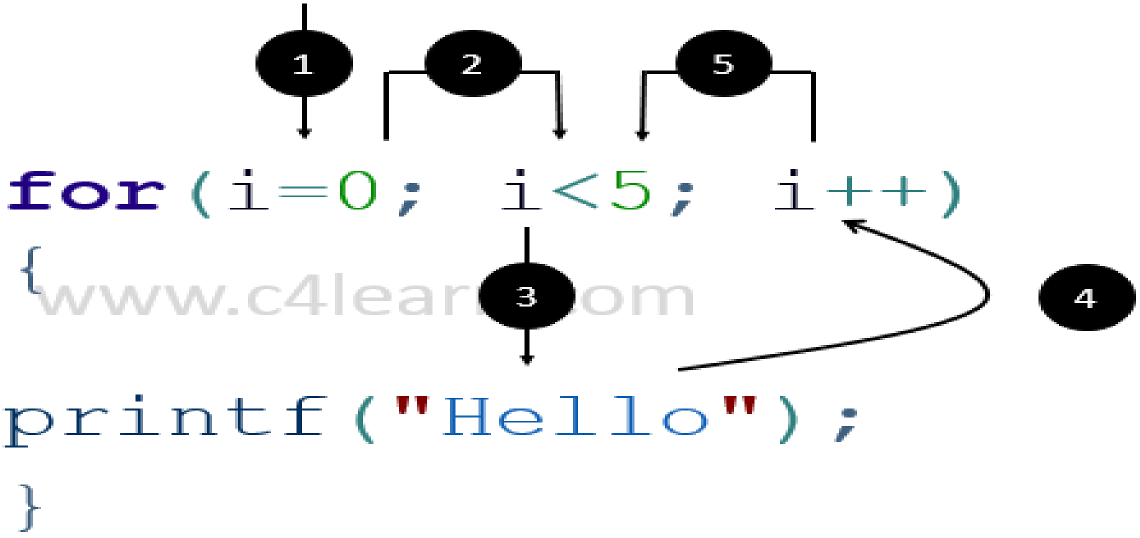
Part of Loop	**Explanation**
initial expression	It is initial expression used to initialize subscript/loop variable
test expression	Expression decides whether to go inside the loop or not
update expression	Updating the loop variable for next iteration of loop

Flow chart- for loop



Explanation of For Loop :

- Firstly the for loop executes the initialize statement in which the subscript variable will be initialized with the initial value.
- After the initialize statement, the condition part of the loop will be executed if the condition becomes true then body of the loop will be executed otherwise the loop will be terminated
- If the loop condition becomes true then body of the loop will be executed. After the execution of the for loop body the control goes to the third part of the loop statement i.e Expression Updation
- After updating subscript variable control again goes to execute condition statement.

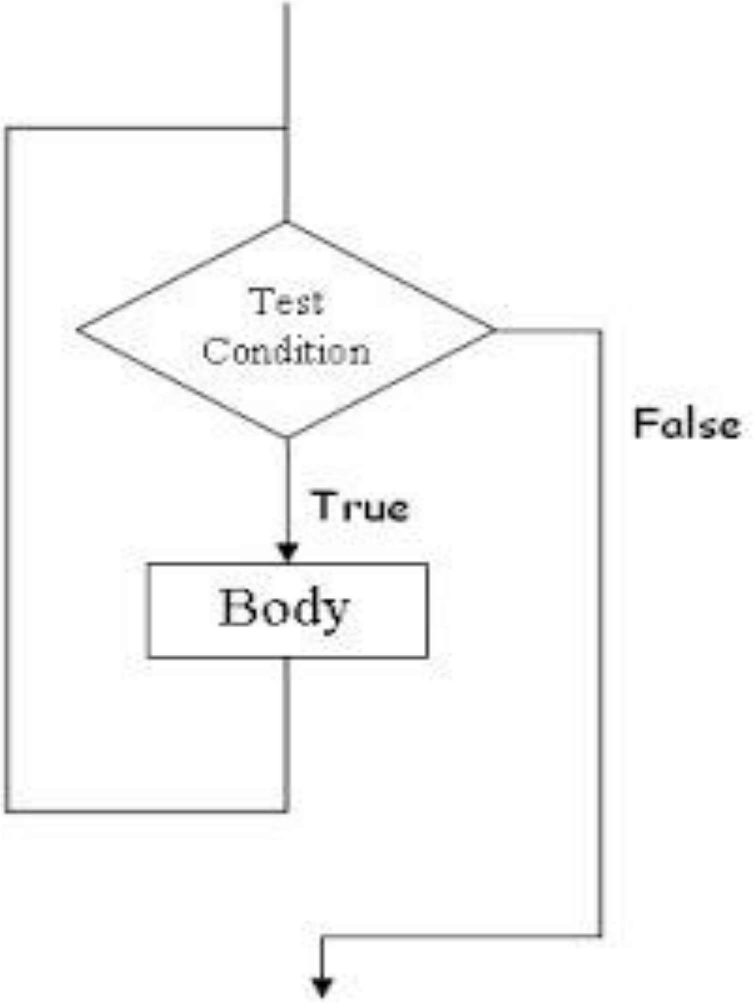


```
for(i=0; i<5; i++)
{
printf("Hello");
}
```

