## Topics to be Covered:

### Interactive Programming using Python

#### 1.Interactive programming using :

#### IDLE,Jupyter Notebook and Google Colab

#### 2.variable creation

#### 3.use of type()

#### 4.Input and Output function

#### 5.Indetations and Comments

#### 6.Conditional statement : if,else,elif

### Literate Programming

### Markdown and Jupyter Notebooks

### Introduction to Python Programming Language

=====================================================================

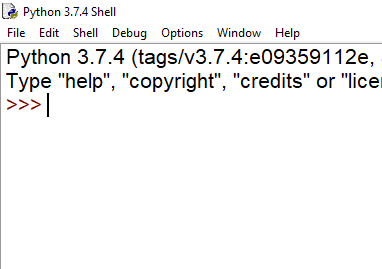
### Interactive Programming using Python:

Interactive programming, also known as live coding, refers to any **computer programming** language that allows the creator to make changes to the program while it is already running. In traditional programming, the coder first writes out the program and then saves it. He then runs the program on the computer. If an error occurs, it's back to the drawing board to type out new code and run the program all over again. With interactive programming, the designer can make changes to the code without having to run the program over again.

Another use for interactive programming is to allow input from the user in what is called an interactive application. This can be as simple as asking the user for her name and then displaying it on the screen. The program has an interactive element by changing the value of the user's name based on what she types. When the program was created, it did not know her name and the value was blank. Once it learned her name, it put that value into the program while the program was still running and then displayed it on the screen.

Python is interactive. When a Python statement is entered, and is followed by the Return key, if appropriate, the result will be printed on the screen, immediately, in the next line. This is particularly advantageous in the debugging process. In interactive mode of operation, Python is used in a similar way as the Unix command line or the terminal.

The interactive Python shell looks like:



Interactive Python is very much helpful for the debugging purpose. It simply returns the >>> prompt or the corresponding output of the statement if appropriate and returns **error** for incorrect statements. In this way if you have any doubts like: whether a syntax is correct, whether the module you are importing exists or anything like that, you can be sure within seconds using Python interactive mode.

An illustration for interactive python looks like this:

Example : simple Calculations with numbers and strings

### Variable creation:

### Use of type():

### Input and Output functions : input() and print()

### Indentations and Comments:

### conditional statement: if,else,elif