Python Language Fundamentals

Python Character Set

Character set is a set of valid characters that a language can recognize.

Letters	A-Z, a-z
Digits	0-9
Special Symbols	Space + - * / ** () []
White Spaces	Blank space, tab, carriage return, new line, form feed
Other Characters	Python can process all ASCII and Unicode characters as part of data or literals.

Tokens

The smallest individual unit in a program is known as a Token or a lexical unit. Python has following tokens:

i) Keyword
ii) Identifiers
iii) Literals
iv) Operators
v) Punctuators

Keywords

Keywords are the special word that convey special meaning to the language compiler/ interpreter. Python programming language contains the following keywords
for e.g.

```
def  for  in   or   while    class         from     if   import
     finally  etc
```

Identifier

They are general names given to different part of the program viz variable, object, classes, functions, list, dictionaries etc.

Rules for Identifier

- The first character must be a letter; the underscore(_) counts as a letter.
- The digits 0-9 can be part of the identifier except for the first character.
- Identifiers are unlimited in length.
- Case is significant in Python i.e. Python case sensitive, as it treat uppercase, lowercase letter differently.
- An identifier must not be a keyword of Python.
- An identifier cannot contain special character except underscore(_).

Literals/ Values

Literals are data items that have a fixed value. Python allows several kind of literals:

i) String Literal
ii) Numeric Literal
iii) Floating Point Literal
iv) Boolean Literal
v) Special Literal None

String Literal

The text enclosed in quotes forms a string literal. For e.g. ‘abc’, “abc” all are string literal in Python.

Python allow you have certain non graphic characters in string value. Non graphic characters are those that cannot be typed directly from keyboard. e.g. backspace, tab, carriage return etc.

The non graphic character can be represented using escape sequence i.e. represented by backslash (\) followed by one or more character.

\\	\n
\’	\t
\”	\r
\b	

Python allows two types of string

i) Single line String

   The string enclosing text in (‘ ’) or double quotes(“ ”) are normally single line string i.e. must be terminate in one line

   Text1 = ‘hello there’

ii) Multiple line String

   Multiple line strings can be created in two ways

   a. By adding a backslash at the end of the normal single quote/double quote string.

```
Text1 = ‘hello\
        World’
```

   b. By typing the text in triple quotation marks. Both triple apostrophe(‘’’) or triple quotation marks will work.

```
str1 = '''Hello
       world
       There I come!!!
       cheers.'''
str2 = """Hello
       world
       This is another multiline string"""
```

Numeric Literals

The numeric literals in Python can belong to any of the following four different numerical types

Int	Integers are +ve or –ve whole numbers with no decimal point.
Long	Long are integers of unlimited size. Written like integers and followed by L or l
Float	Numbers written with decimal point.
Complex	Are in the form of a + bj

Python allows three types of **Integer Literals:**

i) Decimal Integer literal
An integer literal consisting of a sequence of digits taken to be decimal integer literal unless it begins with 0 (digit zero)
`For e.g.  124, 41, +97, -17`

ii) Octal Integer literal
Octal integer literal consisting of a sequence of digits with 0 is taken to be an octal integer.
`For e.g.  010,  014`

iii) Hexadecimal literal
A sequence of digits preceded by 0x or 0X is taken to be an hexadecimal integer.
`For e.g. 0xAX,  OXB15`

Floating Point Literal

1. Fractional form : A real literal in fraction form consist of signed or unsigned digits including a decimal point between digits.
`2.0, 17.5, -13.0, -0.00625`

2. Exponent form

    0.58 X 101 = 0.58 E 01

    The following are the valid real literal 152E05, 1.52E07, 0.152E08

Boolean Literal

Python supports two Boolean values i.e. True or False.

Special Literal None

Python has one special literal which is **None**. The None value in Python means "There is no useful information." or "There is nothing here."

Operators

Operators are token that trigger some computation when applied to variables and other object in an expression.

Arithmetic Operators

+	Addition	%	Remainder/Modulus
-	Subtraction	**	exponent
*	Multiplication	/	Division

Punctuators

Punctuators are the symbols that are used in programming language to organize sentence structure, and indicate the rhythm and emphasis of expression statement and program structure.

Most common punctuators are

```
' " # \ () [] {} @ , : . = ;
```

Comments

Comments are the additional readable information which is read by programmer but ignored by Python interpreter.

