JUMP Statements in C

- jump statements are used to jump from one part of the code to another altering the normal flow of the program. They are used to transfer the program control to somewhere else in the program.

Types of Jump Statements in C

There are 4 types of jump statements in C:

1. break
2. continue
3. goto
4. return

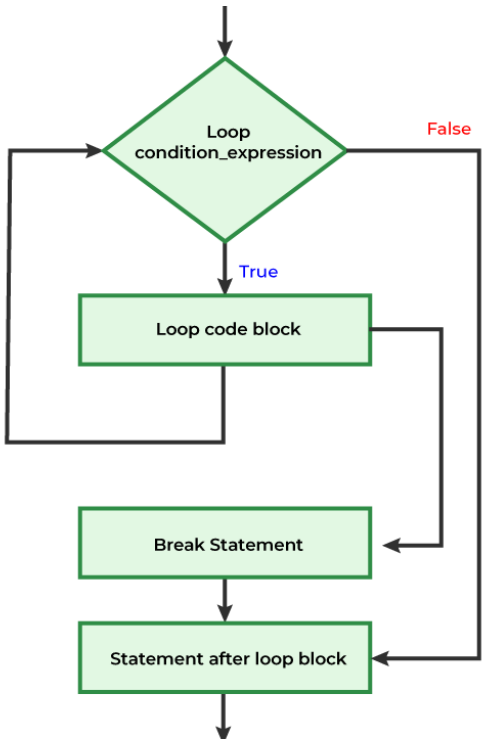
1. break in C

- The break statement exits or terminates the loop or switch statement based on a certain condition, without executing the remaining code.

Syntax of break in C

```
break;
```

Flowchart of break Statement



Uses of break in C

- The break statement is used
  - To come out of the loop.
  - To come out from the nested loops.
  - To come out of the switch case.
- The break statement is also used inside the switch statement to terminate the switch statement after the matching case is executed.

Example: C program to illustrate the break in c loop

```
#include <stdio.h>

int main()

{

        int i;

        for (i = 1; i <= 10; i++) {

                // when i = 6, the loop should end

                if (i == 6) {

                        break;

                }

                printf("%d ", i);

        }

        printf("Loop exited.\n");

        return 0;

}
```

Output

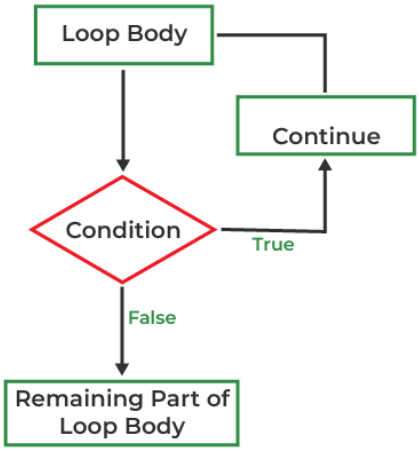
1 2 3 4 5 Loop exited.

2. Continue in C

- The continue statement in C is used to skip the remaining code after the continue statement within a loop and jump to the next iteration of the loop.
- When the continue statement is encountered, the loop control immediately jumps to the next iteration, by skipping the lines of code written after it within the loop body.

Syntax of continue in C

```
continue;
```



Example: C Program to illustrate the continue statement

```
#include <stdio.h>
int main()
{
        int i;
        for (i = 0; i < 5; i++) {
                if (i == 2) {
                        // continue to be executed if i = 2
                        printf("Skipping iteration %d\n", i);
                        continue;               }
                printf("Executing iteration %d\n", i);
        }
        return 0;
}
```

Output:

```
Executing iteration 0
Executing iteration 1
Skipping iteration 2
Executing iteration 3
Executing iteration 4
```

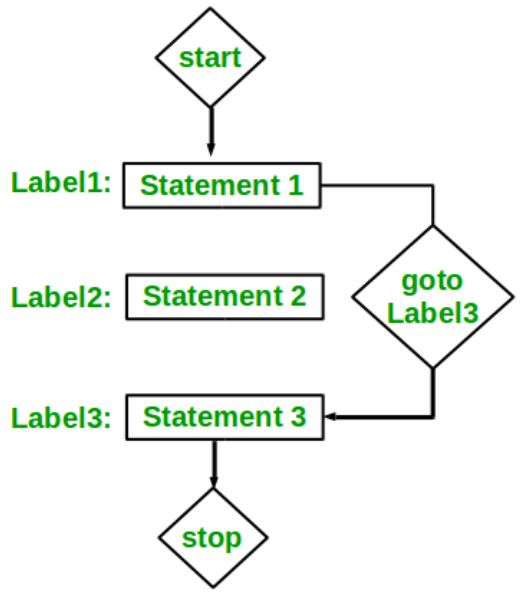
3. Goto Statement in C

- The goto statement is used to jump to a specific point from anywhere in a function.
- It is used to transfer the program control to a labeled statement within the same function.

Syntax of goto Statement

```
goto label;
.
.
label:
//code
```

Flowchart of goto Statement



```
// C program to check if a number is even or not using goto statement
#include <stdio.h>
void checkEvenOrNot(int num)
{
        if (num % 2 == 0)
                // jump to even
                goto even;
        else
                // jump to odd
                goto odd;
even:
        printf("%d is even", num);
        // return if even
        return;
odd:
        printf("%d is odd", num);
}
int main()
{
        int num = 26;
        checkEvenOrNot(num);
        return 0;
}
```

Output

26 is even

4. Return Statement in C

- The return statement in C is used to terminate the execution of a function and return a value to the caller.
- It is commonly used to provide a result back to the calling code.

Syntax:

```
return expression;
```

```
#include <stdio.h>
int add(int a, int b)
{
        int sum = a + b;
        return sum; // Return the sum as the result of the
                                // function
}
void printMessage()
{
        printf("GeeksforGeeks\n");
        return; // Return from the function with no value (void)
}
int main()
{
        int result = add(5, 3);
        printf("Result: %d\n", result);
        printMessage();
        return 0;
}
```

Output

```
Result: 8
GeeksforGeeks
```