Lecture 2: C++ Jump Statement

C++ Control Statement

There are three types of control statements in C++:

Decision Making Statements (if-else, switch statements)

Iterative Statements (for, while, and do-while loops)

Jump Statements

Jump Statement:

Jump statement is used to terminating or continues the loop inside a program or to stop the execution of a function.

Jump statements in C++ mainly have four types:

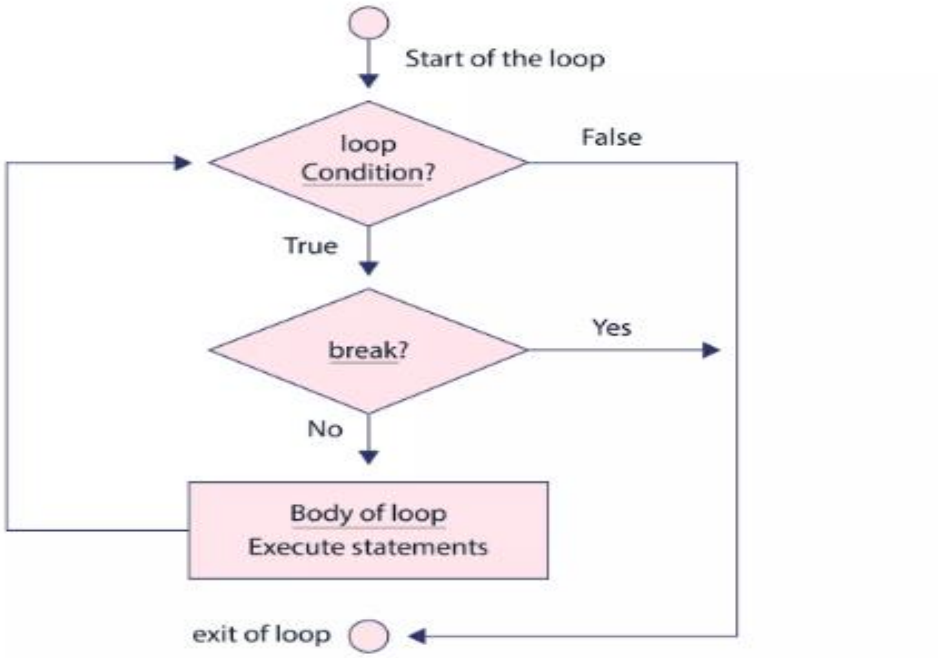
- Break statements.
- Continue statements.
- Goto statements
- Return statements.

Break Statement:

The break statements, used within loops or iterative statements. The objective of the break statement is to stop the execution of the loop and transfer the program control to the statement just after the loop.

In C++, the break statement terminates the loop when it is encountered.

flow chart of the break statement



Syntax of Break :

```
break;
```

```
for (init; condition; update) {
    // code
    if (condition to break) {
        break;
    }
    // code
}

---------------------------------------

while (condition) {
    // code
    if (condition to break) {
        break;
    }
    // code
}
```

Example 1: break with for loop:

```
// program to print the value of i
#include <iostream>
using namespace std;
int main() {
 for (int i = 1; i <= 5; i++) {
 // break condition
 if (i == 3) {
 break;
 }
 cout << i << endl;
 }
return 0;
}
```

Output:

```
1
2
```

In the above program, the for loop is used to print the value of i in each iteration. Here, notice the code:

```
if (i == 3) {
    break;
}
```

Note: The break statement is usually used with decision-making statements.

Example 2: break with while loop:

```
// program to find the sum of positive numbers
// if the user enters a negative numbers, break ends the loop
// the negative number entered is not added to sum
#include <iostream>
using namespace std;
int main() {
 int number;
 int sum = 0;
 while (true) {
 // take input from the user
 cout << "Enter a number: ";
 cin >> number;
 // break condition
 if (number < 0) {
 break;
 }
 // add all positive numbers
 sum += number;
 }
 // display the sum
 cout << "The sum is " << sum << endl;
 return 0;
}
```

Output:

```
Enter a number: 2
Enter a number: 2
Enter a number: 2
Enter a number: -2
The sum is 6
```

In the above program, the user enters a number. The while loop is used to print the total sum of numbers entered by the user.

Here, notice the code.

```
if (number < 0) {
 break;
 }
```

This means, when the user enters a negative number, the break statement terminates the loop and codes outside the loop are executed.

The while loop continues until the user enters a negative number.