CHANDIGARH UNIVERSITY

Python Programming

Loop Control Statements

Outline

- while loop
- range( ) Function
- for Loop
- Nested Loops
- break Statement
- continue Statement

Loop Control Statements

- In general, statements are executed sequentially- The first statement in a function is executed first, followed by the second, and so on. There may be a situation when you need to execute a block of code several number of times.
- A loop statement allows us to execute a statement or group of statements multiple times.
- Python programming language provides the following types of loops to handle looping requirements.
    - while
    - for
    - Nested loops

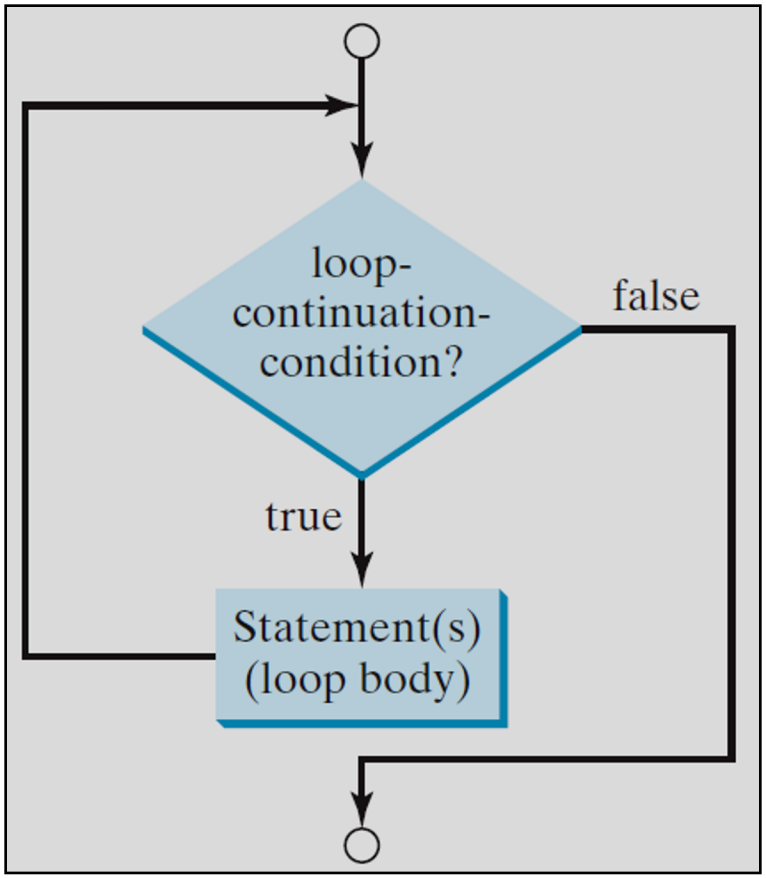
Loop Control Statements

- **while Loop Statements**
- A while loop statement in Python programming language repeatedly executes a target statement as long as a given condition is true.
- It tests the condition before executing the loop body.
- When the condition becomes false, program control passes to the line immediately following the loop.
- In Python, all the statements indented by the same number of character spaces after a programming construct are considered to be part of a single block of code.
- Python uses indentation as its method of grouping statements.

Loop Control Statements

- **while Loop Statements**
- The syntax for the while loop is:

```
while  test-condition:
  # Loop body
  Statement(s)
```



Loop Control Statements

- **while Loop**

while.py - E:/Python3.4/Programs/while.py (3.4.2)

File Edit Format Run Options Windows Help

```
num=eval(input("Enter the number:"))
fact=1
i=1
while i<=num:
    fact=fact*i
    i=i+1
print("Factorial of",num,"is:",fact)
```

