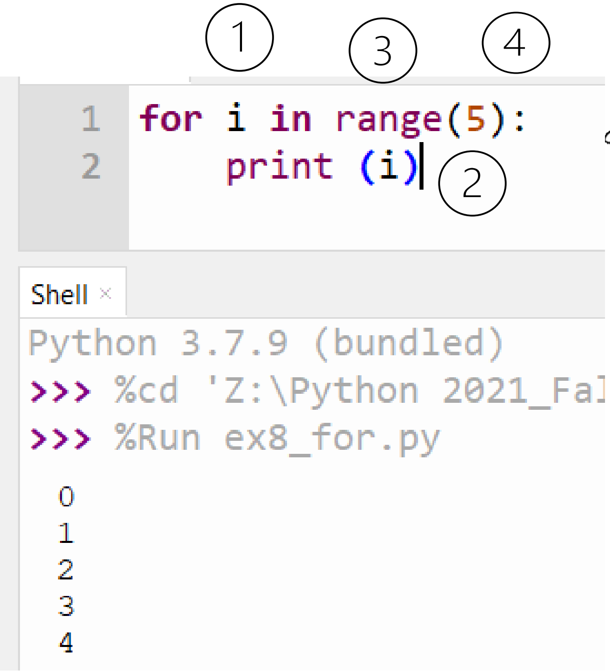
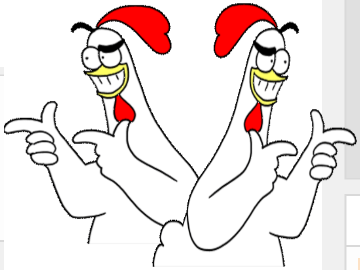
CT60A0203
Introduction to Programming: Python
Week 4A

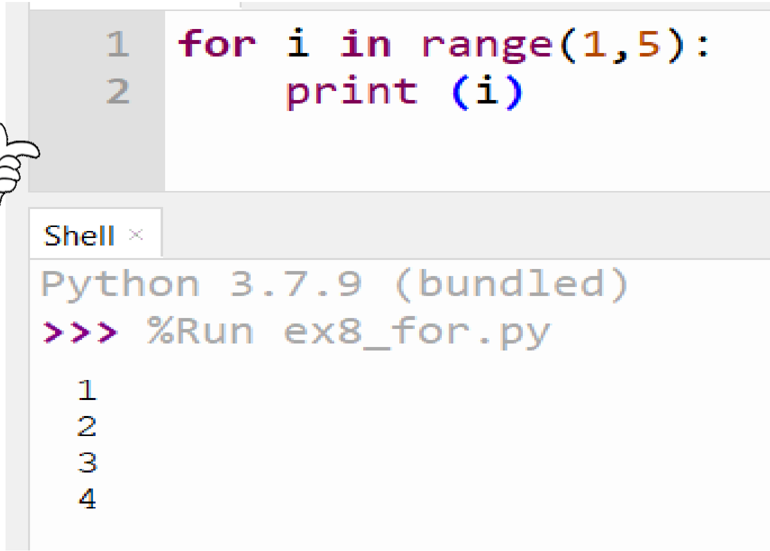
- Another looping statement : for loop in Python



```
for i in range(5):
    print (i)
```

```
Shell ×
Python 3.7.9 (bundled)
>>> %cd 'Z:\Python 2021_Fal
>>> %Run ex8_for.py

  0
  1
  2
  3
  4
```



