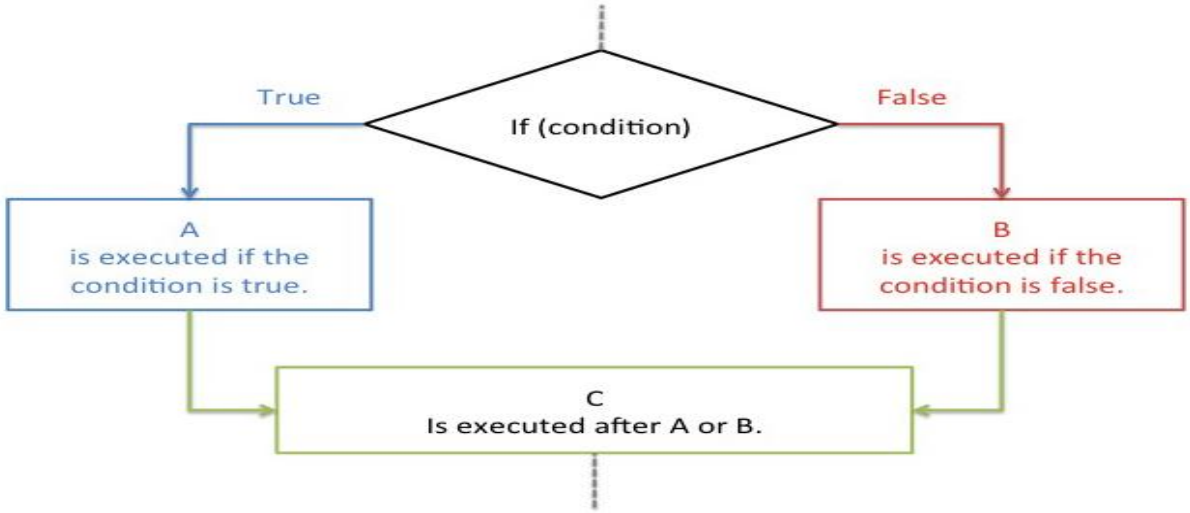
Topic1

Loop and Conditional Statements

A conditional statement is a mechanism that allows for conditional execution of instructions based upon the outcome of a conditional statement, which can either be true or false.



Different Type of Conditional Statement

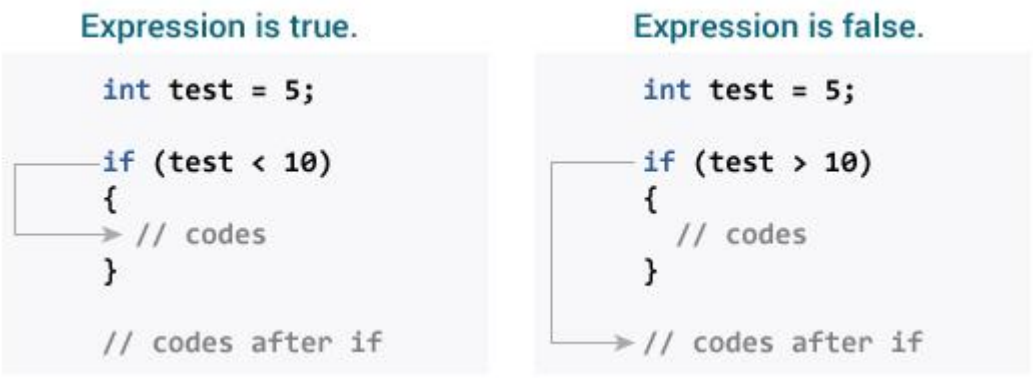
- if statement
- if...else statement
- if...else Ladder (if...else if.... else Statement)
- Nested if...else

……

How if statement works?

The `if` statement evaluates the test expression inside the parenthesis.

- If the test expression is evaluated to true (nonzero), statement(s) inside the body of “if” is executed.
- If the test expression is evaluated to false (0), statement(s) inside the body of “`if`” is skipped from execution.