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600 64 bit
(AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ========== RESTART ==========
>>>
Enter the number:6
Factorial of 6 is: 720
>>> ========== RESTART ==========
>>>
Enter the number:12
Factorial of 12 is: 479001600
```

Loop Control Statements

- **range( ) function**
- The built-in function range() is the function to iterate over a sequence of numbers.
- It generates an iterator of arithmetic progressions.
- range() generates an iterator to progress integers starting with 0 upto n-1.
- The general form of the range function call is
  - range(begin,end,step)
- Where *begin* is the first value in the range, if omitted, the default value is 0
- *end* is one past the last value in the range, the end value may not be omitted
- *Step* is change in the amount to increment or decrement, if the parameter is omitted, it defaults to 1 (counts up by ones)
- begin, end, and step must all be integer values; floating-point values and other types are not allowed.

Loop Control Statements

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600
64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> range(5)
range(0, 5)
>>> list(range(5))
[0, 1, 2, 3, 4]
>>> list(range(0,5))
[0, 1, 2, 3, 4]
>>> list(range(1,5))
[1, 2, 3, 4]
>>> range(1,10)
range(1, 10)
>>> list(range(1,10))
[1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> list(range(1,10,2))
[1, 3, 5, 7, 9]
>>> list(range(5,0,-1))
[5, 4, 3, 2, 1]
>>> list(range(-4,4))
[-4, -3, -2, -1, 0, 1, 2, 3]
>>> list(range(0,1))
[0]
>>> list(range(1,1))
[]
>>> list(range(0))
[]
```