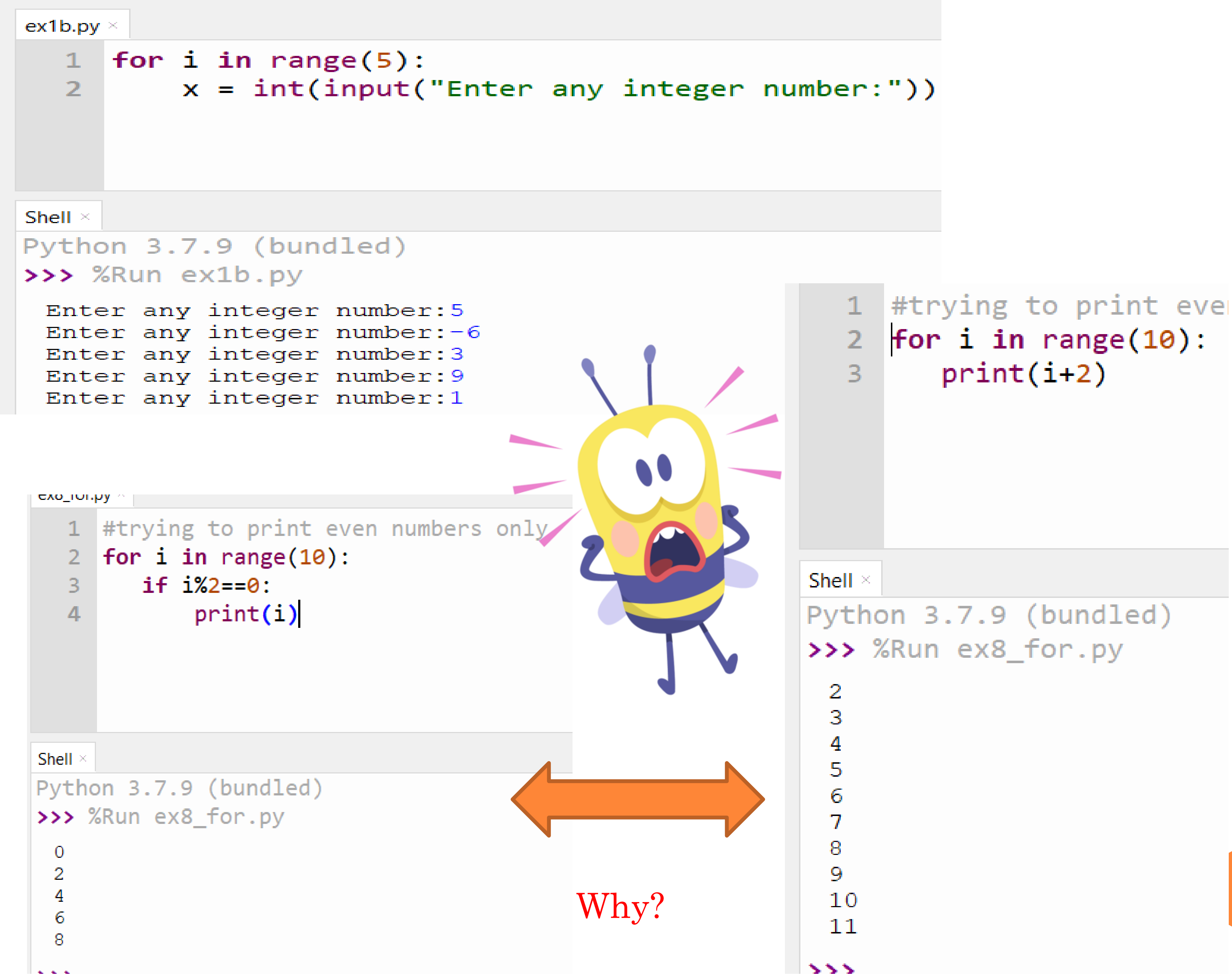


```
for i in range(1,5):
    print (i)
```

```
Shell ×
Python 3.7.9 (bundled)
>>> %Run ex8_for.py

  1
  2
  3
  4
```

How for loop works?

Step 1 – initialization [default value is 0 if not mentioned in the range ()]
Step 2 - executing the block of statement
Step 3 - incrementing the step value (+1)
Step 4 - checking the iteration's end value (<end value)
if the iteration is True, repeat steps 2-4 until loop iteration become False



```
for i in range(5):
    x = int(input("Enter any integer number:"))
```

```
Python 3.7.9 (bundled)
>>> %Run ex1b.py

  Enter any integer number:5
  Enter any integer number:-6
  Enter any integer number:3
  Enter any integer number:9
  Enter any integer number:1
```