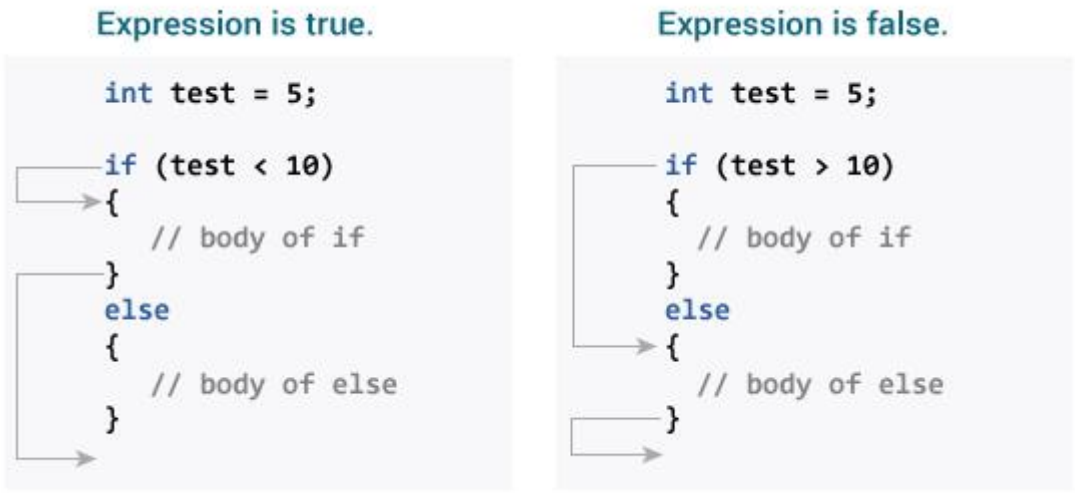
How if...else statement works?

If test expression is evaluated to true,

- statement(s) inside the body of `if` statement is executed
- statement(s) inside the body of `else` statement is skipped from execution.

If test expression is evaluated to false,

- statement(s) inside the body of `else` statement is executed
- statement(s) inside the body of `if` statement is skipped.



if...else Ladder (if...else if.... else Statement)

The `if...else` statement executes two different codes depending upon whether the test expression is true or false. Sometimes, a choice has to be made from more than 2 possibilities.

The if...else ladder allows you to check for multiple test expressions and execute different statement(s).

Nested if...else

It is possible to include `if...else` statement(s) inside the body of another `if...else` statement.

