Name:- Divyang Bagla

ERP ID:- 1032180739

Batch A2

Roll No. :- PC33

Lab Assignment No. 1

Aim:-

Getting Started with Python:

- Install Python
- Verify Installation
- Perform simple operations with respect to
  1. Keyword 2. Literals 3. Comments 4. DocString
  5. Indentation Error 6. "Hello World" 7. Single and Multi-line statement

Theory:-

Steps to install python in windows 10:-

1. Step 1 − Select Version of Python to Install. ...
2. Step 2 − Download Python Executable Installer. ...
3. Step 3 − Run Executable Installer. ...
4. Step 4 − Verify Python is installed on Windows. ...
5. Step 5 − Verify Pip was installed.

Verify Installation :-

```
C:\Users\Divyang>py --version
Python 3.8.8
```

Define Following :-

Keywords :- Python keywords are special reserved words that have specific meanings and purposes and can't be used for anything but those specific purposes. These keywords are always available—you'll never have to import them into your code.

Python keywords are different from Python's **built-in functions and types**. The built-in functions and types are also always available, but they aren't as restrictive as the keywords in their usage.

An example of something you *can't* do with Python keywords is assign something to them. If you try, then you'll get a **SyntaxError**. You won't get a SyntaxError if you try to assign something to a built-in function or type, but it still isn't a good idea.

```
>>> help("keywords")

Here is a list of the Python keywords.  Enter any keyword to get more hel

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not
```

Literals :- Literals are a notation for representing a fixed value in source code. They can also be defined as raw value or data given in variables or constants.

Numeric literals

```
x = 24
y = 24.3
z = 2+3j
print(x, y, z)
```

Comments :- A comment in Python **starts with the hash character, # , and extends to the end of the physical line**. A hash character within a string value

is not seen as a comment, though. To be precise, a comment can be written in three ways - entirely on its own line, next to a statement of code, and as a multi-line comment block.

DocString:- Python documentation strings (or docstrings) provide a convenient way of associating documentation with Python modules, functions, classes, and methods.

It's specified in source code that is used, like a comment, to document a specific segment of code. Unlike conventional source code comments, the docstring should describe what the function does, not how.

What should a docstring look like?

- The doc string line should begin with a capital letter and end with a period.
- The first line should be a short description.
- If there are more lines in the documentation string, the second line should be blank, visually separating the summary from the rest of the description.
- The following lines should be one or more paragraphs describing the object's calling conventions, its side effects, etc.

Example:-

```
def my_function():

    '''Demonstrates triple double quotes
    docstrings and does nothing really.'''

    return None

print("Using __doc__:")
print(my_function.__doc__)

print("Using help:")
help(my_function)
```

Output:-

```
Using __doc__:

Demonstrates triple double quotes

    docstrings and does nothing really.

Using help:

Help on function my_function in module __main__:


my_function()

    Demonstrates triple double quotes

    docstrings and does nothing really.
```

Indentation Error:- Python is a procedural language. The indentation error can occur when the spaces or tabs are not placed properly. There will not be an issue if the interpreter does not find any issues with the spaces or tabs. If there is an error due to indentation, it will come in between the execution and can be a show stopper.

Example:-

```
site = 'edu'
if site == 'edu':
print('Logging in to EduCBA!')
else:
print('Please type the URL again.')
print('You are ready to go!')
```

In above there is an indentation error is present.

Conclusion:- Installed python and learned about the keywords, literals, single and multi line comments, indentation error etc.

Lab Assignment 1Code

Lab Assignment No. 1

KeyWords

In [1]:
```
help("keywords")
```

```
Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not
```

Literals

In [2]:
```
x = 24
y = 24.3
z = 2+3j
print(x, y, z)
```

```
24 24.3 (2+3j)
```

In [4]:
```
s = 'python'

# in double quotes
t = "python"

# multi-line String
m = '''geek
            for
                geeks'''

print(s)
print(t)
print(m)
```

```
python
python
geek
            for
                geeks
```

Comments in Python

In [5]:
```
# Single Line Comments
```

In [9]:
```
'''
THIS IS A MULTILINE COMMENT
USING STRING LITERALS!
'''
```

Out[9]:
```
'\nDO NOT FORGET TO PROPERLY\nINDENT THE  STARTING OF STRING \nLITERALS WITHIN YOUR CODE! '
```

DocString

In [10]:
```
def my_function():
    '''Demonstrates triple double quotes
    docstrings and does nothing really.'''

    return None

print("Using __doc__:")
print(my_function.__doc__)

print("Using help:")
help(my_function)
```

```
Using __doc__:
Demonstrates triple double quotes
    docstrings and does nothing really.
Using help:
Help on function my_function in module __main__:

my_function()
    Demonstrates triple double quotes
    docstrings and does nothing really.
```

Indentation Error

In [11]:
```
n = 10
for i in range(0,n):
print(i)
```

```
  File "<ipython-input-11-0ce7dd5839d8>", line 3
    print(i)
    ^
IndentationError: expected an indented block
```

Print "Hello World"

In [13]:
```
print("Hello World !")
```

```
Hello World !
```

Single And Multi Line Statements

In [14]:
```
#single line statement
print("This is a trial code")
#multi line statement
print('''Welcome!!
This is Jupter Notebook
Assign 1''')
```

```
This is a trial code
Welcome!!
This is Jupter Notebook
Assign 1
```

In [ ]:
```

```