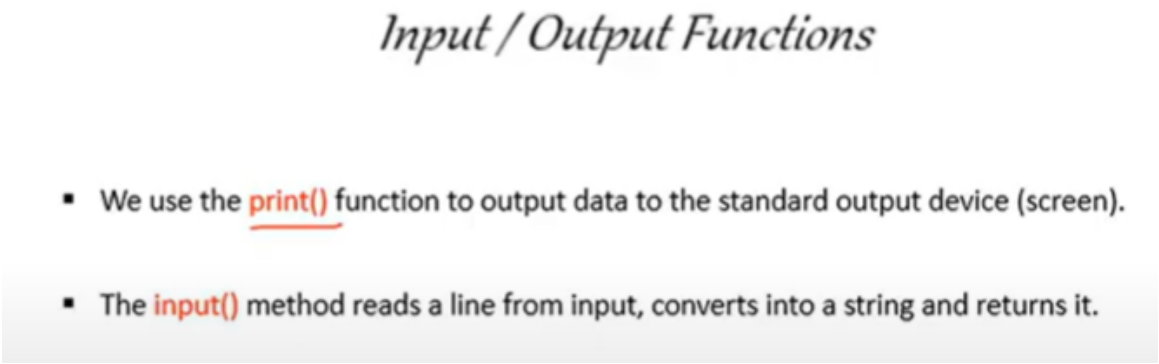
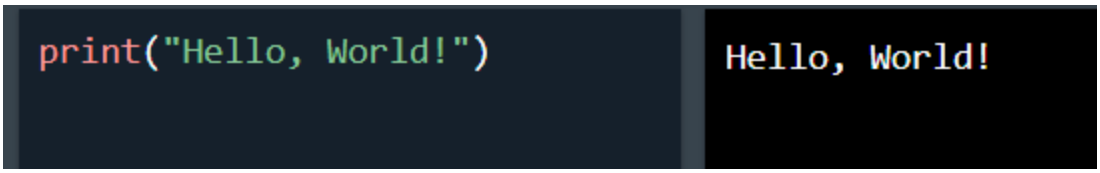
Python(Notes) part-1

python input/output :





Built-in Functions(——input()——)

Python input() Function

Definition and Usage

The `input()` function allows user input.

```
print("Enter your name:")
x = input()
print("Hello, " + x)
```

```
Enter your name:
Hasna Hena Joti
Hello, Hasna Hena Joti
```

```
x = input("Enter your name:")
print("Hello, " + x)
```

```
Enter your name:Hasna Hena Joti
Hello, Hasna Hena Joti
```

Variables

Python Variables

Variables are containers for storing data values.

```
x = 5
y = "John"
print(x)
print(y)
```

```
5
John
```

Variables do not need to be declared with any particular *type*, and can even change type after they have been set.

Example

```
x = 4       # x is of type int
x = "Sally" # x is now of type str
print(x)
```

Casting

If you want to specify the data type of a variable, this can be done with casting.

Example

```
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
```

```
x = str(3)
y = int(3)
z = float(3)

print(x)
print(y)
print(z)
```

```
3
3
3.0
```

Get the Type

You can get the data type of a variable with the `type()` function.

Example

```
x = 5
y = "John"
print(type(x))
print(type(y))
```

```
x = 5
y = "John"
print(type(x))
print(type(y))
```

```
<class 'int'>
<class 'str'>
```

python id() function

Python id() Function

Definition and Usage

The `id()` function returns a unique id for the specified object.

All objects in Python has its own unique id.

The id is assigned to the object when it is created.

The id is the object's memory address, and will be different for each time you run the program. (except for some object that has a constant unique id, like integers from -5 to 256)

```
x = ('apple', 'banana', 'cherry')
y = id(x)
print(y)

# This value is the memory address
of the object and will be
different every time you run the
program
```

```
60314731
```

A variable can have a short name (like x and y) or a more descriptive name.