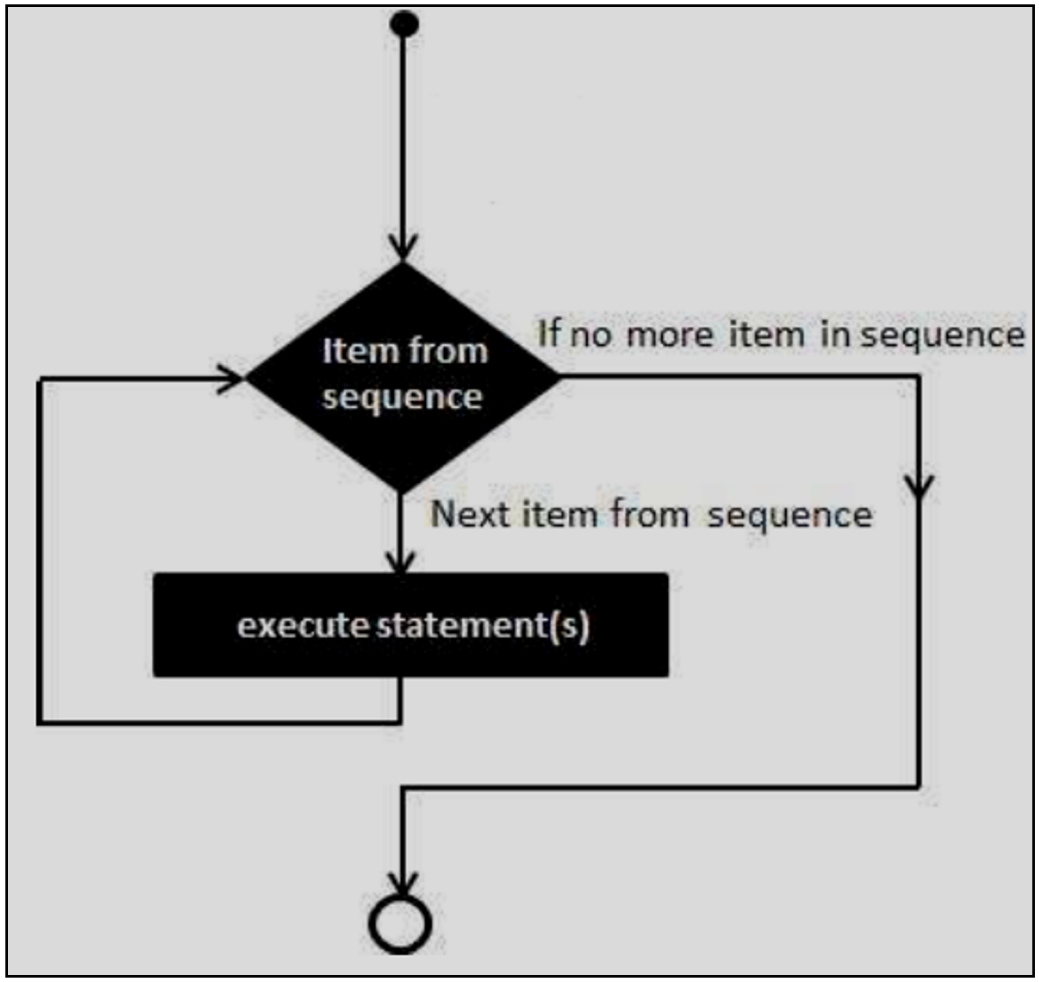
Loop Control Statements

- **for loop**
- The for statement in Python has the ability to iterate over the items of any sequence, such as a list or a string.
- It repeats a set of statements over a group of values.
  - Syntax:

    `for` ***variableName*** `in` ***groupOfValues*** **:**

    statements
    - Indent the statements to be repeated with tabs or spaces.
    - ***variableName*** gives a name to each value, so you can refer to it in the ***statements***.
    - ***groupOfValues*** can be a range of integers, specified with the range function.

Loop Control Statements



Loop Control Statements

- for loop

for.py - E:\Python3.4\Programs\for.py (3.4.2)

File Edit Format Run Options Windows Help

```
num=eval(input("Enter the number:"))
fact=1
i=1
for i in range(1,num+1):
    fact=fact*i
print("Factorial of",num,"is:",fact)
```

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600 64 bit
(AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ========== RESTART ==========
>>>
Enter the number:6
Factorial of 6 is: 720
>>> ========== RESTART ==========
>>>
Enter the number:12
Factorial of 12 is: 479001600
```

Loop Control Statements

- **Nested Loops**
- Python programming language allows the use of one loop inside another loop.

nestedloop1.py - E:/Python3.4/Programs/nestedloop1...

File Edit Format Run Options Windows Help

```
for i in range(1,11):
    for j in range(1,11):
        k=i*j
        print(format(k,"4d"),end=" ")
print()
print("End of the program")
```

```
>>>
   1    2    3    4    5    6    7    8    9   10
   2    4    6    8   10   12   14   16   18   20
   3    6    9   12   15   18   21   24   27   30
   4    8   12   16   20   24   28   32   36   40
   5   10   15   20   25   30   35   40   45   50
   6   12   18   24   30   36   42   48   54   60
   7   14   21   28   35   42   49   56   63   70
   8   16   24   32   40   48   56   64   72   80
   9   18   27   36   45   54   63   72   81   90
  10   20   30   40   50   60   70   80   90  100
End of the program
```

Loop Control Statements

- **Nested Loops**

star.py - E:/Python3.4/Programs/star.py (3.4.2)

File Edit Format Run Options Windows Help

```
i=1
while i < 7:
    j = 0
    while j < i:
        print('*',end='')
        j=j+1
    print()
    i=i+1
print("End of the program")
```

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600 64 bit
(AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
*
**
***
****
*****
******
End of the program
```

Loop Control Statements

- **Nested Loops**

pyramid.py - E:/Python3.4/Programs/pyramid.py (3.4.2)

File Edit Format Run Options Windows Help

```
# Python Program to print pyramid pattern
k = 0
rows = 5
for i in range(1, rows+1):
    for space in range(1, (rows-i)+1):
        print(end="  ")
    while k != (2*i-1):
            print("* ", end="")
            k = k + 1
    k = 0
    print()
```

