**Break and Continue statement in C**

Tthe **break** and **continue** statements in [C](https://www.geeksforgeeks.org/c-programming-language/). They are the same type of statements which is used to alter the flow of a program still they have some difference between them.

**break statement:** This statement terminates the smallest enclosing loop (i.e., **while**, [do-while](https://www.geeksforgeeks.org/c-c-do-while-loop-with-examples/), [for loop](https://www.geeksforgeeks.org/range-based-loop-c/), or [switch statement](https://www.geeksforgeeks.org/switch-statement-cc/)). Below is the program to illustrate the same:

|  |
| --- |
| // C program to illustrate the  // break statement  #include <stdio.h>    // Driver Code  **int** main()  {    **int** i = 0, j = 0;        // Iterate a loop over the      // range [0, 5]  **for** (**int** i = 0; i < 5; i++) {    **printf**("i = %d, j = ", i);            // Iterate a loop over the          // range [0, 5]  **for** (**int** j = 0; j < 5; j++) {                // Break Statement  **if** (j == 2)  **break**;    **printf**("%d ", j);          }    **printf**("\n");      }    **return** 0;  } |

**Output:**

i = 0, j = 0 1

i = 1, j = 0 1

i = 2, j = 0 1

i = 3, j = 0 1

i = 4, j = 0 1

**Explanation:** In the above program the inner for loop always ends when the value of the variable **j** becomes **2**.

**continue statement:** This statement skips the rest of the loop statement and starts the next iteration of the loop to take place. Below is the program to illustrate the same.

|  |
| --- |
| // C program to illustrate the  // continue statement  #include <stdio.h>    // Driver Code  **int** main()  {    **int** i = 0, j = 0;        // Iterate a loop over the      // range [0, 5]  **for** (**int** i = 0; i < 5; i++) {    **printf**("i = %d, j = ", i);            // Iterate a loop over the          // range [0, 5]  **for** (**int** j = 0; j < 5; j++) {                // Continue Statement  **if** (j == 2)  **continue**;    **printf**("%d ", j);          }    **printf**("\n");      }    **return** 0;  } |

**Output:**

i = 0, j = 0 1 3 4

i = 1, j = 0 1 3 4

i = 2, j = 0 1 3 4

i = 3, j = 0 1 3 4

i = 4, j = 0 1 3 4

**Explanation:** In the above program the inner for loop always skip the iteration when the value of the variable **j** becomes **2**.

**Tabular Difference Between the break and continue statement:**

| **Break Statement** | **Continue Statement** |
| --- | --- |
| The Break statement is used to exit from the loop constructs. | The continue statement is not used to exit from the loop constructs. |
| The break statement is usually used with the switch statement, and it can also use it within the while loop, do-while loop, or the for-loop. | The continue statement is not used with the switch statement, but it can be used within the while loop, do-while loop, or for-loop. |
| When a break statement is encountered then the control is exited from the loop construct immediately. | When the continue statement is encountered then the control automatically passed from the beginning of the loop statement. |
| **Syntax:** break; | **Syntax:** continue; |
| Break statements uses switch and label statements. | It does not use switch and label statements. |
| Leftover iterations are not executed after the break statement. | Leftover iterations can be executed even if the continue keyword appears in a loop. |