Python FAQs

1. Is python case sensitive?

   Python is a case sensitive language.

2. How to write comments in python?

   Use '#' for single-line comments

   For multi-line, you can use three quotes

```
'''this is
a multi line
comment in
python'''

#single line comment
```

Figure 1: Comments in python

For multi-line comments, you can select all the lines that you want to comment on and then use 'ctrl + /' ('command + /' for mac) to comment on all the lines.

3. How to see help for documentation in python?

The help() function is used to display the documentation related to the functions, modules etc.

4. What is the difference between a list, a tuple, a set, and a dictionary?

List	Tuple	Set	Dictionary
A collection of items of any data type Example: X=['a',2,True,'b']	A collection of items of any data type Example: X=('a',2,True,'b')	A set in python is like a mathematical set. It does not hold duplicate values and arranges in ascending order Example: X={1,2,3,4}	A Python dictionary contains key-value pairs just like a real world dictionary has word and its meaning pair Example: X= {1:'Jan', 2:'Feb', 3:'Mar'}
Mutable (can be edited)	Immutable (Cannot be edited)	Mutable (can be edited)	Mutable (can be edited)
Supports Indexing	Supports Indexing	Does not Supports Indexing in python	Supports Indexing using keys
Function: list()	Function: tuple()	Function: set()	Function: dict()

5. Why are there so many different brackets used in python?

   There are three types of brackets used primarily in python:

   1. parenthesis **()**
   2. Square Brackets **[]**

3. Curly Brackets **{}**

[] square bracket is used for Indexing or accessing values within a list, tuple, dictionary or data frame, etc., also to represent a list or an array

() parenthesis is used to call functions and also used in Tuple.

{} curly brackets are used in dictionaries and sets.

6. What is indexing?

Indexing is used to access the items/data stored inside a list, a tuple, a dictionary, data frame, etc.

```
lst =[1,2,4,5] #list

dct={'one': 1, 'two':2, 'three':3} #dictionary

tupl=(1,2,3,4,5) #tuple

set={1,2,3,4} # set
```

Indexing value '2' from all of the above:

```
lst[1] = 2 # use index position to access the value

dct['two']=2 #use the key name to access the value

tupl[1]=2  # use index position to access the value

#set does not support indexing in python
```

Consider below DataFrame:

```
data= [[1,2],[3,4]] #a nested list (a list containing 2 lists)

df=pd.DataFrame([[1,2],[3,4]],columns=['Col1','Col2'])
```

	Col1	Col2
0	1	2
1	3	4

Indexing in DataFrame: Getting the value 2

```
df['Col2'] #will return all values in Col2

df['Col2'][0] #will return the value at index 0 of Col2
```

Using iloc in DataFrame to access values

```
df.iloc[0,1] #will give the value 2
```

'iloc' has two arguments, the first one is for rows and the second one is for columns. 0 in the above code gets the elements of the 0th index and 1 is for the second column in the data frame df.

```
df.iloc[:,1] #will give all rows of Col2
```

df.iloc[:,:] #will give all rows and columns

df.iloc[1,:] #will give all columns and second row

7. What is the difference between an array and a list?

   Both the list and array are similar but one major difference is that an array is optimised for arithmetical operations. (Numerical Python – numpy)

   List can hold items of different datatypes but an array can hold only items of the same data type.

8. What is the dimension of an array?

   The number of dimensions and items in an array is defined by its shape

   ```
   Arr=np.array([2,3,4])
   Arr.shape
   (3,)
   ```

9. What is the dimensionality of a Data-Frame?

   The shape of the dataset is the dimensionality which gives a tuple of number of rows and number of columns

   ```
   df.shape
   ```

10. The drop function in pandas doesn't save the new data-frame with a dropped column?

    Drop function has a parameter 'inplace' which is False by default. That means it will not make changes to the original data frame but returns a temporary copy of the data frame. You can use inplace=True inside the drop function to retain the changes.

    df.drop('column_name', axis=1,inplace=True)

11. Why is 'inplace' used in pandas?

    Inplace parameter, by default it's false, it creates a copy of the data frame so that it doesn't affect the original data.

    When inplace is True, the original data frame is affected.

12. How do I change the directory in Jupyter Notebook?

    To change the directory, write below command in your Jupyter Notebook:

    Example:

    ```
    import os

    os.chdir('c:/users/IT')  # changes directory to mentioned path
    ```

    To see the current working directory, write below command in your Jupyter Notebook:

```
import os

os.getcwd() #gets the current working directory
```

13. How do I install a package using Jupyter Notebook?

    Type the below command in your Jupyter Notebook:

    ```
    !pip install package-name
    ```

    Example: !pip install sklearn

14. What is a user defined function and why is it used?

    A function is used to automate a repetitive task. For example, you have to convert a string of Date-time to date-time format, you will not always write the same code again and again. Thus, a user defined function is used.

    Syntax:

    def function_name(arguments):

    ```
        code here
    ```

    To call a function, the syntax is: type 'function_name(arguments)' and run the code.

    Example:

    ```
    def square(x):
        return x*x;
    ```

    Calling it - square(2) and after running the code it returns square of 2 that is 4.