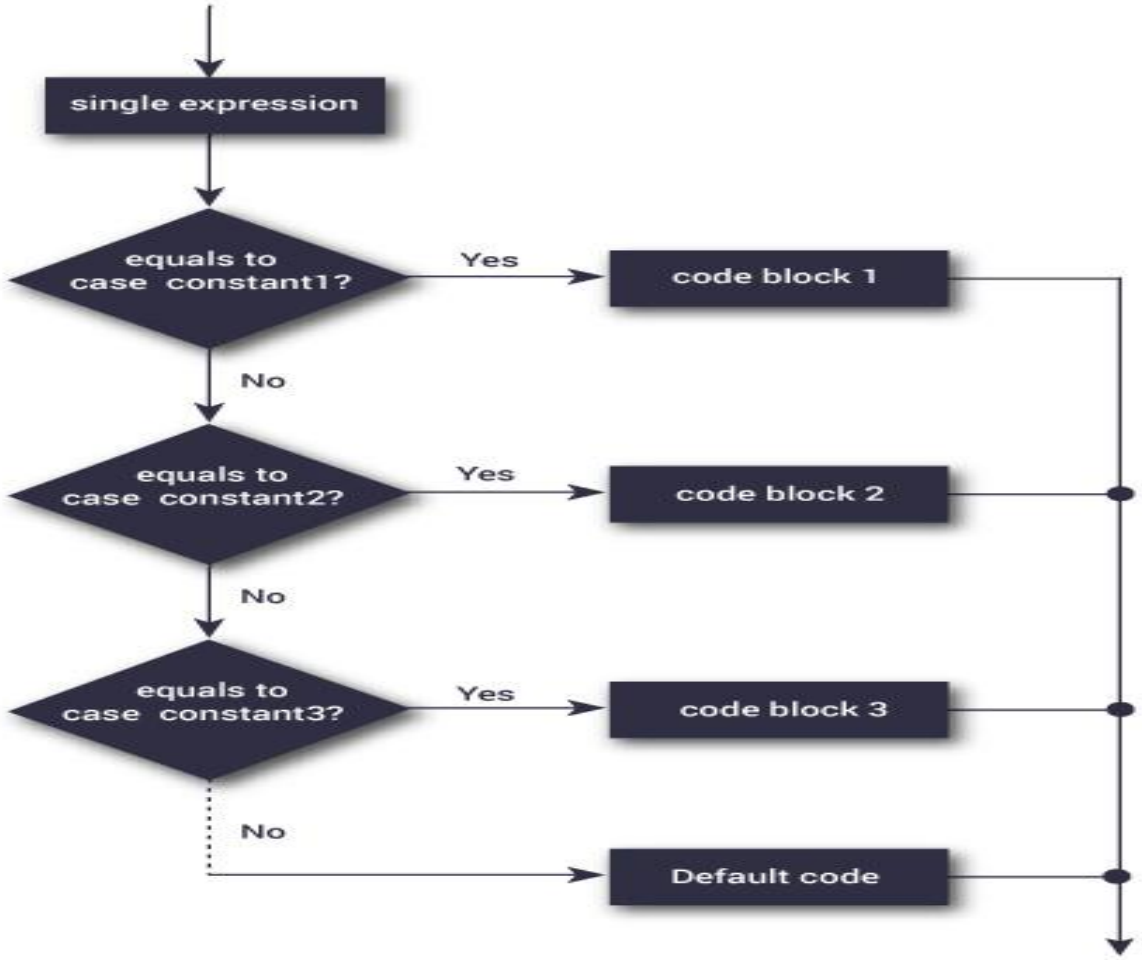
This program below relates two integers using either `<`, `>` and `=` similar like in `if...else` ladder example. However, we will use nested `if...else` statement to solve this problem.

Topic 2

switch...case Statement

The `if... else... if` ladder allows you to execute a block code among many alternatives. If you are checking on the value of a single variable in `if...else...if`, it is better to use `switch` statement.

***The switch statement is often faster than nested `if...else` (not always). Also, the syntax of switch statement is cleaner and easy to understand.



Loop

Loops are used in programming to repeat a block of code until a specific condition is met. There are three loops in C programming:

Different Type of loop

- *While*
- *Do while*
- *For*

Programming for Loop

```
for (initialization Statement; test Expression; update Statement)

{
```

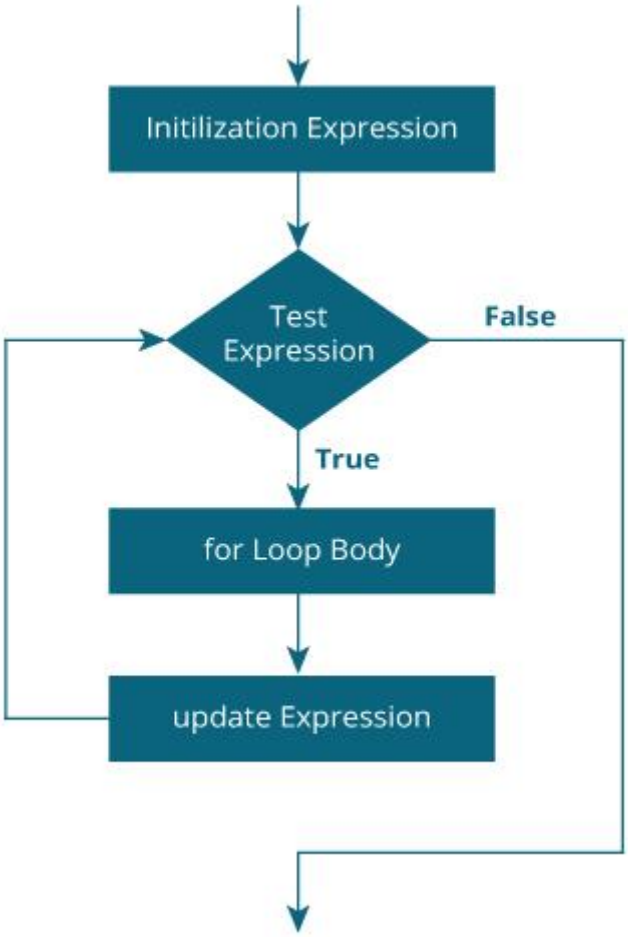
```
    // codes

}
```

The initialization statement is executed only once.

