break, continue and goto statements

The `break;` `continue;` and `goto;` statements are used to alter the normal flow of a program. Loops perform a set of repetitive task until text expression becomes false but it is sometimes desirable to skip some statement/s inside loop or terminate the loop immediately without checking the test expression. In such cases, break and continue statements are used.

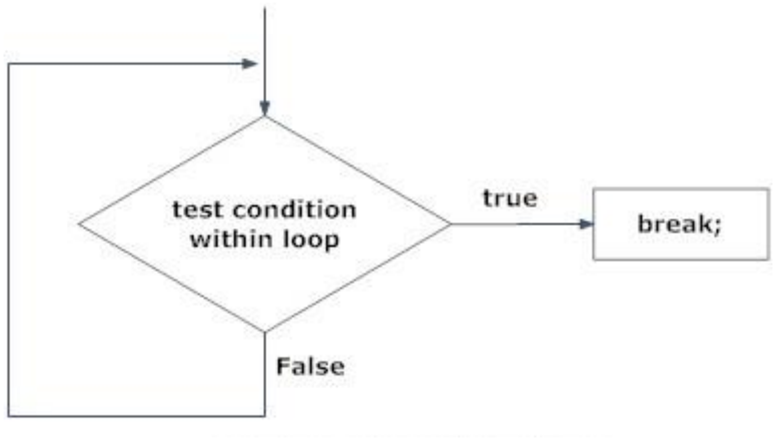
break statement

In C programming, break statement is used with conditional if statement. The break is used in terminating the loop immediately after it is encountered. it is also used in switch...case statement.

Syntax:

```
break;
```

The break statement can be used in terminating loops like `for`, `while` and `do...while`



Example 1:

```
//Write a C Program Which uses break statement.
#include<stdio.h>
#include<conio.h>
void main(){
    int num, sum=0;
    int i,n;
    printf("Note: Enter Zero for break loop!\n");
    printf("Enter Number of inputs\n");

    scanf("%d",&n);
```

```c
    for(i=1;i<=n;++i){
        printf("Enter num%d: ",i);
        scanf("%d",&num);
        if(num==0) {
            break;      /*this breaks loop if num == 0 */
            printf("Loop Breaks\n");
        }

        sum=sum+num;
    }
    printf("Total is %d",sum);
    getch();
}
```

Output:

```
Command Prompt
Note: Enter Zero for break loop!
Enter Number of inputs
5
Enter num1: 5
Enter num2: 10
Enter num3: 0
Loop Breaks
Total is 15
```

Example 2

```c
#include<stdio.h>
#include<stdlib.h>
void main ()
{
    int i;
    for(i = 0; i<10; i++)
    {
        printf("%d ",i);
        if(i == 5)
        break;
    }
    printf("came outside of loop i = %d",i);

}
```

Output

```
0 1 2 3 4 5 came outside of loop i = 5
```

Example 3

```c
#include<stdio.h>
int main(){
int i=1,j=1;//initializing a local variable
for(i=1;i<=3;i++){
for(j=1;j<=3;j++){
printf("%d &d\n",i,j);
if(i==2 && j==2){
break;//will break loop of j only
}
}//end of for loop
return 0;
}
```

Output

```
1 1
1 2
1 3
2 1
2 2
3 1
3 2
3 3
```

continue statement

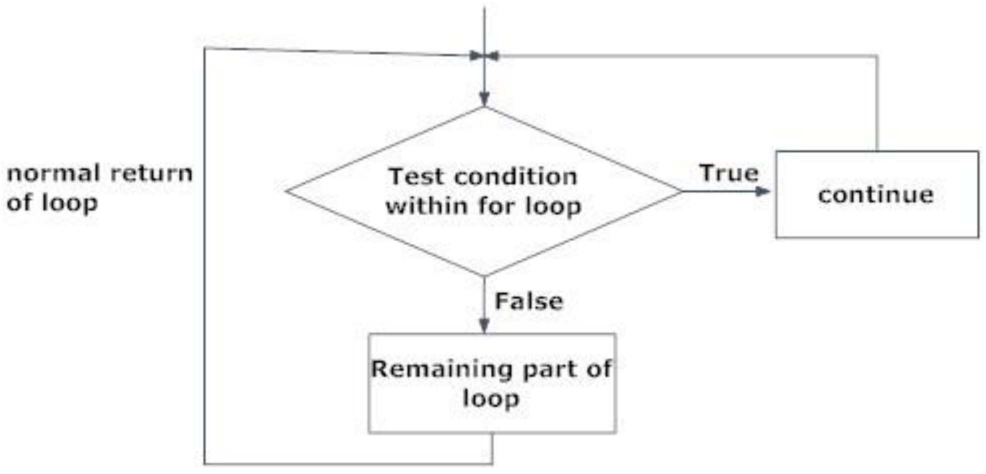
It is sometimes desirable to skip some statements inside the loop. In such cases, continue statement is used.

