Python Print() Statement

print() is a function in Python that allows us to display whatever is written inside it. In case an operation is supplied to print, the value of the expression after the evaluation is printed in the terminal. For example,

```
print("Hello World!!")
print(2+4)
print(67*22)

print("Hello Student", end="")
print("Welcome to Python Tutorial")


# -----OUTPUT-----
Hello World!!
6
1474
Hello StudentWelcome to Python Tutorial
```

Escape Sequences

- An **Escape Sequence** character in Python is a sequence of characters that represents a single character.
- It doesn't represent itself when used inside string literal or character.
- It is composed of two or more characters starting with backslash \ but acts as a single character. Example \n depicts a new line character.

Some more examples of escape sequence characters are shown below:

Escape Sequences	Description
\n	Inserts a new line in the text at the point
\\	Inserts a backslash character in the text at the point
\"	Inserts a double quote character in the text at that point
\'	Inserts a single quote character in the text at that point
\t	Inserts a tab in the text at that point
\f	Inserts a form feed In the text at that point
\r	Inserts a carriage return in the text at that point
\b	Inserts a backspace in the text at that point

```
print("I am good at Python.\nWe will be learing it from Now")
print("Hello World\tPython is good.")
print("We can use backslash like this \\.")
print("Good Morning","We are studying Python.")

# -----OUTPUT-----
I am good at Python.
We will be learing it from Now
Hello World     Python is good.
We can use backslash like this \.
Good Morning We are studying Python
```

Comments

Comments are used to write something which the programmer does not want to execute. Comments can be written to mark author name, date when the program is written, adding notes for your future self, etc.

There are two types of comments in Python Language -:

- Single Line Comment
- Multi Line Comment

Single Line Comment: Single Line comments are the comments which are written in a single line, i.e., they occupy the space of a single line.

- We use # (hash/pound to write single-line comments).

```
print("Hello Student.")

# I can write whatever I want to write

print("This is just a print statement.") # I can write this too
# This Is a comment and It will not execute



# -----OUTPUT-----
Hello Student.
This is just a print statement.
```

Multi-Line Comment: Multi-Line comments are the comments which are created by using multiple lines, i.e., they occupy more than one line in a program.

- We use ' ' '..... Comment ....' ' ' for writing multi-line comments in Python (Use lines enclosed with three quotes for writing multi-line comments). An example of a multi-line comment is shown below:

```
'''This is a comment
Date: 2 January 1996
Multi-line comment ends here
'''
print("Hello Student!!!")

# Remember this is Single Line Comment
print("I am learning Python.")



# -----OUTPUT-----
Hello Student!!!
I am learning Python.
```

Variables

A variable is a name given to any storage area or memory location in a program.

In simple words, we can say that a variable is a container that contains some information, and whenever we need that information, we use the name of that container to access it. Let's create a variable:

```
var1 = "Hello World"
var2 = 23
var3 = 56.66
var4 = 55+77
print(var1)
print(var2)
print(var3)
print(var4)


# -----OUTPUT-----
Hello World
23
56.66
132
```

Multiple Assignment of Variables can also be done. This mean defining multiple variable in one line.

```
a, b, c = 5, 10, 15
print(a, b, c)

x = y = z = "Hello World"
print(x)
print(y)
print(z)
print(x, y, z)


# -----OUTPUT-----
5 10 15
Hello World
Hello World
Hello World
Hello World Hello World Hello World
```


Data Types

A data type, in programming, is a classification that specifies which type of value a variable has. Primarily there are following data types in Python.

- Integers (<class 'int'>): Used to store integers
- Floating point numbers (<class 'float'>): Used to store decimal or floating-point numbers
- Strings (<class 'str'>): Used to store strings
- Booleans (<class 'bool'>): Used to store True/False type values
- None: None is a literal to describe 'Nothing' in Python

Rules for defining a Data Type in Python:

- A variable name can contain alphabets, digits, and underscores (_). For E.g. : demo_xyz = 'It's a string variable'
- A variable name can only start with an alphabet and underscore.
- It can't start with a digit. For an example, 5harry is illegal and not allowed.
- No white-space is allowed to be used inside a variable name.
- Also, reserved keywords are not recommended to be used as variable names.

```
# Variable in Python:
abc = "It's a string variable"
_abcnum = 40 # It is an example of int variable
abc123 = 55.854 # It is an example of float variable
print(_abcnum + abc123) # This will give sum of 40 + 55.854


# -----OUTPUT-----
95.854
```


type() Function in Python

type() function is a function that allows a user to find data type of any variable. It returns the data type of any data contained in the variable passed to it.