Then, the test expression is evaluated. If the test expression is false (0), for loop is terminated. But if the test expression is true (nonzero), codes inside the body of `for` loop is executed and the update expression is updated.

This process repeats until the test expression is false.

The `for` loop is commonly used when the number of iterations is known.



Programming while

while loop

The syntax of a while loop is:

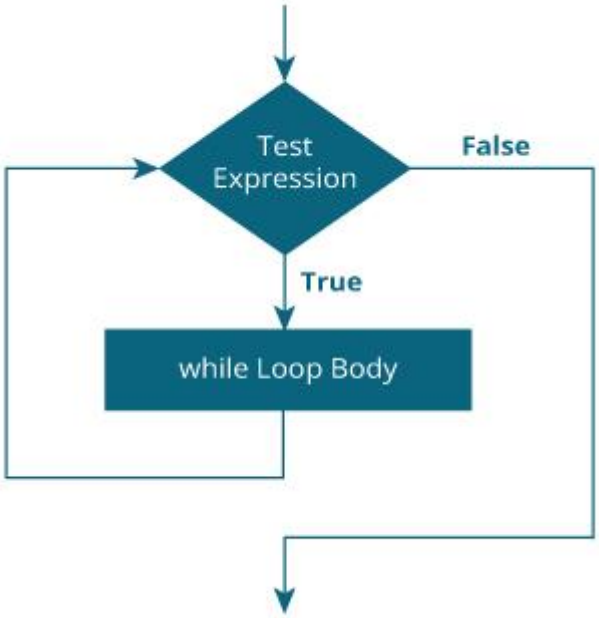
```
while (test Expression)

{

    //codes

}
```

How while loop works?

The while loop evaluates the test expression.

If the test expression is true (nonzero), codes inside the body of while loop are executed. The test expression is evaluated again. The process goes on until the test expression is false.

When the test expression is false, the while loop is terminated.



do...while loop

The `do….. While` loop is similar to the `while` loop with one important difference. The body of `do...while` loop is executed once, before checking the test expression. Hence, the `do...while loop` is executed at least once.

do...while loop Syntax

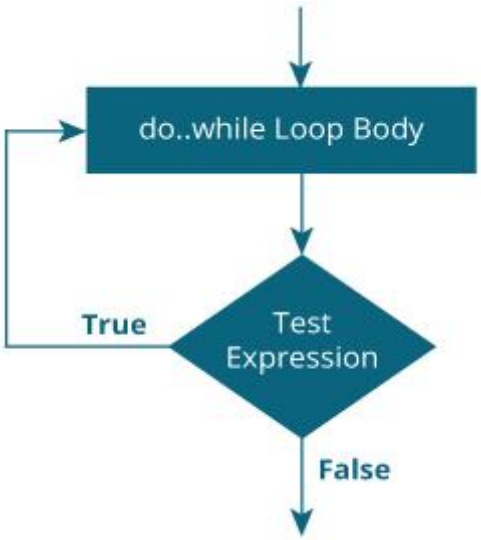
```
do

{

   // codes

}

while (test Expression);
```

How do...while loop works?

The code block (loop body) inside the braces is executed once.

Then, the test expression is evaluated. If the test expression is true, the loop body is executed again. This process goes on until the test expression is evaluated to 0 (false).

When the test expression is false (nonzero), the `do...while` loop is terminated.



break and continue Statement

It is sometimes desirable to skip some statements inside the loop or terminate the loop immediately without checking the test expression.