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600 64 bit
(AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
=
>>>
          *
        * * *
      * * * * *
    * * * * * * *
  * * * * * * * * *
```

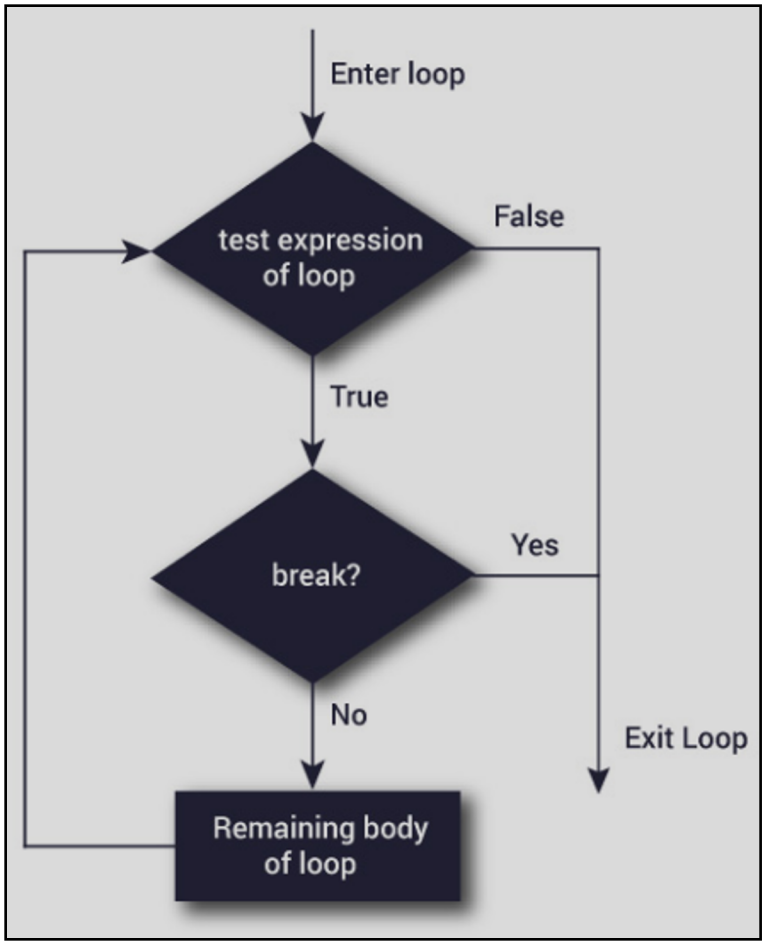
Loop Control Statements

- **break and continue statements**
- In Python, break and continue statements can alter the flow of a normal loop. Loops iterate over a block of code until test expression is false, but sometimes you wish to terminate the current iteration or even the whole loop without checking test expression. The break and continue statements are used in these cases.
- **break statement**
- The break statement terminates the loop containing it.
- Control of the program flows to the statement immediately after the body of the loop.
