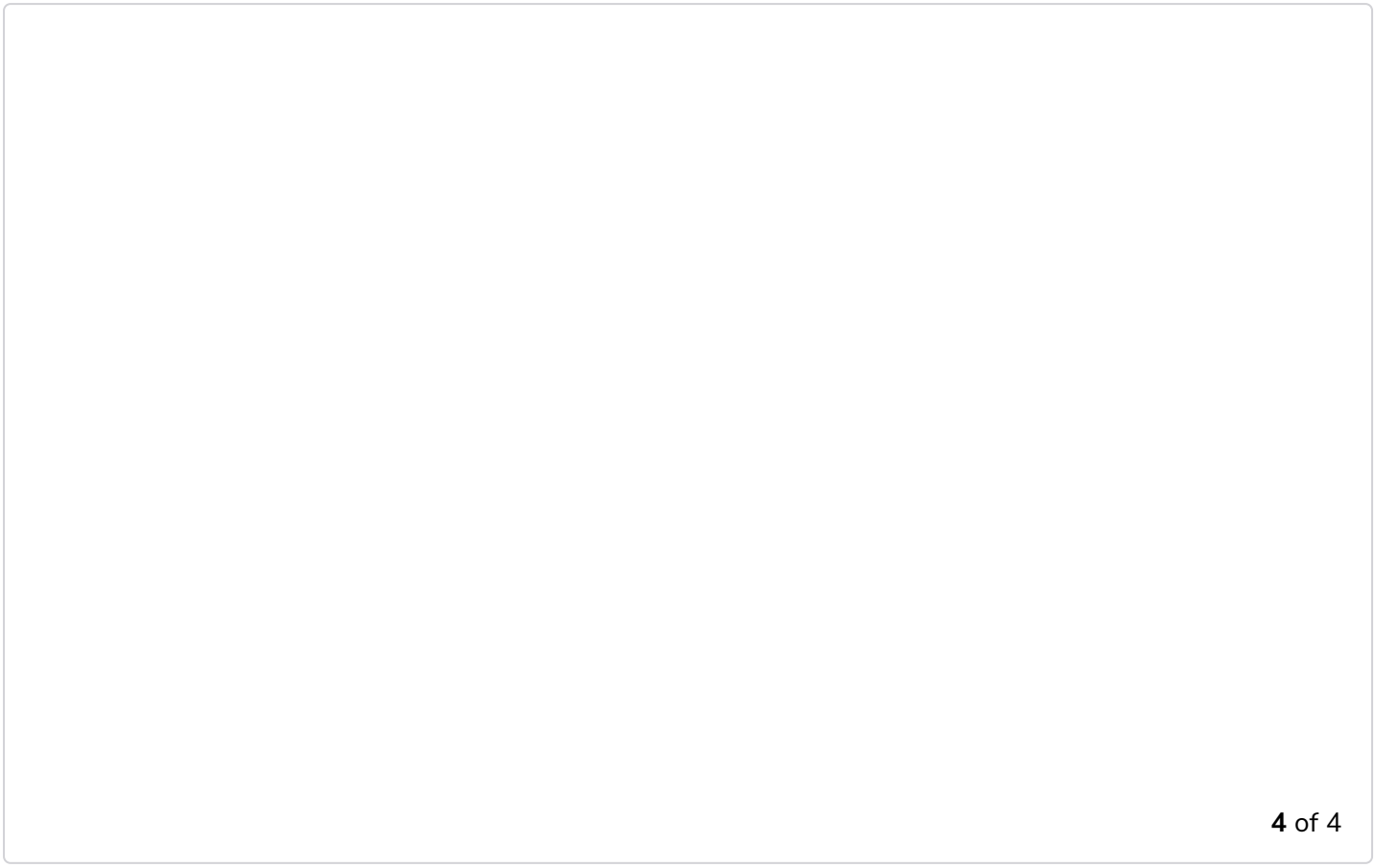
break Statement

In this lesson, you will be introduced to the break statement in C++.

We'll cover the following

Introduction

Suppose you have a coupon to buy five ice-creams free of cost. But the ice-cream man only has three ice-creams. In this case, you can have free ice-creams, but the ice-cream man runs out of ice-creams.



