**C – continue statement with example**

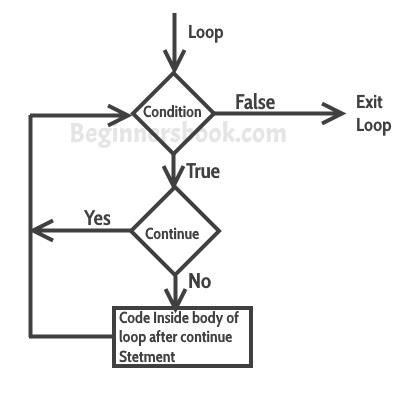
The **continue statement** is used inside [loops](https://beginnersbook.com/2014/01/c-loops-examples/). When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for next iteration, skipping the execution of statements inside the body of loop for the current iteration.

**C – Continue statement**

Syntax:

continue;

**Flow diagram of continue statement**



### Example: continue statement inside for loop

#include <stdio.h>

int main()

{

for (int j=0; j<=8; j++)

{

if (j==4)

{

/\* The continue statement is encountered when

\* the value of j is equal to 4.

\*/

continue;

}

/\* This print statement would not execute for the

\* loop iteration where j ==4 because in that case

\* this statement would be skipped.

\*/

printf("%d ", j);

}

return 0;

}

Output:

0 1 2 3 5 6 7 8

Value 4 is missing in the output, why? When the value of variable j is 4, the program encountered a continue statement, which makes the control to jump at the beginning of the for loop for next iteration, skipping the statements for current iteration (that’s the reason printf didn’t execute when j is equal to 4).

### Example: Use of continue in While loop

In this example we are using continue inside while loop. When using while or do-while loop you need to place an increment or decrement statement just above the continue so that the counter value is changed for the next iteration. For example, if we do not place counter– statement in the body of “if” then the value of counter would remain 7 indefinitely.

#include <stdio.h>

int main()

{

int counter=10;

while (counter >=0)

{

if (counter==7)

{

counter--;

continue;

}

printf("%d  ", counter);

counter--;

}

return 0;

}

Output:

10 9 8 6 5 4 3 2 1 0

The print statement is skipped when counter value was 7.

### Another Example of continue in do-While loop

#include <stdio.h>

int main()

{

int j=0;

do

{

if (j==7)

{

j++;

continue;

}

printf("%d ", j);

j++;

}while(j<10);

return 0;

}

Output:

0 1 2 3 4 5 6 8 9

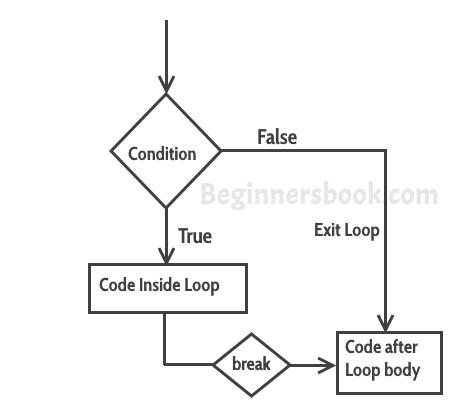
# C – break statement in C programming

The break statement is used inside loops and [switch case](https://beginnersbook.com/2014/01/switch-case-statements-in-c/).

## C – break statement

1. It is used to come out of the loop instantly. When a break statement is encountered inside a loop, the control directly comes out of loop and the loop gets terminated. It is used with [if statement](https://beginnersbook.com/2014/01/c-if-statement/), whenever used inside loop.  
2. This can also be used in switch case control structure. Whenever it is encountered in switch-case block, the control comes out of the switch-case(see the example below).

### Flow diagram of break statement



Syntax:

break;

### Example – Use of break in a while loop

#include <stdio.h>

int main()

{

int num =0;

while(num<=100)

{

printf("value of variable num is: %d\n", num);

if (num==2)

{

break;

}

num++;

}

printf("Out of while-loop");

return 0;

}

**Output:**

value of variable num is: 0

value of variable num is: 1

value of variable num is: 2

Out of while-loop

### Example – Use of break in a for loop

#include <stdio.h>

int main()

{

int var;

for (var =100; var>=10; var --)

{

printf("var: %d\n", var);

if (var==99)

{

break;

}

}

printf("Out of for-loop");

return 0;

}

**Output:**

var: 100

var: 99

Out of for-loop

### Example – Use of break statement in switch-case

#include <stdio.h>

int main()

{

int num;

printf("Enter value of num:");

scanf("%d",&num);

switch (num)

{

case 1:

printf("You have entered value 1\n");

break;

case 2:

printf("You have entered value 2\n");

break;

case 3:

printf("You have entered value 3\n");

break;

default:

printf("Input value is other than 1,2 & 3 ");

}

return 0;

}

Output:

Enter value of num:2

You have entered value 2

You would always want to use break statement in a switch case block, otherwise once a case block is executed, the rest of the subsequent case blocks will execute. For example, if we don’t use the break statement after every case block then the output of this program would be:

Enter value of num:2

You have entered value 2

You have entered value 3

Input value is other than 1,2 & 3

# C – goto statement with example

The goto statement is rarely used because it makes program confusing, less readable and complex. Also, when this is used, the control of the program won’t be easy to trace, hence it makes testing and debugging difficult.

## C – goto statement

When a goto statement is encountered in a C program, the control jumps directly to the label mentioned in the goto stateemnt  
**Syntax of goto statement in C**

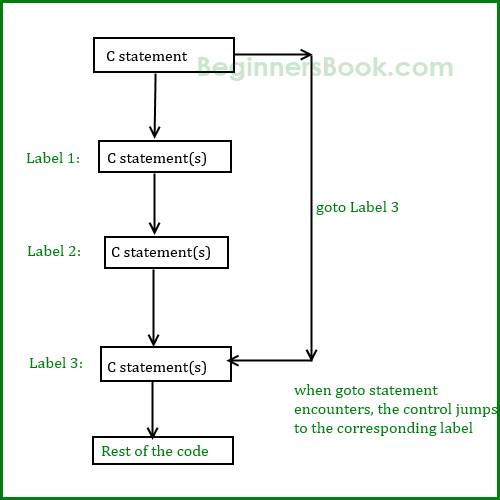
goto label\_name;

..

..

label\_name: C-statements

### Flow Diagram of goto



### Example of goto statement

#include <stdio.h>

int main()

{

int sum=0;

for(int i = 0; i<=10; i++){

sum = sum+i;

if(i==5){

goto addition;

}

}

addition:

printf("%d", sum);

return 0;

}

Output:

15

**Explanation:** In this example, we have a label addition and when the value of i (inside loop) is equal to 5 then we are jumping to this label using goto. This is reason the sum is displaying the sum of numbers till 5 even though the loop is set to run from 0 to 10.