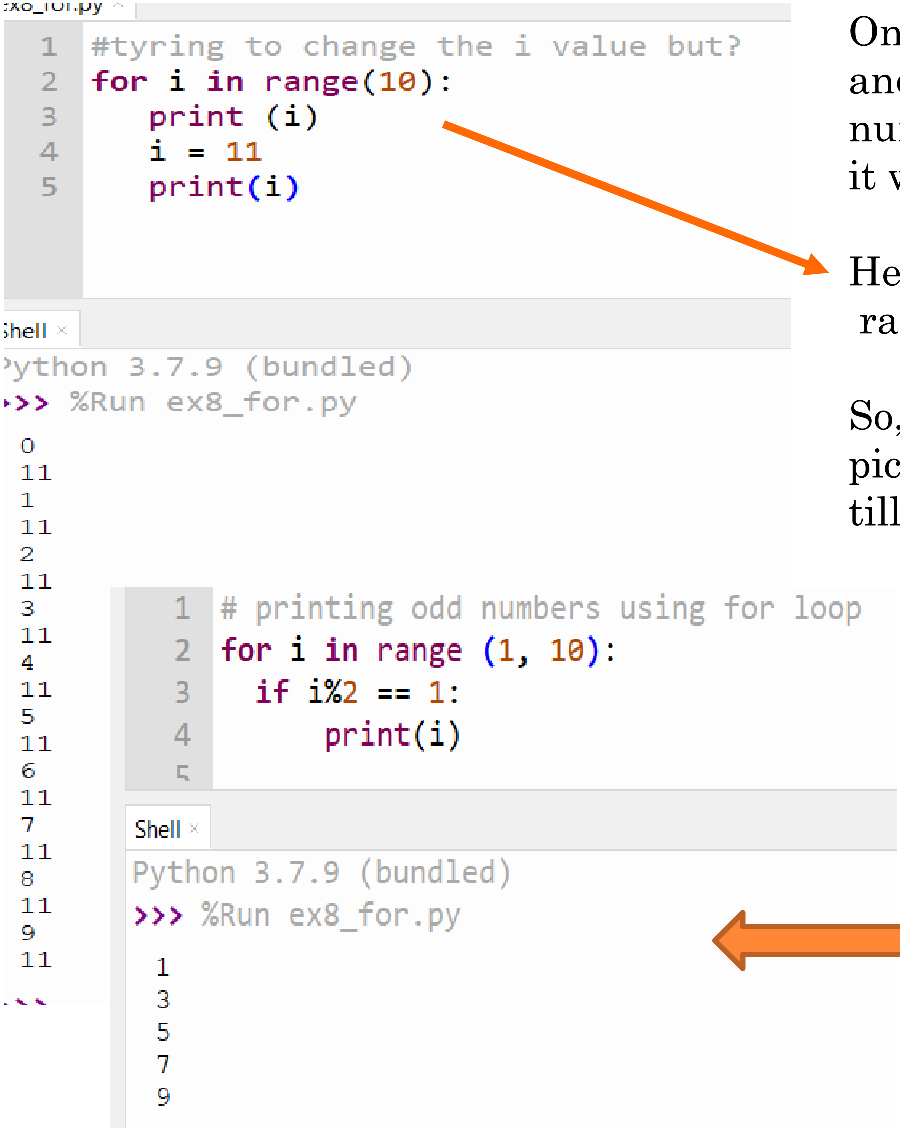
```
#trying to print even numbers only
for i in range(10):
    if i%2==0:
        print(i)
```

```
Python 3.7.9 (bundled)
>>> %Run ex8_for.py

  0
  2
  4
  6
  8

>>>
```

```
#trying to print eve
for i in range(10):
    print(i+2)
```

```
Python 3.7.9 (bundled)
>>> %Run ex8_for.py

  2
  3
  4
  5
  6
  7
  8
  9
  10
  11

>>>
```

Why?



```
#tyring to change the i value but?
for i in range(10):
    print (i)
    i = 11
    print(i)
```

```
ython 3.7.9 (bundled)
>> %Run ex8_for.py

 0
 11
 1
 11
 2
 11
 3
 11
 4
 11
 5
 11
 6
 11
 7
 11
 8
 11
 9
 11
```