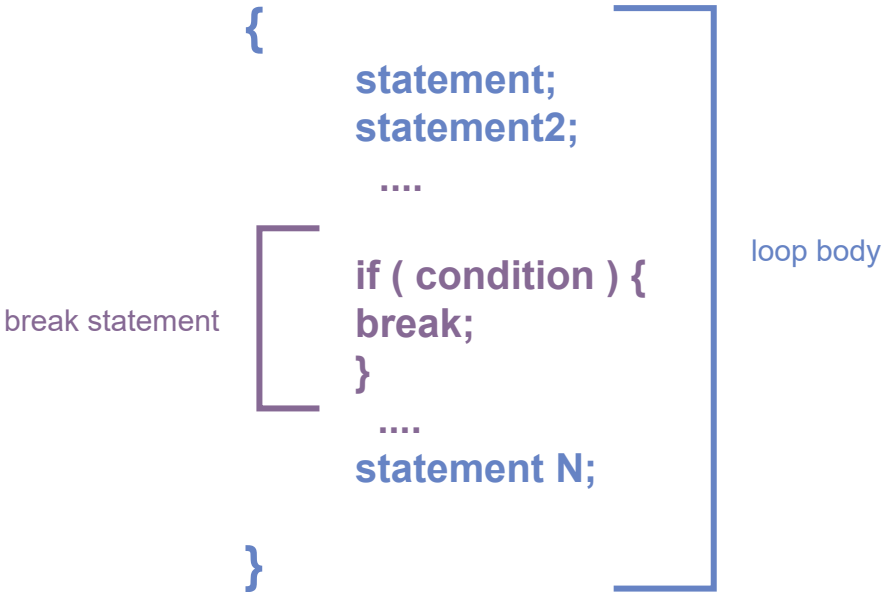
In the era of programming, we can use the `break` statement for such situations. The `break` statement can be used to jump out of the loop immediately when a particular condition evaluates to true.

---

*The **break statement** terminates the loop and transfers the control to the very next statement after the loop body.*

Use case

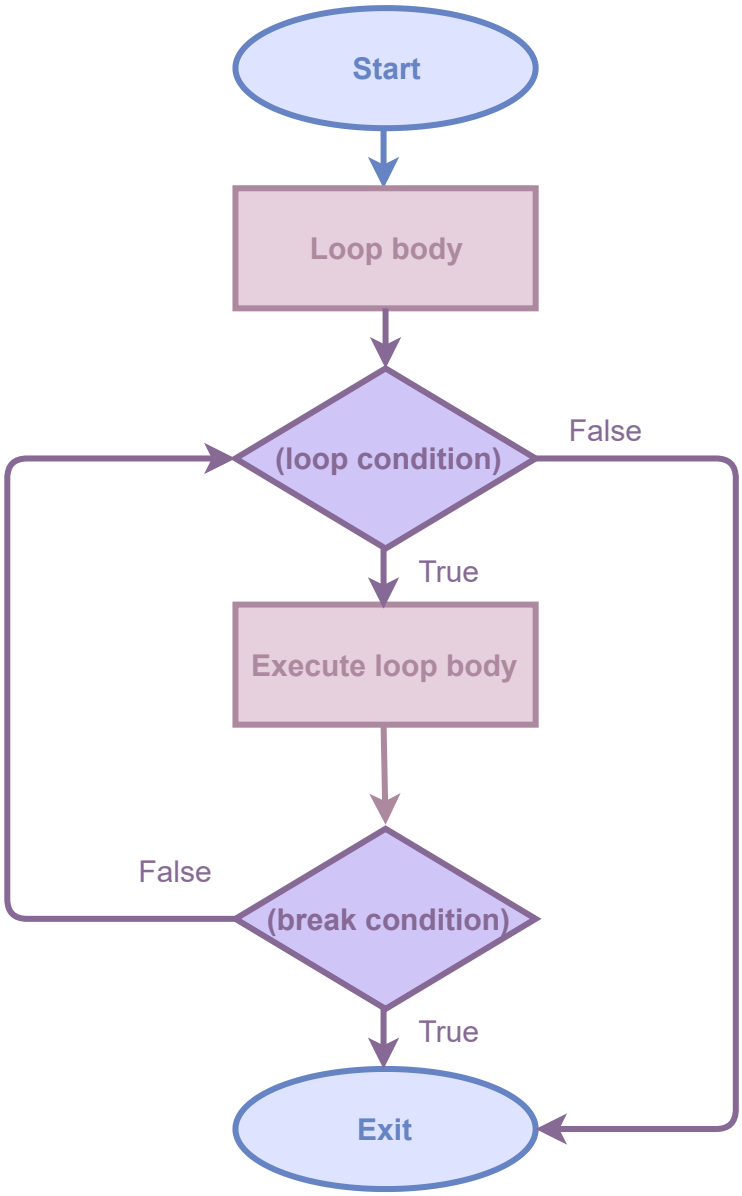
Let's go over a use case of the `break` statement. It is so simple to use. You just have to write a `break` after the line you want to terminate the loop!



