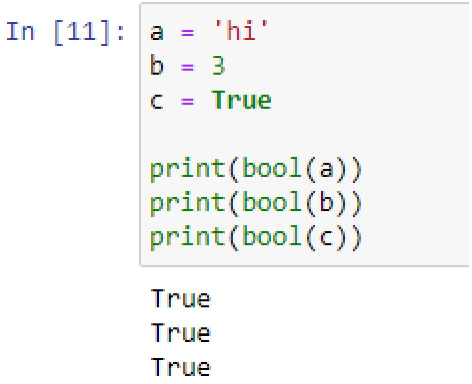
1. What are the two values of the Boolean data type? How do you write them?

The two values of Boolean Data Type are **True** and **False**.

They can be written as :



```
In [11]: a = 'hi'
         b = 3
         c = True

         print(bool(a))
         print(bool(b))
         print(bool(c))

True
True
True
```

2. What are the three different types of Boolean operators?

The three different types of Boolean operators are OR, AND, NOT

3. Make a list of each Boolean operator's truth tables (i.e. every possible combination of Boolean values for the operator and what it evaluate).

Truth Tables for Boolean Operators:

OR

A	**B**	**A or B**
True	True	True
True	False	True
False	True	True
False	False	False

AND

A	**B**	**A and B**
True	True	True
True	False	False
False	True	False
False	False	False

NOT

X	Not X
True	False
False	True

4. What are the values of the following expressions?

(5 > 4) and (3 == 5) - **FALSE**

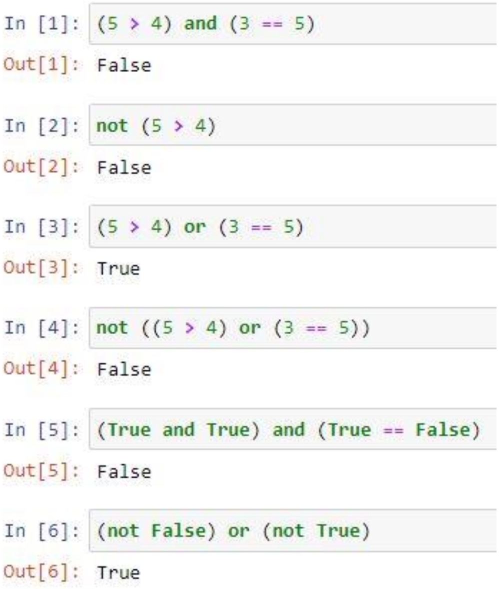
not (5 > 4) - **FALSE**

(5 > 4) or (3 == 5) - **TRUE**

not ((5 > 4) or (3 == 5)) - **FALSE**

(True and True) and (True == False) - **FALSE**

(not False) or (not True) - **TRUE**



```
In [1]: (5 > 4) and (3 == 5)
Out[1]: False

In [2]: not (5 > 4)
Out[2]: False

In [3]: (5 > 4) or (3 == 5)
Out[3]: True

In [4]: not ((5 > 4) or (3 == 5))
Out[4]: False

In [5]: (True and True) and (True == False)
Out[5]: False

In [6]: (not False) or (not True)
Out[6]: True
```

5. What are the six comparison operators?

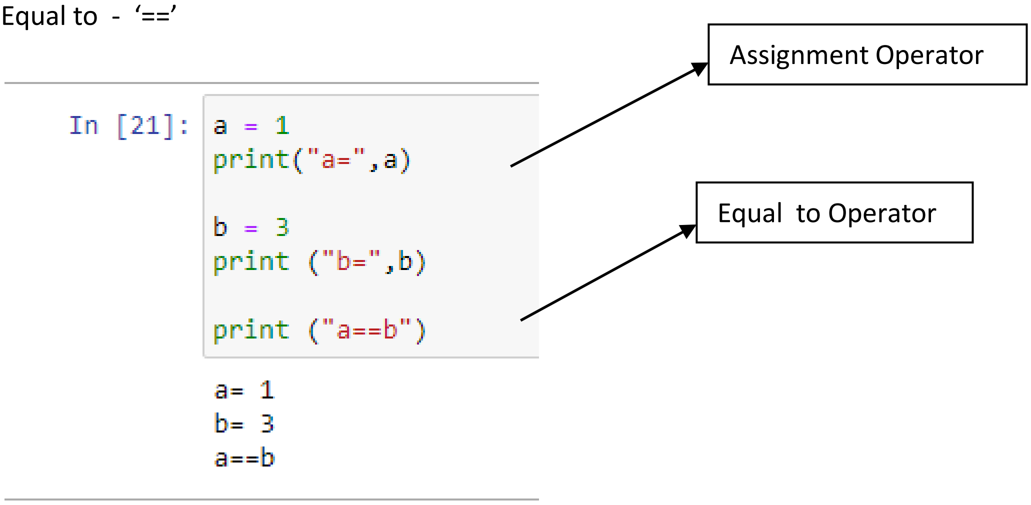
Python has six comparison operators:

- Greater than >
- Less than <
- Equal to ==
- Greater than or equal to <=
- Less than or equal to >=
- Not equal to ! =

6. How do you tell the difference between the equal to and assignment operators? Describe a condition and when you would use one.

