Loop Control Statement

- A loop is a sequence of instructions that is continually repeated until a certain condition is reached.
- Loop control statements change execution from their normal sequence.
- When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

Sr.No.	Name of the control statement	Description
1	**Break statement**	This command terminates the loop's execution and transfers the program's control to the statement next to the loop.
2	**Continue statement**	This command skips the current iteration of the loop. The statements following the continue statement are not executed once the Python interpreter reaches the continue statement.
3	**Pass statement**	The pass statement is used when a statement is syntactically necessary, but no code is to be executed.

❖ Continue Statement

The continue statement in Python returns the control to the beginning of the loop.

Example:

```
#print all letter except ‘e’ and ‘p’
for letter in ‘ToppperWorld’:
    If letter == ‘e’ or letter == ’p’:
        Continue
print(‘Current Letter:’, letter)
```

Output:

```
Current Letter : T
Current Letter : o
Current Letter : r
Current Letter : W
Current Letter : o
Current Letter : r
Current Letter : l
Current Letter : d
```

❖ Break Statement

The break statement in Python brings control out of the loop.

Example:

```
for letter in 'topperworld':
        if letter == 'o' or letter == 'p':
                break
print('Current Letter :', letter)
```

Output:

```
Current Letter : o
```

❖ Pass Statement

We use pass statement in Python to write empty loops. Pass is also used for empty control statements, functions and classes.

Example:

```
for letter in 'TopperWorld':


        # break the loop as soon it sees 'o' or 'p'
        if letter == 'o' or letter == 'p':
                break
print('Current Letter :', letter)
```

Output:

```
Current Letter : o
```