- **goto STATEMENT**

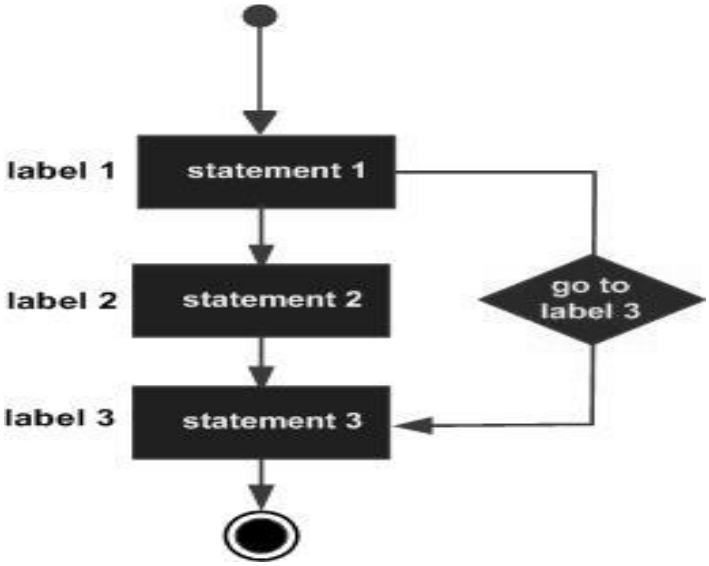
A **goto** statement in C programming language provides an unconditional jump from the **goto** to a **labeled** statement in the same function.

The syntax for a **goto** statement in C is as follows:

```
goto label1;
        Statement 1;
        Statement 2;
label1:
        Statement 3;
```

Here **label** can be any plain text except C keyword and it can be set anywhere in the C program above or below to **goto** statement.

Flow Diagram:




```
// A simple program to use of goto statement
#include<stdio.h>
#include<conio.h>
void main ()
{
/* local variable definition */
int a =10;

/* do loop execution */
   LOOP:do
{
if( a ==15)
{
/* skip the iteration */
```

```
        a = a +1;
goto LOOP;
}
     printf("value of a: %d\n", a);
     a++;

}while( a <20);
}
```

OUTPUT:

```
value of a: 10
valueof a: 11
valueof a: 12
valueof a: 13
valueof a: 14
valueof a: 15
valueof a: 16
valueof a: 17
valueof a: 18
valueof a: 19
```

Break Statement

The **break** statement in C programming has the following two usages −

- When a **break** statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.
- It can be used to terminate a case in the **switch** statement (covered in the next chapter).

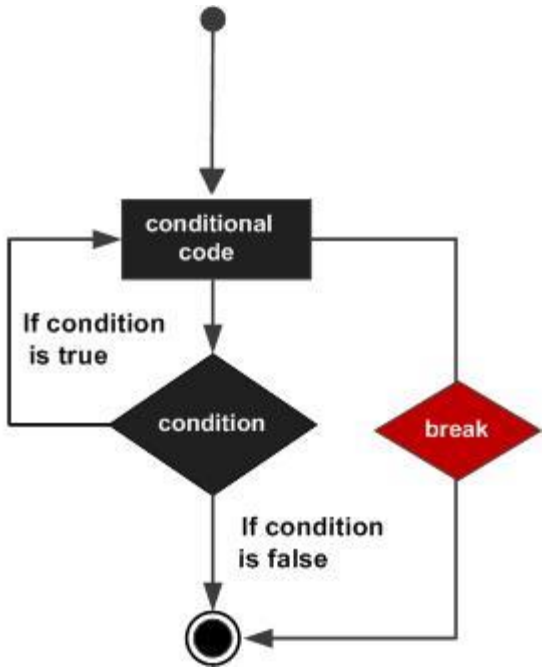
If you are using nested loops, the break statement will stop the execution of the innermost loop and start executing the next line of code after the block.

Syntax

The syntax for a **break** statement in C is as follows −

```
break;
```

Flow Diagram



Example

```
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 10;

   /* while loop execution */
   while( a < 20 ) {

      printf("value of a: %d\n", a);
      a++;
if( a > 15) {
         /* terminate the loop using break statement */
         break;
      }
   }

   return 0;
}
```

When the above code is compiled and executed, it produces the following result −

value of a: 10

value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15

Continue Statement

The **continue** statement in C programming works somewhat like the **break** statement. Instead of forcing termination, it forces the next iteration of the loop to take place, skipping any code in between.

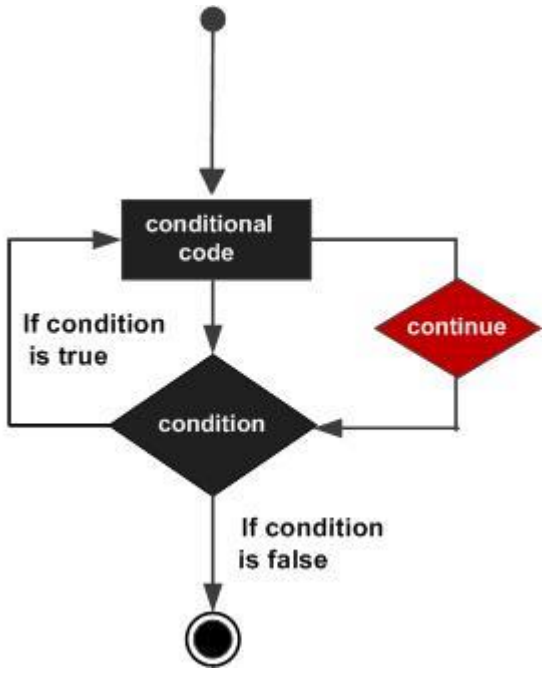
For the **for** loop, **continue** statement causes the conditional test and increment portions of the loop to execute. For the **while** and **do...while** loops, **continue** statement causes the program control to pass to the conditional tests.

Syntax

The syntax for a **continue** statement in C is as follows −

Continue;

Flow Diagram



Example

```
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 10;

   /* do loop execution */
   do {

      if( a == 15) {
         /* skip the iteration */
         a = a + 1;
         continue;
      }
printf("value of a: %d\n", a);
      a++;

   } while( a < 20 );

   return 0;
}
```

When the above code is compiled and executed, it produces the following result −

```
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 16
value of a: 17
value of a: 18
value of a: 19
```