Assignment operator - ‘=’

Equal to - ‘==’



```
In [21]: a = 1
         print("a=",a)

         b = 3
         print ("b=",b)

         print ("a==b")

         a= 1
         b= 3
         a==b
```

Assignment Operator

Equal to Operator

7. Identify the three blocks in this code:

spam = 0

if spam == 10:

print('eggs')

if spam > 5:

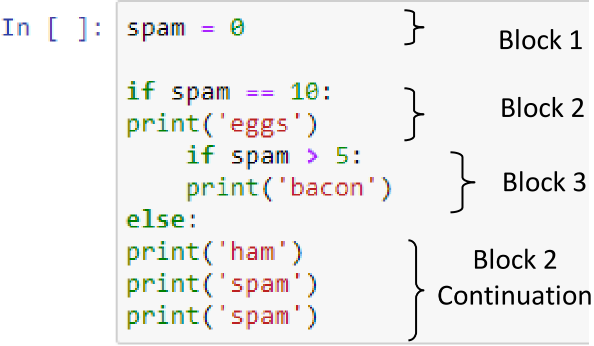
print('bacon')

else:

print('ham')

print('spam')

print('spam')



```
In [ ]: spam = 0

        if spam == 10:
        print('eggs')
            if spam > 5:
            print('bacon')
        else:
        print('ham')
        print('spam')
        print('spam')
```

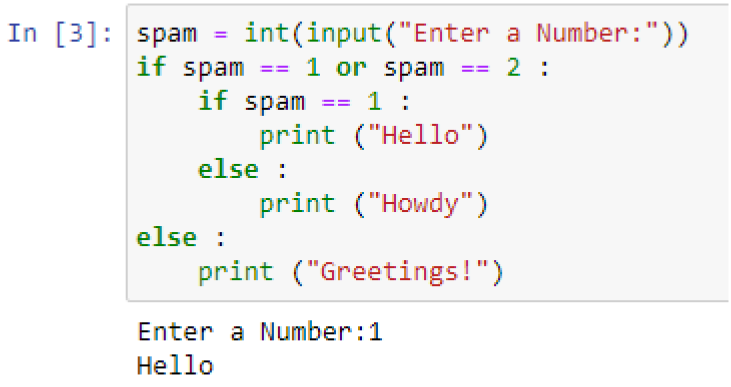
Block 1

Block 2

Block 3

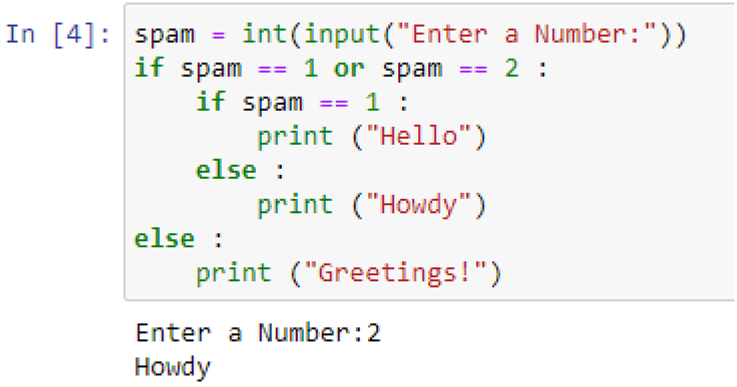
Block 2 Continuation

8. Write code that prints Hello if 1 is stored in spam, prints Howdy if 2 is stored in spam, and prints Greetings! if anything else is stored in spam.