15. What is a function call?

    A function is just calling an already defined or a user defined function to perform some task

    Syntax: function_name(arguments)

    Example: sum function in python - sum([1,2,3]) gives sum of all elements of the list passed to it as an argument.

16. What is the difference between return and print when using a function?

A return statement returns a value computed during an operation performed by the function. The return statement is the main way through which a function returns a value.

Print statement gives the output in a string format to show users what is going on inside the program.

```
In [34]: def function_with_return():
             return "I returned"
         function_with_return()

Out[34]: 'I returned'

In [32]: def function_with_print():
             print ("I printed")
         function_with_print()

         I printed
```

Figure 2: Function Print & Return Code Snippet

17. What is OOPs?

OOPs is object oriented programming concept which allows users create the objects that they want and then create methods to handle those objects and manipulate these objects to get the results.

Core **OOPS** concepts are:

- Abstraction
- Encapsulation
- Polymorphism
- Inheritance
- Association
- Aggregation
- Composition

18. What is a class?

A class is a blueprint for creating objects and providing initial values for state (attributes) and implementations of methods(functions)

Use 'class' keyword to define a class, see below code snippet:

```
In [38]: class MyClass:
           x = 'Hello World'

In [39]: c= MyClass()
         c.x

Out[39]: 'Hello World'
```

Figure 3: Code snippet for Class

19. What is a keyword?

Keywords are the reserved words in Python. We can name a variable, function with same name as a keyword.

In the below example, when 'pass' is used the font colour is green which indicates it is a keyword.

```
In [41]: Pass= 1
         Pass

Out[41]: 1

In [43]: pass =1
         pass

           File "<ipython-input-43-d508d0a6cfd0>", line 1
             pass ==1
                  ^
         SyntaxError: invalid syntax
```

Figure 4: Error when using keyword as a variable

20. What is a data type?

Variables can hold values of different data types like int, float, string, boolean.

To check the data type, uses type function 'type()'

- Int means integers (non-decimal point numeric values)
- Float means numeric values with decimal points
- String means values with alphabets, alphanumeric words (non-numeric)
- Boolean means True or False

```
In [72]: type(True)
Out[72]: bool

In [45]: type(1)
Out[45]: int

In [46]: type(1.0)
Out[46]: float

In [47]: type('Hello World')
Out[47]: str

In [48]: type('$')
Out[48]: str
```

Figure 5: Type function example

21. How to plot multiple plots?

```
import matplotlib.pyplot as plt

fig = plt.figure()
fig.subplots_adjust(hspace=0.4, wspace=0.4)
for i in range(1, 7):
    x=[1,2,3,4,5]
    ax = fig.add_subplot(2, 3, i)
    plt.plot(x)

plt.show()
```

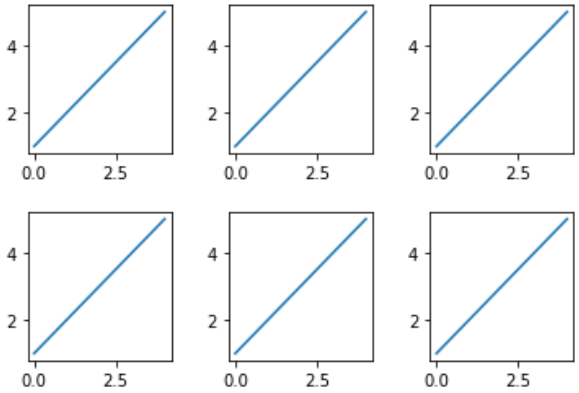

Figure 6: Multiple plots using matplotlib

22. What is the difference between groupby function and sortby functions?

DataFrame.sort_values("column") - returns a data frame that is sorted on the bases of the values of the column passed in the above code in default Ascending order.

DataFrame.groupby("column")- It is used to split the data into groups based on the values of the column given in the above code. In the above data frame, it will split the data frame with respect to values in col1 -like all A's will be together. now we can apply aggregate functions here - if I want to see the sum of col2 values with respect to every group of col1.(please refer to figure 7)

In [17]:
```
df = pd.DataFrame({ 'col1': ['A', 'A', 'B', np.nan, 'D', 'C'],'col2': [2, 1, 9, 8, 7, 4],'col3': [0, 1, 9, 4, 2, 3],})
df
```
executed in 10ms, finished 14:35:28 2020-03-11

Out[17]:

	col1	col2	col3
0	A	2	0
1	A	1	1
2	B	9	9
3	NaN	8	4
4	D	7	2
5	C	4	3

In [15]:
```
df.sort_values(by=['col1']) # return the data frame that is sorted on the bases of values in col1
```
executed in 10ms, finished 14:35:00 2020-03-11

Out[15]:

	col1	col2	col3
0	A	2	0
1	A	1	1
2	B	9	9
5	C	4	3
4	D	7	2
3	NaN	8	4

In [16]:
```
df.groupby("col1")["col2"].sum()
```
executed in 64ms, finished 14:35:14 2020-03-11

Out[16]:
```
col1
A    3
B    9
C    4
D    7
Name: col2, dtype: int64
```

Figure 7: Groupby and sort_values using Pandas