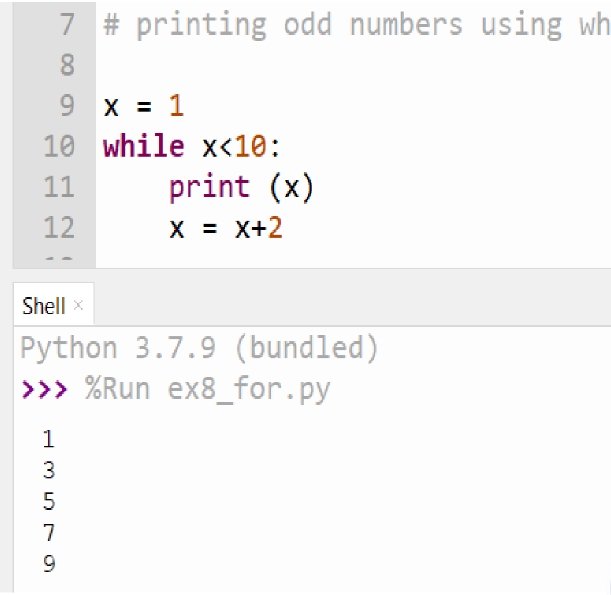
Once the for loop is formed with starting value and end value, it creates a list that contain numbers from starting value to end value. So, it will print those only.

Here i value is 0 and end value is <10. So, range = [0,1,2,3,4,5,6,7,8,9]

So, every time when the loop iterates it will pick the values listed above one after another till end of the predefined range created above.

```
# printing odd numbers using for loop
for i in range (1, 10):
  if i%2 == 1:
      print(i)
```

```
Python 3.7.9 (bundled)
>>> %Run ex8_for.py

 1
 3
 5
 7
 9
```



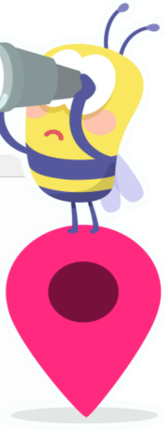
```
# printing odd numbers using while loop

x = 1
while x<10:
    print (x)
    x = x+2
```

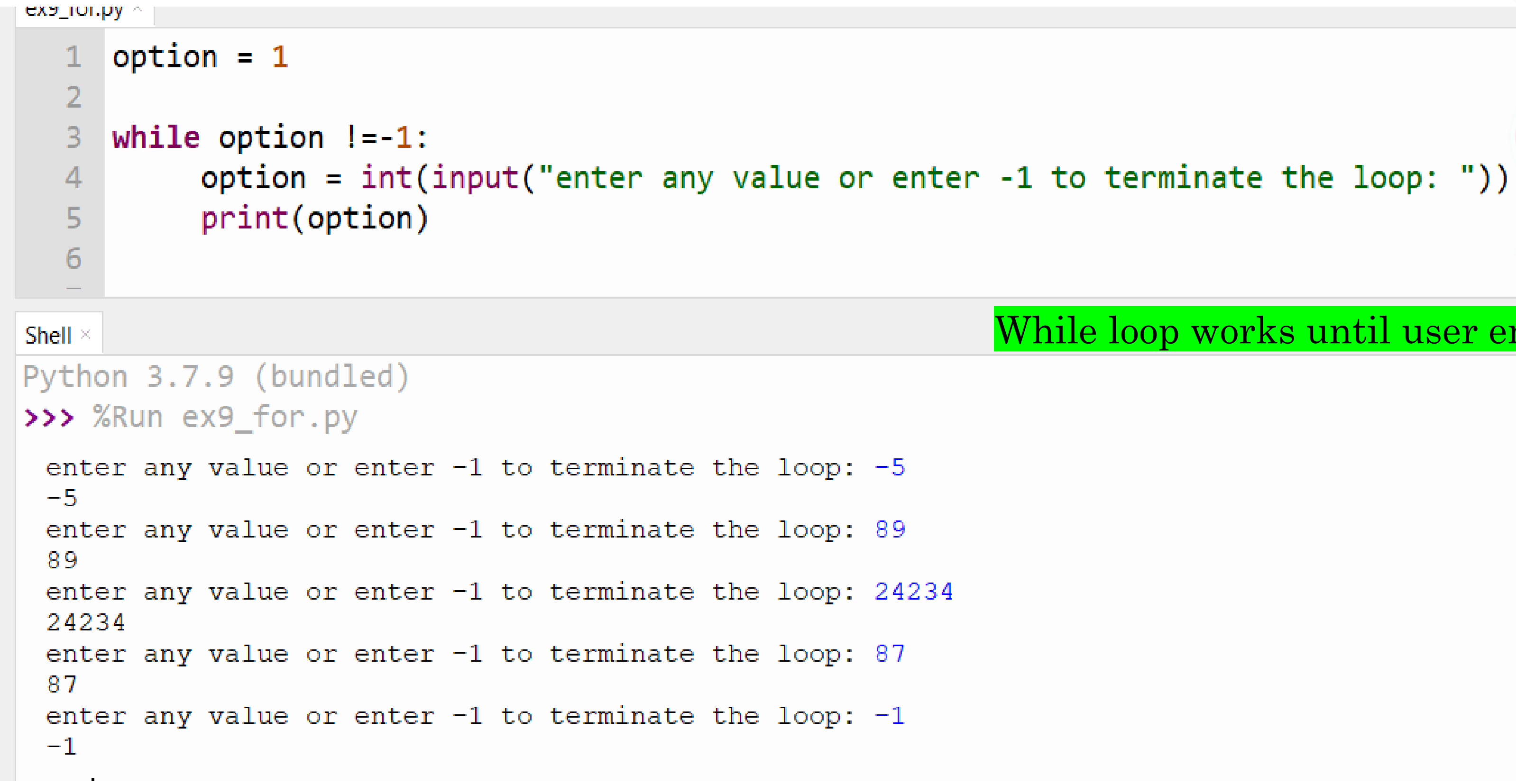
```
Python 3.7.9 (bundled)
>>> %Run ex8_for.py

 1
 3
 5
 7
 9
```