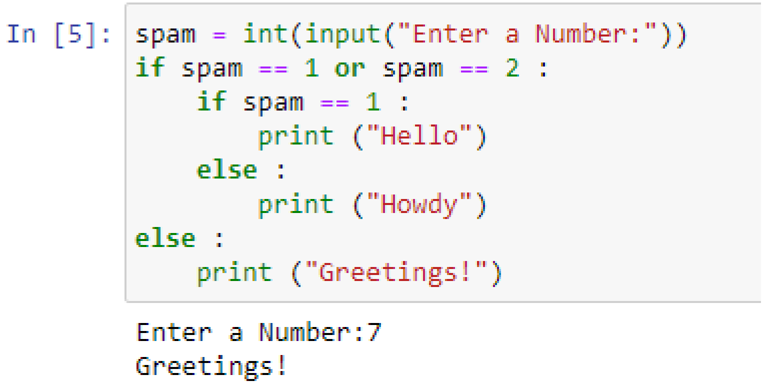
```
In [3]: spam = int(input("Enter a Number:"))
        if spam == 1 or spam == 2 :
            if spam == 1 :
                print ("Hello")
            else :
                print ("Howdy")
        else :
            print ("Greetings!")

        Enter a Number:1
        Hello
```



```
In [4]: spam = int(input("Enter a Number:"))
        if spam == 1 or spam == 2 :
            if spam == 1 :
                print ("Hello")
            else :
                print ("Howdy")
        else :
            print ("Greetings!")

        Enter a Number:2
        Howdy
```



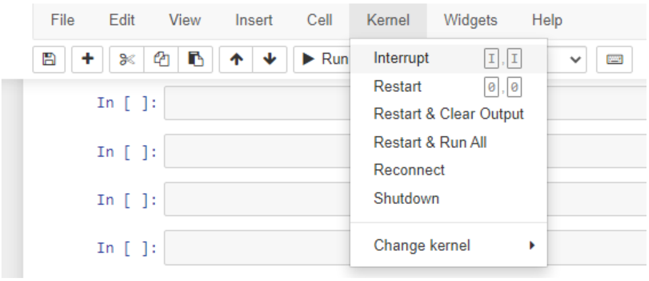
```
In [5]: spam = int(input("Enter a Number:"))
        if spam == 1 or spam == 2 :
            if spam == 1 :
                print ("Hello")
            else :
                print ("Howdy")
        else :
            print ("Greetings!")

        Enter a Number:7
        Greetings!
```

9. If your programme is stuck in an endless loop, what keys you'll press?

To stop an endless loop:

1. Click 'I' letter key two times by using Keyboard.
2. Click 'Interrupt' in Kernel tab of Jupyter notebook by using Mouse.



3. Click Keyboard Interrupt (Ctrl – C) to the Kernel.

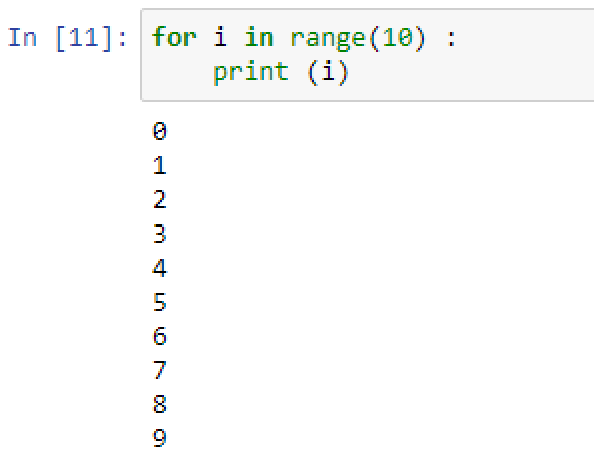
10. How can you tell the difference between break and continue?

Break refers to Stopping the Loop.

Continue refers to Skipping of present Iteration.

11. In a for loop, what is the difference between range(10), range(0, 10), and range(0, 10, 1)?

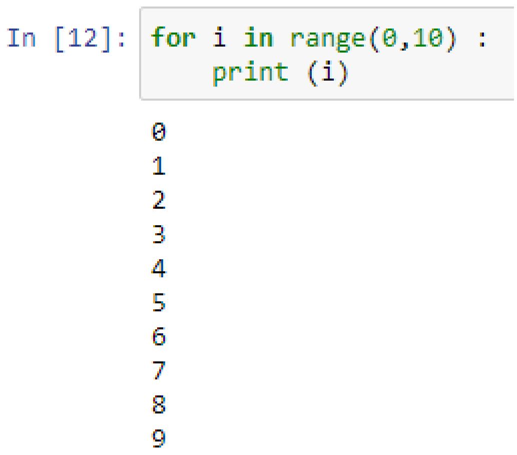
The output is same for all the cases. But there is difference on how they executes:



```
In [11]: for i in range(10) :
             print (i)

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

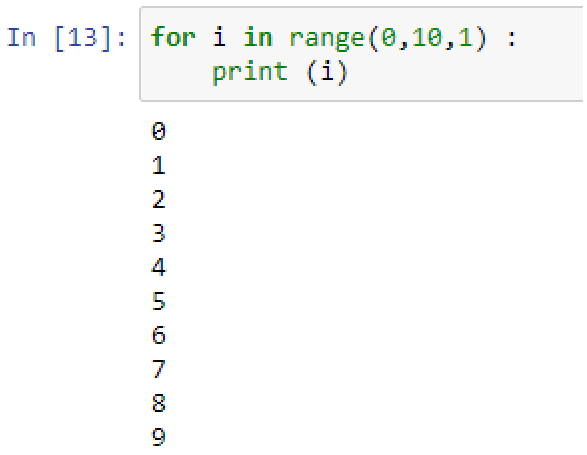
In this Case, Starting point is taken as **Zero** by default. End point is mentioned explicitly. Step size is **One** by default.