The physical line beginning with **#** are the full line comments.

# Definition of function

Multiple line comments

'''----------------------

---------------------------'''

Working in Python

once you have Python installed on your computer, you are ready to work on it. You can work in Python in following different ways:

i) in Interactive mode

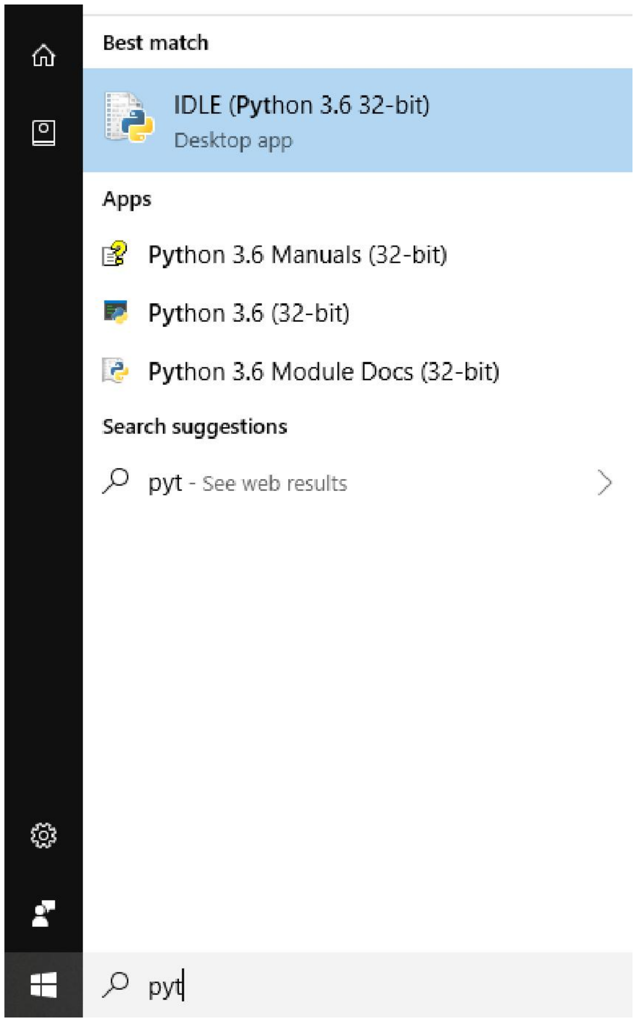
ii) in Script mode

Working in Interactive mode

Interactive mode of working means you type the command – one command at a time, and the Python executes the given command there and then and gives you output. In interactive mode, you type the command in front of Python command prompt >>>.

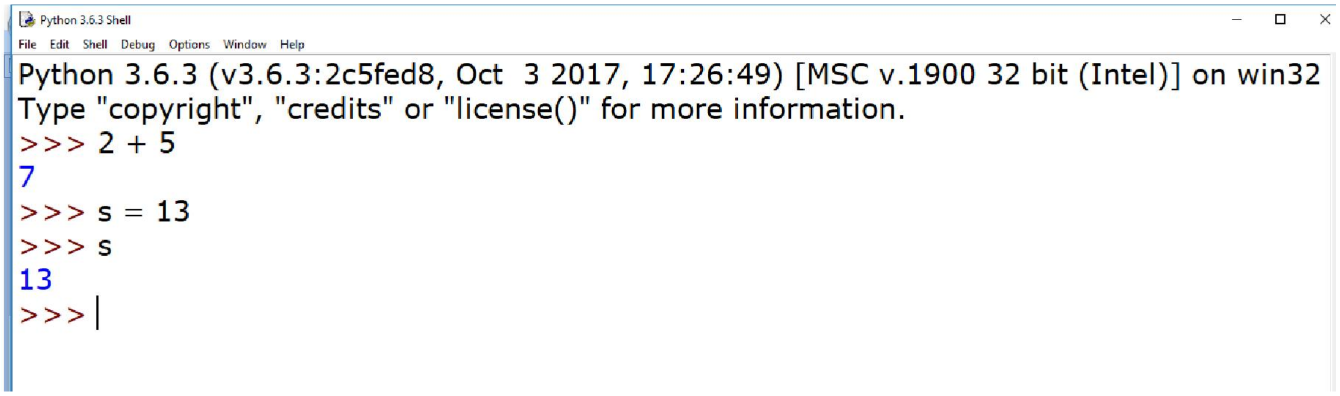
To work in interactive mode, follows the process given below:

i) Click start button ---> All Programs --→ Python 3.6 IDEL



ii) it will open Python Shell, where you will see the Python prompt (three '>' signs i.e. >>>)

iii) Type the command in from of this Python prompt and Python will immediately give you the result.



```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 5
7
>>> s = 13
>>> s
13
>>> |
```