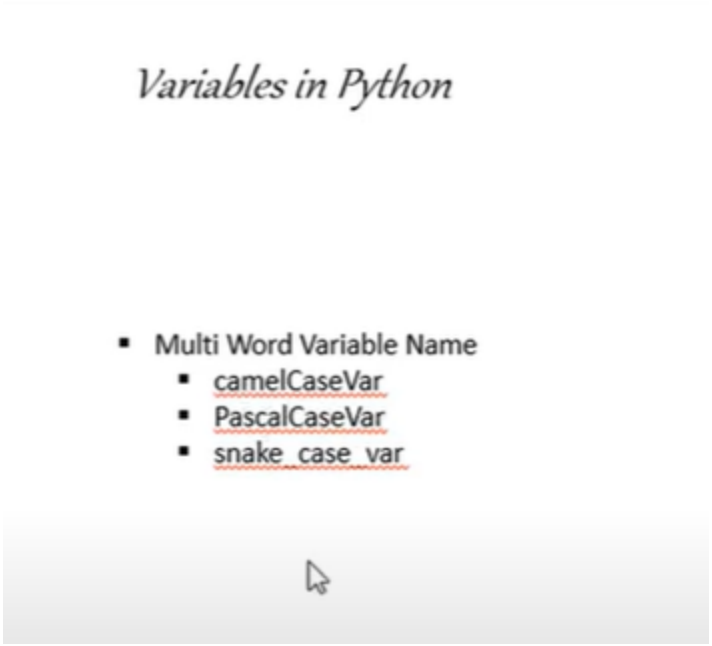
- A variable name must start with a letter or the underscore (_) character.
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-Z, 0-9, and _).
- Variable names are case-sensitive (x, X, _x are three different variable).

Valid Example:

```
Var = 10
Var2 = 100
_var = 20
Var_2 = 10
V1a2r3 = 30
My_name = 'shakil'
```

Invalid Example:

```
9Var = 'data science'
Var-2 = 'study mart'
&var = 20
My name = 'shakil'
```



Case	Description	Example
CamelCase	First word is lowercase, subsequent words are capitalized.	`myVariableName`
Snake_Case	Words are separated by underscores, all lowercase.	`my_variable_name`
PascalCase	Every word starts with a capital letter, no spaces.	`MyVariableName`

python local Variables

Python Local Variables

Local variables in Python are those which are initialized inside a function and belong only to that particular function. It cannot be accessed anywhere outside the function. Let's see how to create a local variable.

Creating local variables in Python

Defining and accessing local variables

```
def f():

    # local variable
    s = "I love Geeksforgeeks"
    print(s)


# Driver code
f()
```

Output

```
I love Geeksforgeeks
```

Can a local variable be used outside a function?

If we will try to use this local variable outside the function then let's see what will happen.

```
def f():

    # local variable
    s = "I love Geeksforgeeks"
    print("Inside Function:", s)

# Driver code
f()
print(s)
```

Output:

```
NameError: name 's' is not defined
```

python Global variables

Python Global Variables

These are those which are defined outside any function and which are accessible throughout the program, i.e., inside and outside of every function. Let's see how to create a Python global variable.

Create a global variable in Python

Defining and accessing Python global variables.

```
# This function uses global variable s
def f():
    print("Inside Function", s)

# Global scope
s = "I love Geeksforgeeks"
f()
print("Outside Function", s)
```

Output

```
Inside Function I love Geeksforgeeks
Outside Function I love Geeksforgeeks
```

The variable s is defined as the global variable and is used both inside the function as well as outside the function.

If a variable with the same name is defined inside the scope of the function as well then it will print the value given inside the function only and not the global value.

```
# This function has a variable with
# name same as s.
def f():
    s = "Me too."
    print(s)

# Global scope
s = "I love Geeksforgeeks"
f()
print(s)
```

Output

```
Me too.
I love Geeksforgeeks
```