```
In [12]: for i in range(0,10) :
             print (i)

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

Secondly, Starting point is given as **Zero**. End point is mentioned explicitly. Step size is **One** by default.

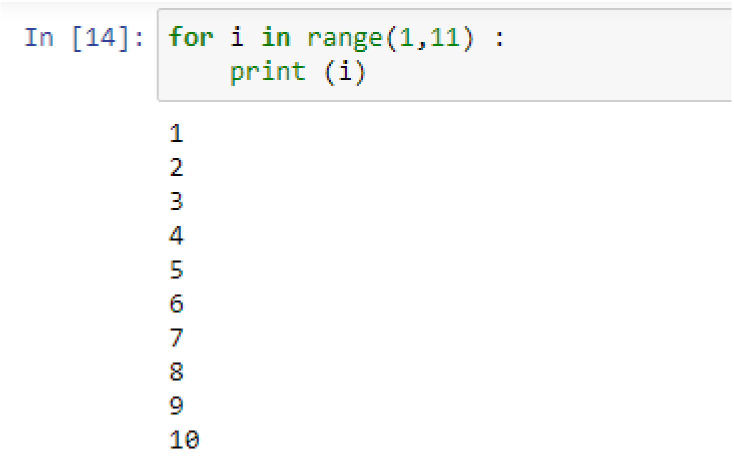


```
In [13]: for i in range(0,10,1) :
             print (i)

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

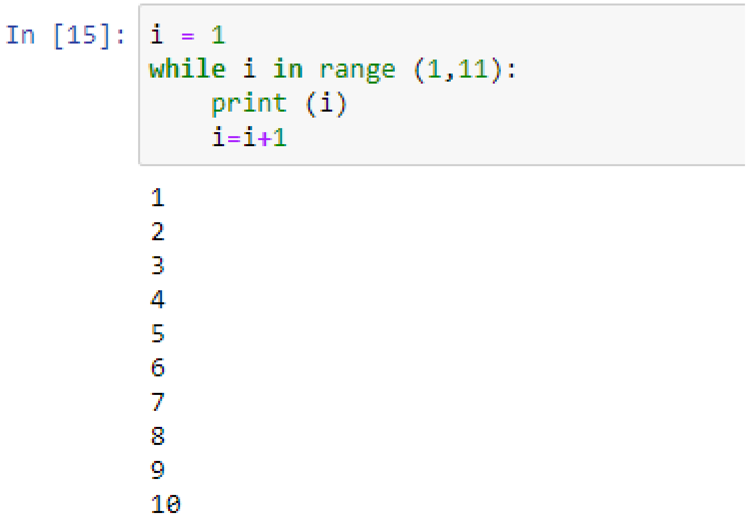
Here, Starting point is mentioned as **Zero**. End point is mentioned explicitly. Step size is mentioned as **One.**
These are differences between three cases.

12. Write a short program that prints the numbers 1 to 10 using a for loop. Then write an equivalent program that prints the numbers 1 to 10 using a while loop.



```
In [14]: for i in range(1,11) :
             print (i)

1
2
3
4
5
6
7
8
9
10
```

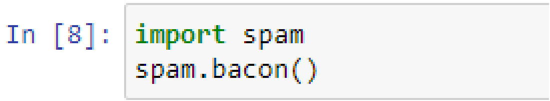


```
In [15]: i = 1
         while i in range (1,11):
             print (i)
             i=i+1

1
2
3
4
5
6
7
8
9
10
```

13. If you had a function named bacon() inside a module named spam, how would you call it after importing spam?

Considering Function as bacon() and Spam as a module. To call it we write following syntax as shown below:



```
import spam
spam.bacon()
```