Working in Script Mode

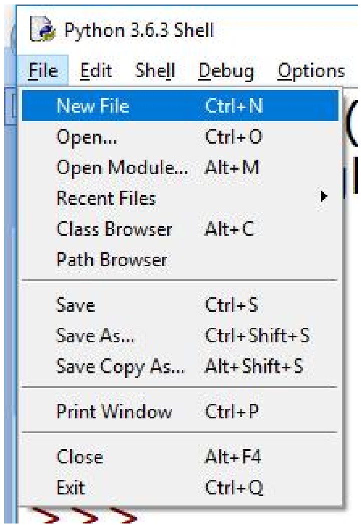
What if you want to save all the commands in the form of program files and want to see all the output lines together rather than, with interactive mode you cannot do so, for:

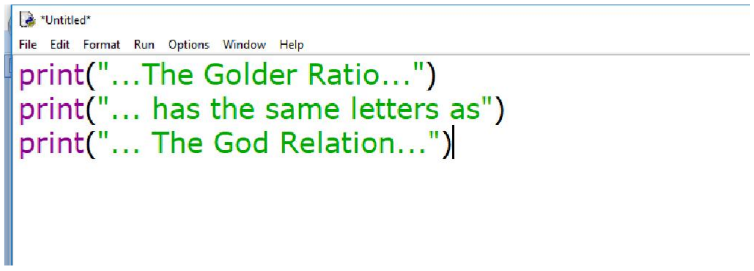
The solution of above problem is the Script mode. To work in a script mode, you need to do the following:

i) Click File ----> New in IDEL Python shell

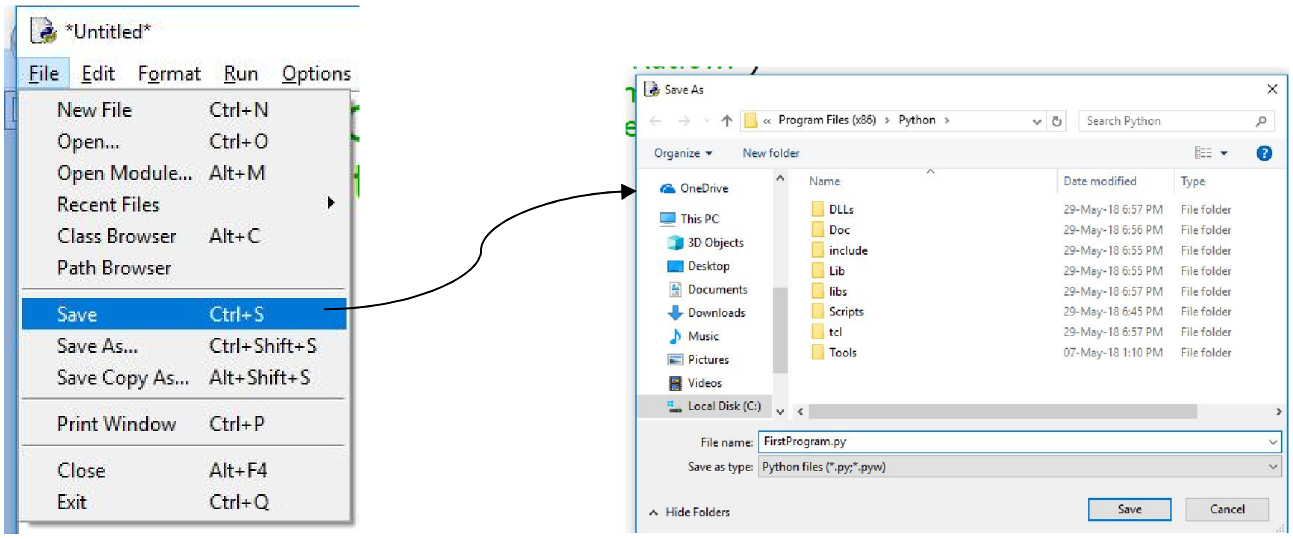
ii) In the New window that opens, type the commands you want to save in the form of a program.