So, for loop meant for iteration to repeat certain process (block of statements) until definite number of times. However, while loop can be used for similar process even the number of iteration is not known.

So, when should I use for loop and while loop?

Use for loop if you want to repeat the process until fixed number of times. Whereas while loop can be used if you want to repeat the process until you get desired results or indefinite number of times and so on.



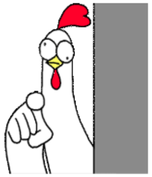


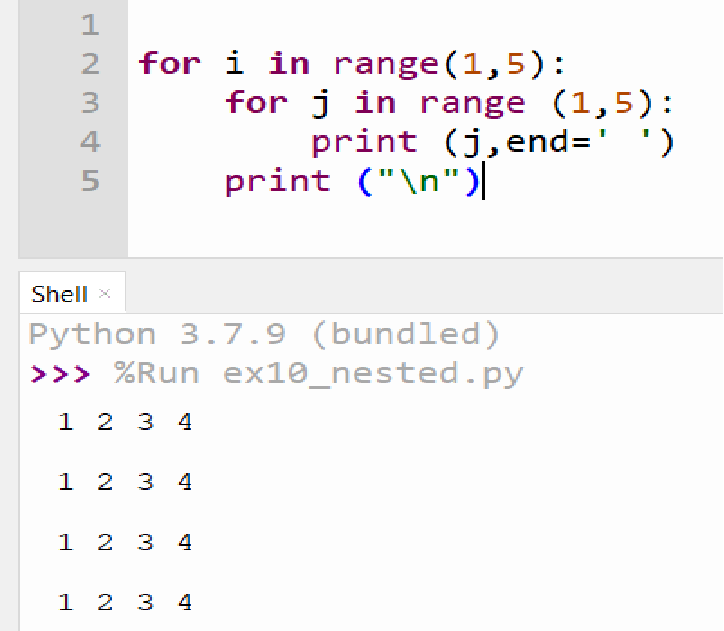
```python
option = 1

while option !=-1:
     option = int(input("enter any value or enter -1 to terminate the loop: "))
     print(option)

```

While loop works until user enters -1

```
Python 3.7.9 (bundled)
>>> %Run ex9_for.py

 enter any value or enter -1 to terminate the loop: -5
 -5
 enter any value or enter -1 to terminate the loop: 89
 89
 enter any value or enter -1 to terminate the loop: 24234
 24234
 enter any value or enter -1 to terminate the loop: 87
 87
 enter any value or enter -1 to terminate the loop: -1
 -1
```