- If break statement is inside a nested loop (loop inside another loop), break will terminate the innermost loop.

Loop Control Statements

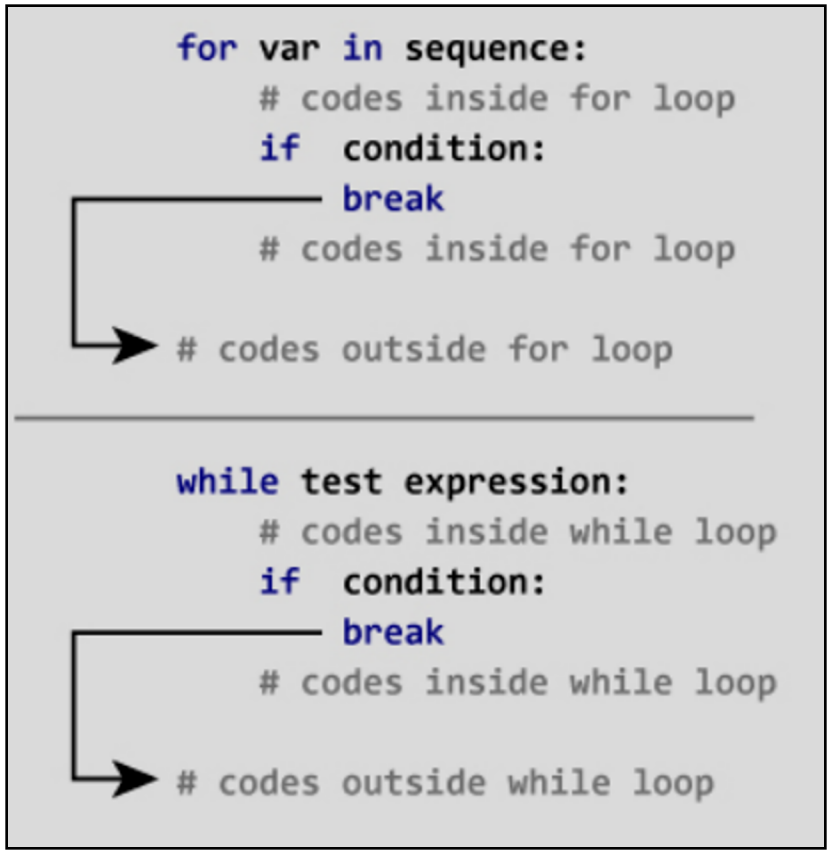
- **break statement**

- Syntax of break
  - break

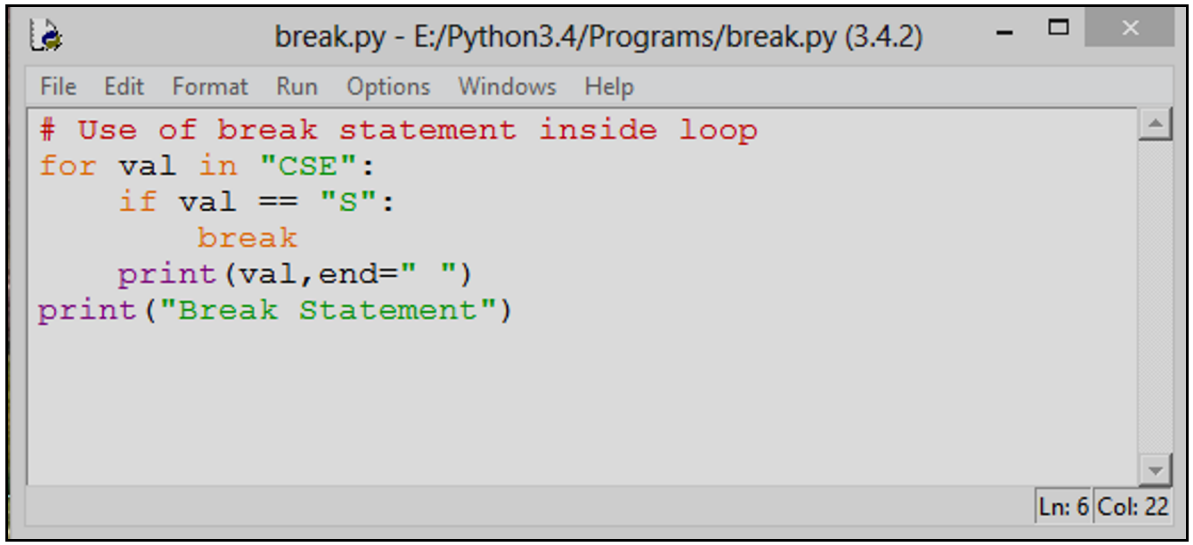


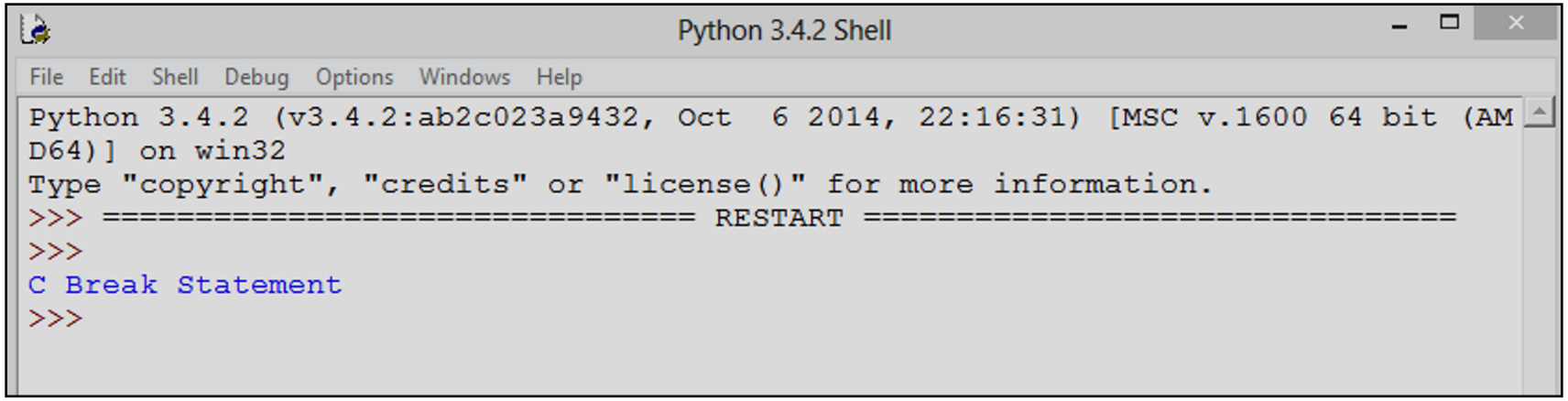
Loop Control Statements

- The working of break statement in for loop and while loop is shown below.



Loop Control Statements





Loop Control Statements

- **continue statement**
- The continue statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.
- The continue statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.
- **Syntax of Continue**
  - continue

Loop Control Statements

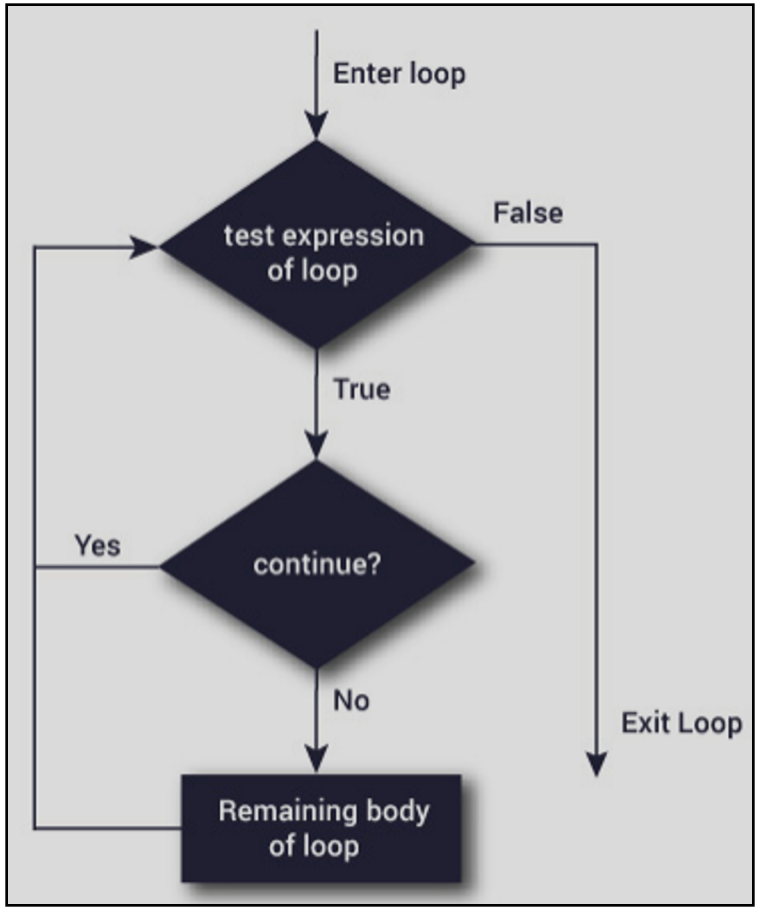
- **continue statement**

```
for var in sequence:
    # codes inside for loop
    if  condition:
        continue
    # codes inside for loop

# codes outside for loop
```

```
while test expression:
    # codes inside while loop
    if  condition:
        continue
    # codes  inside while loop

# codes outside while loop
```



Loop Control Statements

continue.py - E:/Python3.4/Programs/continue.py (3.4.2)

File Edit Format Run Options Windows Help

```
# Use of continue statement inside loop
for val in "CSE":
    if val == "S":
        continue
    print(val,end=" ")
print("Continue Statement")
```

Python 3.4.2 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:16:31) [MSC v.1600
64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
=======
>>>
C E Continue Statement
>>>
```

THANKS.....