General

# Explain:

What is Jupyter and what is Spyder?  
Jupyter: Line by line compilation, command line interface is really good, showcase your work as a data scientist,

Spyder: Mostly for programers, Its not line by line compiler, It has 3-4 views which are required for programming.

We will use Jupyter for our data science purpose as it is more user friendly

Explain How Jupyter works

# Our First Program:

1. print(‘Hello World!’)

print(’Write whatever you want and I will write it for you ’)

1. Whatever we write after ‘#’ is comment and it is not processed.

When we write any code we should write comments so that debugging and referencing is easy

1. print(‘Hello # World’)
2. make sure that we write print in small letters and not capital

# Scalar objects:

1. **int** is used for integers e.g. 1, 0, 1000, 5000

**float** is used to represent real numbers e.g. 3.0, 4.0, 100.5, 3.14, 4e3 (4 times 10 to the power of 2) etc

**bool** is used for binary values like **True and False**

**None** is type with single value. We will talk about this when we come to variables

1. + plus

- minus

/ divide

\* multiply

% reminder (I % j is pronounced as “i mod j”)

\*\* power

< less- than

> greater- than

<= less- than- equal

>= greater- than- equal

# Variables and assignments:

Variables provide a way to associate names with objects. These names, associated with objects can be used and called again and again, instead of objects.

An assignment statement associates the name to the left of ‘=’ to the object denoted by the expression to the right of ‘=’.

E.g. pi = 3.14  
 radius = 4  
 area = pi \* (radius \*\* 2)

**Reserved Keywords:** and, del, from, not, while, as, elif, global, or, with, assert, else, if, pass, yield, break, except, import, print, class, exec, in, raise, continue, finally, is, return, def, for, lambda, try

# Readability of code:

a = 3.1416 pi = 3.1416   
b = 4 radius = 4   
c = a\*(b\*\*2) area = pi\*(r\*\*2)

Which code makes more sense?

**Note :**

* **Str() converts object to string**
* **\n is used to go on the new line**

Python Objects and data structures:

String:

Objects of type ‘str’ are used to represent strings of characters. They can be written using single or double quotes. Strings are one of several sequence types in python. Strings **ARE NOT MUTABLE** i.e. the elements in the strings can not be changed.

‘123’ is string of characters and not a number one hundred and twenty-three.

# Operations with String:

* **len() :** Calculates length of the string
* **Indexing & Slicing:** (See the python notebook)

Lists:

Lists can be thought as the most general version of a sequence. Lists **ARE MUTABLE** i.e. the elements of lists can be changed. It can be written as a list of comma-separated values (items) between square brackets.

# Operations with List:

* **Indexing and accessing the elements of list**
* **Updating list**
* **Basic List operations**
* **Built-in List Functions**

(See Python notebook)

Dictionaries:

Let up learn about mapping in python. If you are familier with data structures, you can think of dictionaries as hash tables.

Dictionaries are a collection of objects that are stored by a ‘key’ (In sequences, objects are stored by their relative position). Dictionaries consists of a key and its associated value. This value can be any object.

There are some more data types like tuples, files, sets and Booleans. However, with time constrain we are not going to learn that in the class. Since now you have developed intuition about python objects and data structures, it should not be very hard to learn it by yourself.

Python Statements:

Remember: 2 main factors in python statement are ‘white space (indentation)’ and a ‘colon’. The statement is ended with a colon and whitespace (indentation) is used to describe what takes place in case of the statement.

We will see following python statements:

## if, elif, else statement:

Add text here

## for Loop:

A for loops acts as iterator in Python. It goes through iterable items. Objects that we can iterate over include strings, lists, tuples and built-in iterables for dictionaries such as keys or values.

### while loop:

‘while’ loop is used to perform iterations. A **while** statement executes a statement(s) until a true value occurs. The code statement(s) are looped again and againuntill the condition is met.

while test:  
 code statement  
 else:  
 other code statement

We can also use **break, continue & pass** statements in our loop to add additional functionality.

break: Breaks out of the current closet enclosing loop  
continue: Goes to the top of the closest enclosing loop  
pass: does nothing at all

while test:  
 code statement  
 if test1:  
 break  
 if test2:  
 continue