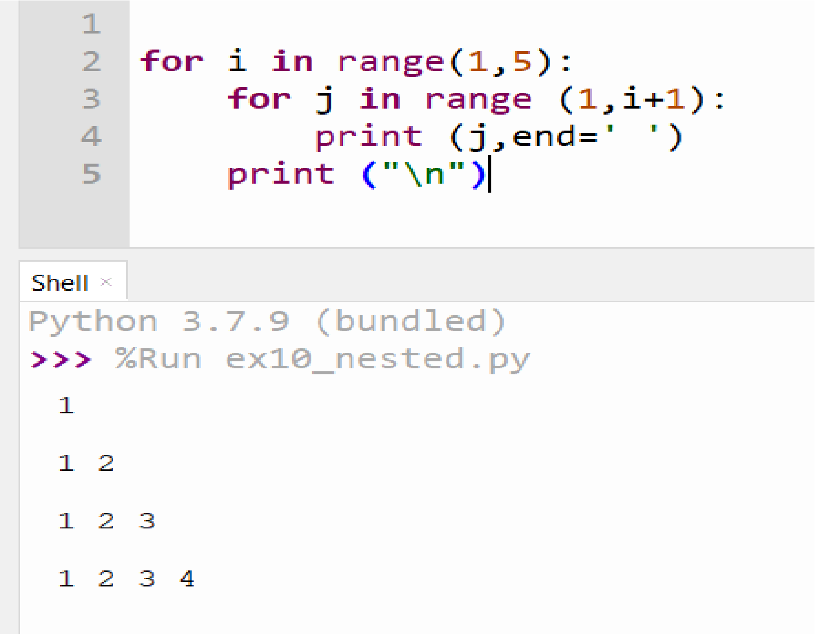
Nested loops: It consists of an outer loop and one or more inner loops. Each time the outer loop is repeated, the inner loops are reentered and started new.





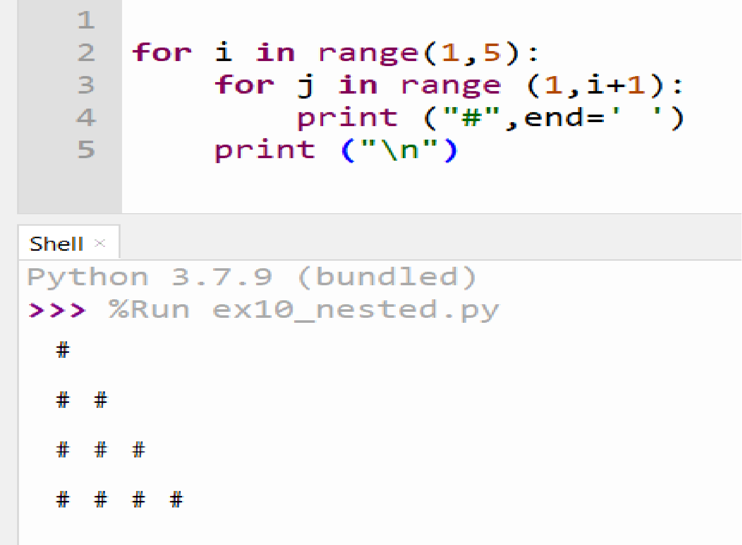
```python
for i in range(1,5):
    for j in range (1,5):
        print (j,end=' ')
    print ("\n")
```

```
Python 3.7.9 (bundled)
>>> %Run ex10_nested.py
 1  2  3  4

 1  2  3  4

 1  2  3  4

 1  2  3  4
```



```python
for i in range(1,5):
    for j in range (1,i+1):
        print (j,end=' ')
    print ("\n")
```

```
Python 3.7.9 (bundled)
>>> %Run ex10_nested.py
 1

 1  2

 1  2  3

 1  2  3  4
```



```python
for i in range(1,5):
    for j in range (1,i+1):
        print ("#",end=' ')
    print ("\n")
```

```
Python 3.7.9 (bundled)
>>> %Run ex10_nested.py
 #

 #  #

 #  #  #

 #  #  #  #
```