

Skipping loops

The Python **break** keyword can be used to prematurely terminate a loop when a specified condition is met. The **break** statement is situated inside the loop statement block and is preceded by a test expression. When the test returns **True**, the loop ends immediately and the program proceeds on to the next task. For example, in a nested inner loop it proceeds to the next iteration of the outer loop.



nest.py

- 1 Start a new program with a statement creating a loop that iterates three times
`for i in range(1, 4) :`
- 2 Next, add an indented statement creating a “nested” inner loop that also iterates three times
`for j in range(1, 4) :`
- 3 Now, add a further-indented statement in the inner loop to display the counter numbers (of both the outer loop and the inner loop) on each iteration of the inner loop
`print('Running i=' , i , ' j=' , j)`
- 4 Save then run this program – to see the counter values on each loop iteration



Compare these nested **for** loops with the nested **while** loops example on page 67.

```
Python Shell
File Edit Shell Debug Options Windows Help
>>> ===== RESTART =====
>>>
Running i = 1 j = 1
Running i = 1 j = 2
Running i = 1 j = 3
Running i = 2 j = 1
Running i = 2 j = 2
Running i = 2 j = 3
Running i = 3 j = 1
Running i = 3 j = 2
Running i = 3 j = 3
>>> |
```