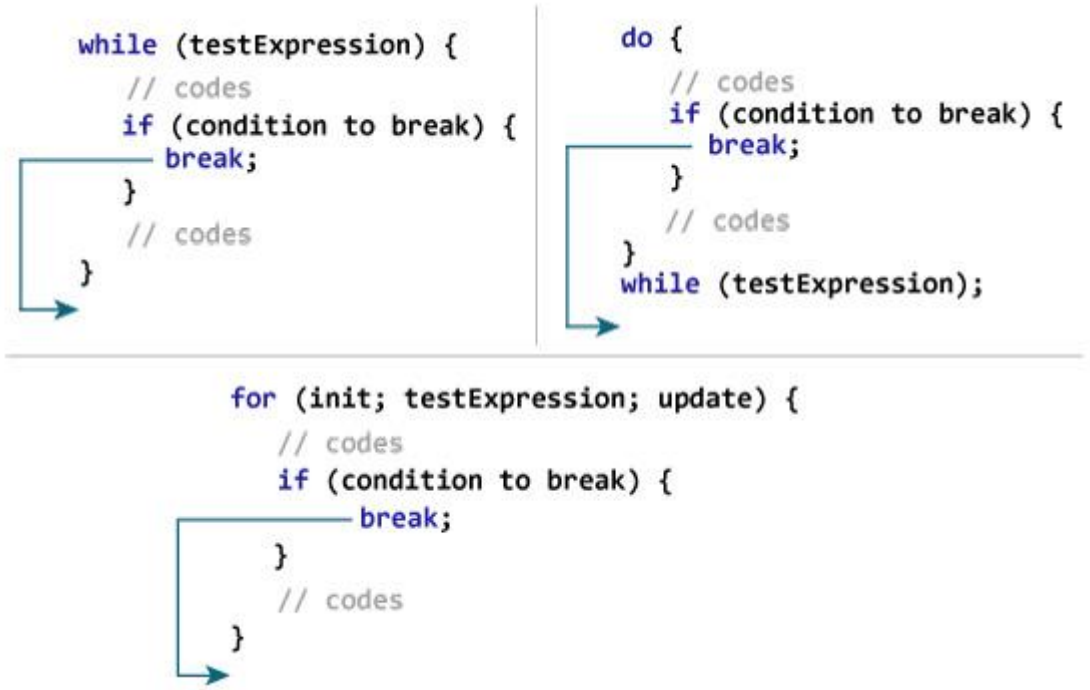
In such cases, `break` and `continue` statements are used.

break Statement

The break statement terminates the loop (for, while and do...while loop) immediately when it is encountered. Its syntax is:

```
break;
```

****The break statement is almost always used with if...else statement inside the loop.

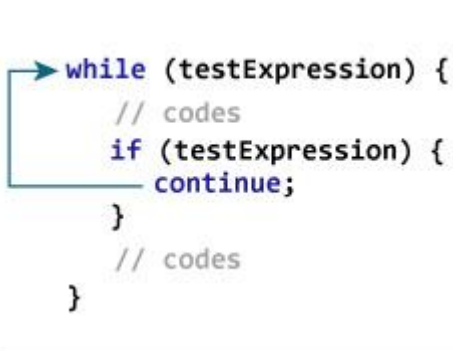


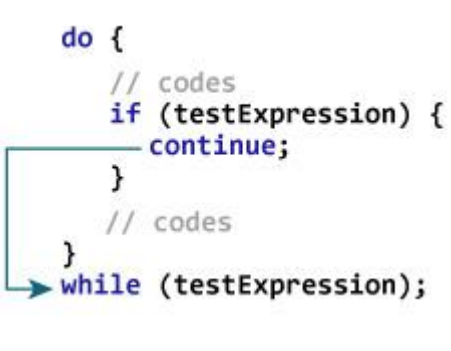
continue Statement

The continue statement skips statements after it inside the loop. Its syntax is:

```
continue;
```

The continue statement is almost always used with `if...else` statement.


while (testExpression) {
// codes
if (testExpression) {
continue;
}
// codes
}


do {
// codes
if (testExpression) {
continue;
}
// codes
}
while (testExpression);


for (init; testExpression; update) {
// codes
if (testExpression) {
continue;
}
// codes
}