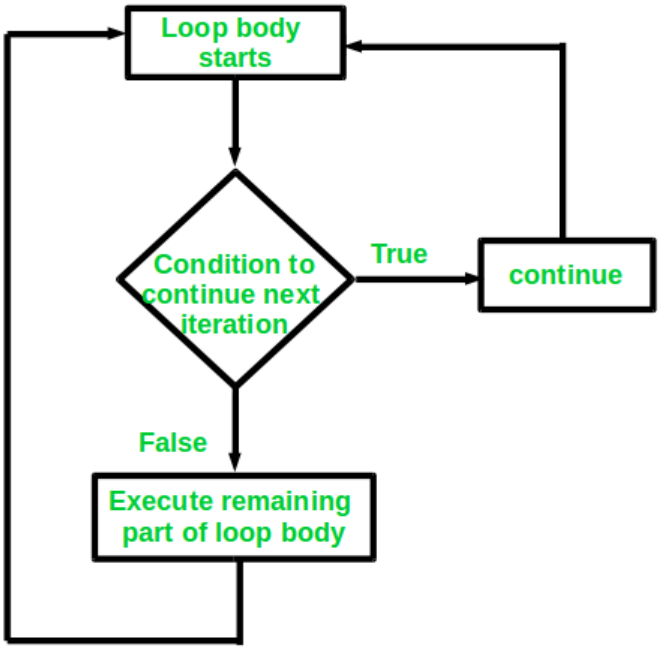
Continue in C++

Continue is also a loop control statement just like the break statement. *continue* statement is opposite to that of break *statement*, instead of terminating the loop, it forces to execute the next iteration of the loop.

As the name suggest the continue statement forces the loop to continue or execute the next iteration. When the continue statement is executed in the loop, the code inside the loop following the continue statement will be skipped and next iteration of the loop will begin.

Syntax:

```
continue;
```



Example:

Consider the situation when you need to write a program which prints number from 1 to 10 and but not 6. It is specified that you have to do this using loop and only one loop is allowed to use.

Here comes the usage of continue statement. What we can do here is we can run a loop from 1 to 10 and every time we have to compare the value of iterator with 6. If it is equal to 6 we will use the *continue* statement to continue to next iteration without printing anything otherwise we will print the value.

Below is the implementation of the above idea:

```
// C++ program to explain the use
// of continue statement

#include <iostream>
using namespace std;

int main()
{
    // loop from 1 to 10
    for (int i = 1; i <= 10; i++) {

        // If i is equals to 6,
        // continue to next iteration
        // without printing
        if (i == 6)
            continue;

        else
            // otherwise print the value of i
            cout << i << " ";
    }

    return 0;
}
```

Output

```
1 2 3 4 5 7 8 9 10
```