while loop

The while loop in c is to be used in the scenario where we don't know the number of iterations in advance. The block of statements is executed in the while loop until the condition specified in the while loop is satisfied. It is also called a pre-tested loop.

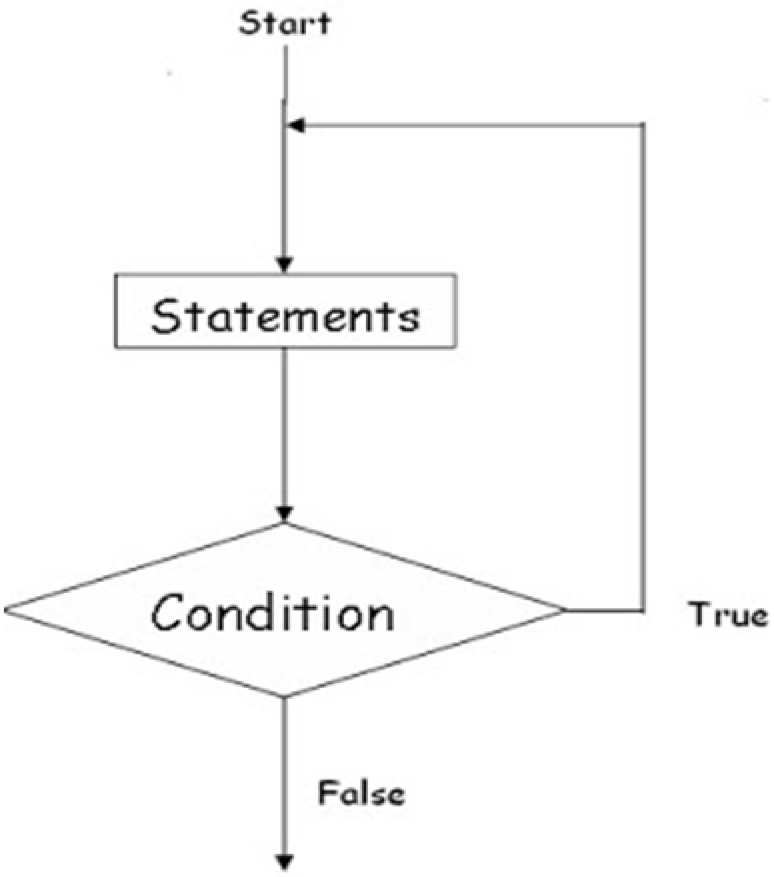
```
initialization;
 while(condition)
{
          ----------
          ----------
          ----------
          ----------
          ----------
          ------
 incrementation;
}
```



do-while loop

The do-while loop continues until a given condition satisfies. It is also called post tested loop. It is used when it is necessary to execute the loop at least once

```
initialization;
do
{
    --------------
     --------------
     --------------
    --------------
incrementation;
 }
while(condition);
```



Difference b/w while loop & do-while loop

- The **difference** is in when the condition gets evaluated.
- In a **do**..**while loop**, the condition is not evaluated **until** the end of each **loop**. That means that a **do**..**while loop** will always run at least once.
- In a **while loop**, the condition is evaluated at the start.

C break statement

- The break is a keyword in C which is used to bring the program control out of the loop.
- The break statement is used inside loops or switch statement.
- The break statement breaks the loop one by one, i.e., in the case of nested loops, it breaks the inner loop first and then proceeds to outer loops.
- The break statement in C can be used in the following two scenarios:

➢ With switch case

➢ With loop

Syntax:

```
//loop or switch case
break;
```

C continue statement

- The **continue statement** in C language is used to bring the program control to the beginning of the loop. The continue statement skips some lines of code inside the loop and continues with the next iteration. It is mainly used for a condition so that we can skip some code for a particular condition.

Syntax:

//loop statements

- **continue**;

//some lines of the code which is to be skipped

C goto statement

- The goto statement is known as jump statement in C.
- As the name suggests, goto is used to transfer the program control to a predefined label.
- The goto statment can be used to repeat some part of the code for a particular condition.
- It can also be used to break the multiple loops which can't be done by using a single break statement.
- However, using goto is avoided these days since it makes the program less readable and complicated.

Syntax:

```
label:
//some part of the code;
goto label;
```