The basic syntax of a `break` statement consists of an `if` keyword followed by a condition in round brackets. The curly brackets contain a `break` keyword that terminates the loop when the condition evaluates to true.

Flowchart

Let's look at the flowchart of the above example of a `break` statement.



- The loop first evaluates its continuation condition.
- If the condition evaluates to `true`, it executes the code inside the loop body. If not, it skips the loop body.
- Inside the loop body, we have the `if` condition followed by a `break` statement.
- If the `break` condition evaluates to `true`, it exits the loop body. If not, it checks the loop condition again.

Example program

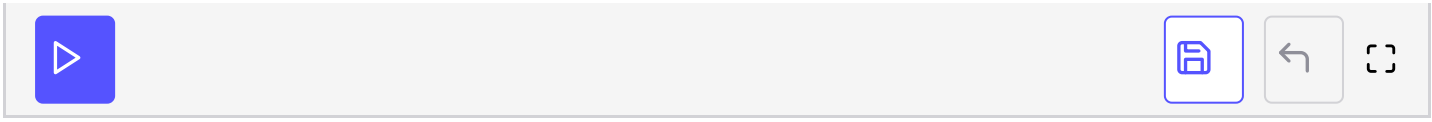
Let's translate the example given above into a C++ program.

Press the **RUN** button and see the output!

```
#include <iostream>
```

```
using namespace std;

int main() {
  // Initialize variable icecream
  int icecream;
  // for loop start
  for (icecream = 5; icecream > 0; icecream--) {
    // loop body
    cout << "Number of free ice-creams = " << icecream << endl;
    // break statement
    if (icecream == 2) {
      break;
    }
    cout << "Buy an icecream" << endl;
  }
  // Exit loop
  cout << "Sorry! We ran out of ice-cream" << endl;
  return 0;
}
```



Explanation

In the code above, we have a `for` loop that iterates from `5` to `1`. However, since we have a `break` statement that is executed when the value of the loop variable is `2`, the loop terminates, and it transfers the control to the very next statement after the loop body.

Line No. 7: Declares a variable `icecream`

Line No. 9:

- `icecream = 5:` The initial value of `icecream` is set to `5`.
- `icecream > 0:` When the loop condition evaluates to true, it executes the statements from **Lines No. 11 to 17**.
- `icecream--:` After executing the loop block, it jumps back to **Line No. 9**, where it decrements the value of `icecream` by `1` and evaluates the condition again.

Line No. 11: Prints the value of the `ice-cream` to the console

Line No. 13: Checks if the value of the `ice-cream` is `2`. If yes, then execute **Line No. 14** to **Line No. 15**. If no, then jump to **Line No. 16**.

Line No. 14: Breaks the loop. When the break statement is executed, the program will exit the loop body and jump to **Line No. 19**.

Line No. 16: Prints `Buy an icecream` to the console

Line No. 19: Prints `Sorry! We ran out of ice-cream` to the console

If `number = 1`, then what is the output of the following code?

```
int number;
  for (number ; number < 4; number++) {
    cout << number << endl;
    if (number == 3) {
      break;
    }
  }
```

Retake Quiz

Interesting so far? Let's discuss the `continue` statement in the upcoming lesson.

Stay tuned!