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:

```
continue;
```

Just like break, continue is also used with conditional if statement.



Example 1:

```
//Write a C Program Which uses continue statement.
#include<stdio.h>
#include<conio.h>
void main(){
    int i, n=20;
    clrscr();
    for(i=1;i<=n;++i){
        if(i % 5 == 0) {
            printf("pass\n");
            continue;      /*this continue the execution of loop if i % 5 == 0 */
        }
        printf("%d\n",i);
    }
    getch();
}
```

Download

Output:

```
Command Prompt
1
2
3
4
pass
6
7
8
9
pass
11
12
13
14
pass
16
17
18
19
pass
```

Continue statement Example 2

```c
#include<stdio.h>
void main ()
{
    int i = 0;
    while(i!=10)
    {
        printf("%d", i);
        continue;
        i++;
    }
}
```

Output

```
infinite loop
```

Continue statement Example 3

```c
#include<stdio.h>
int main(){
int i=1;//initializing a local variable
//starting a loop from 1 to 10
for(i=1;i<=10;i++){
if(i==5){//if value of i is equal to 5, it will continue the loop
continue;
}
printf("%d \n",i);
}//end of for loop
return 0;
}
```

Output

```
1
2
3
4
6
7
8
9
10
```

As you can see, 5 is not printed on the console because loop is continued at i==5.

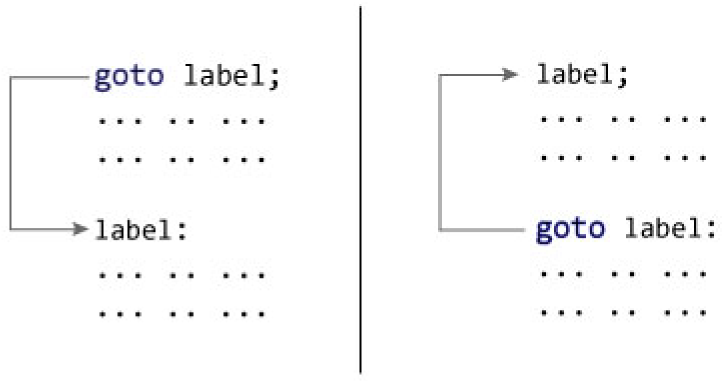
goto statement

In C programming, goto statement is used for altering the normal sequence of program execution by transferring control to some other part of the program.

Syntax:

```
goto label;
.............
.............
.............
label:
statement;
```

In this syntax, `label` is an identifier.
When, the control of program reaches to goto statement, the control of the program will jump to the `label:` and executes the code below it.



Example 1:

```
//Write a C Program Which Print 1 To 10 Number Using goto statement.
#include<stdio.h>
#include<conio.h>
void main()
{
    int i=1;
    clrscr();
    count:                  //This is Label

    printf("%d\n",i);
    i++;
    if(i<=10) {
        goto count;     //This jumps to label "count:"
    }
    getch();
}
```

Download

Output:

Command Prompt

```
1
2
3
4
5
6
7
8
9
10
```

Example 2

```
#include <stdio.h>
int main()
{
  int num,i=1;
  printf("Enter the number whose table you want to print?");
  scanf("%d",&num);
  table:
  printf("%d x %d = %d\n",num,i,num*i);
  i++;
  if(i<=10)
  goto table;
}
```

Output:

```
Enter the number whose table you want to print?10
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
```

When should we use goto?

The only condition in which using goto is preferable is when we need to break the multiple loops using a single statement at the same time. Consider the following example.

```c
#include <stdio.h>
int main()
{
  int i, j, k;
  for(i=0;i<10;i++)
  {
    for(j=0;j<5;j++)
    {
      for(k=0;k<3;k++)
      {
        printf("%d %d %d\n",i,j,k);
        if(j == 3)
        {
          goto out;
        }
      }
    }
  }
  out:
    printf("came out of the loop");
}
```

```
0 0 0
0 0 1
0 0 2
0 1 0
0 1 1
0 1 2
0 2 0
0 2 1
0 2 2
0 3 0
came out of the loop
```

Note:

Though goto statement is included in ANSI standard of C, use of goto statement should be reduced as much as possible in a program.

Reasons to avoid goto statement

- Though, using goto statement give power to jump to any part of program, using goto statement makes the logic of the program complex and tangled.
- In modern programming, goto statement is considered a harmful construct and a bad programming practice.
- The goto statement can be replaced in most of C program with the use of break and continue statements.
- In fact, any program in C programming can be perfectly written without the use of goto statement.
- All programmers should try to avoid goto statement as possible as they can.