iii) Click File -----> Save and then save the file with an extension **.py** . The Python programs has .py extension.





```
print("...The Golder Ratio...")
print("... has the same letters as")
print("... The God Relation...")
```

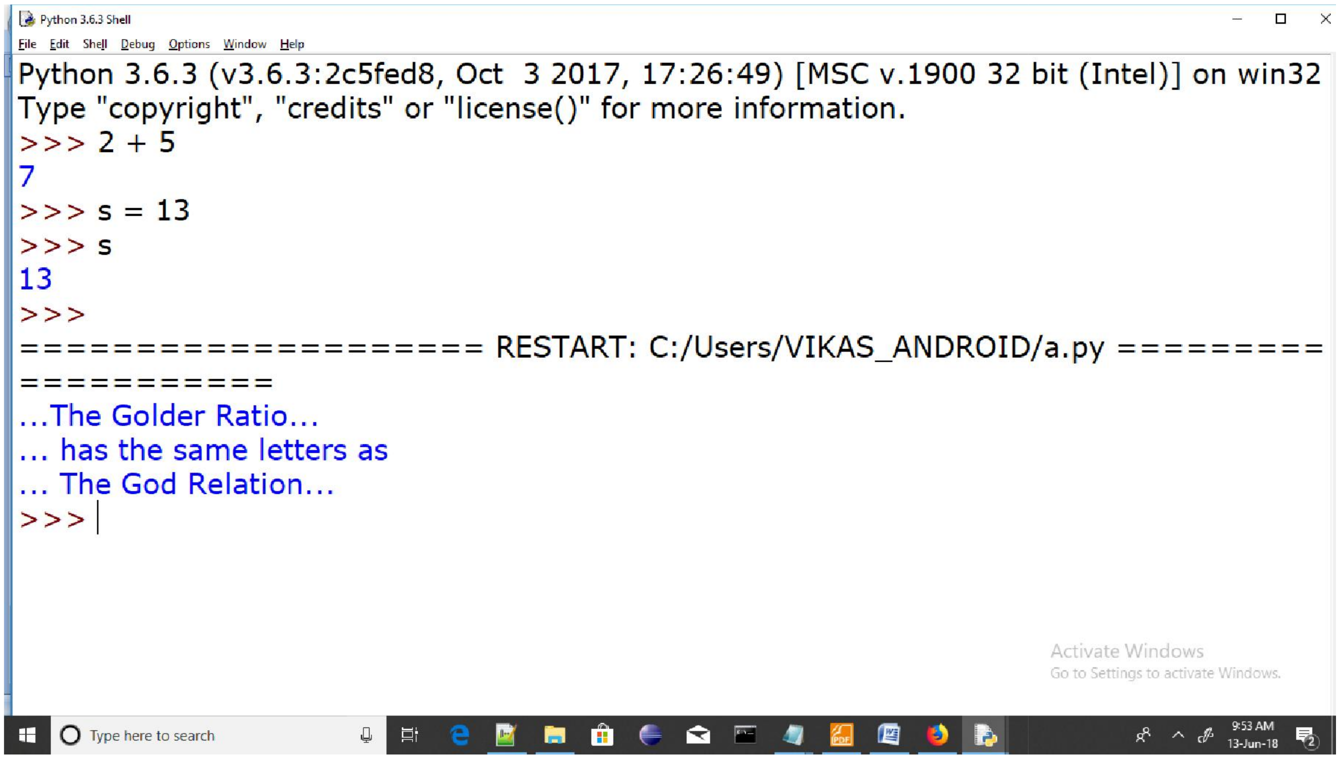


Now your program would be saved on the disk and the saved file will have .py extension.

iv) Click Run -----> Run Module command in the open program / script file's window.

You may also press F5 key.

And it will execute all the commands stored in module/program/script that you have opened.



```
Python 3.6.3 (v3.6.3:2c5fed8, Oct  3 2017, 17:26:49) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 5
7
>>> s = 13
>>> s
13
>>>
==================== RESTART: C:/Users/VIKAS_ANDROID/a.py =========
===========
...The Golder Ratio...
... has the same letters as
... The God Relation...
>>> |
```