Have a look at the code below which depicts the use of type function:

```
# type() Function in Python:
demo1 = "40"
demo2 = 55.5
print(type(demo1)) #It will give output as string type
demo3 = type(demo1) #It will return data type as float
print(demo3) #It will print that data type
print(type(demo2)) #It will give output as int type

# -----OUTPUT-----
<class 'str'>
<class 'str'>
<class 'float'>
```

Note – We can't do arithmetic operations of numbers with strings i.e. we can't add a string to any number. Have a look at the example below:

```
var1 = "It's a String"
var2 = 5
print(var1+var2)  # It will give an
                  #error as we can't add string to any number.


# -----OUTPUT-----
Traceback (most recent call last):
  File "test.py", line 3, in <module>
    print(var1+var2)
TypeError: can only concatenate str (not "int") to str
```

Note – We can add (concatenate) two or more strings and the strings will be concatenated to return another string. Here's the example showing that:

```
var1 = "My Name is "
var2 = "xyz"
var3 = var1+var2+" & I am a Good Boy."
print(var1+var2) # It will give output 'My Name is xyz'
print(var3)
```

Typecasting

Typecasting is the way to change one data type of any data or variable to another datatype, i.e. it changes the data type of any variable to some other data type.

I know it's a bit confusing but let me tell you in a simple manner. Suppose there is a string "34" Note: String is not integer since it is enclosed in double-quotes) and as we know we can't add this to an integer number let's say 6. But to do so we can typecast this string to int data type and then we can add 34+6 to get the output as 40. Have a look at the program below:

```
# Typecasting in Python :
abc = 5
abc2 = '45'
abc3 = 55.95
xyz = 5.0

abc4 = int(abc2)

print(abc+abc4)
print(abc+int(abc2))

print(float(abc)+xyz)

# It will give an error as abc has been changed into string.
print(str(abc)+45)
```

```
# -----OUTPUT-----
50
50
10.0
Traceback (most recent call last):
  File "c:/Users/Anirudh/Desktop/test.py", line 15, in <module>
    print(str(abc)+45)
TypeError: can only concatenate str (not "int") to str
```

There are many functions to convert one data type into another type :

- str() – this function allows us to convert some other data type into a string.
- int() – this function allows us to convert some other data type into an integer. For example, str("34") returns 34 which is of type integer (int)
- float() – this function allows us to convert some other data type into floating-point number i.e. a number with decimals.

Input Function

This function allows the user to receive input from the keyboard into the program as a string.

input() function always takes input as a string i.e. if we ask the user to take a number as input even then it will take it as a string and we will have to typecast it into another data type as per the use case.

If you enter 45 when input() is called, you will get "45" as a string.

```
# Input Function in Python:
print("Enter your name : ")
name = input()  # It will take input from user
print("Your Name is", name)  # It will show the name
xyz = input("Enter your age : ")
print("Your age is", xyz)


# -----OUTPUT-----
Enter your name :
Rahul
Your Name is Rahul
Enter your age : 45
Your age is 45
```

More Print Statement

While printing you can print out the variable name using print() function.

```
age = 50
name = "Elon Musk"

print("Hello! My name is %s." % name)
print("My age is %d." % age)
print("Hello! My name is %s and my age is %d." % (name, age))


# -----OUTPUT-----
Hello! My name is Elon Musk.
My age is 50.
Hello! My name is Elon Musk and my age is 50.
```

Questions

1) Write a program that takes two numbers as input from the user and then prints the sum of these numbers.
2) Write a Program that takes Length and Breadth as input from user and print the Area of Rectangle.
3) Ask 3 numbers from User and Calculate the Average.

H.W

1) Calculate sum of 5 subjects and Find percentage.
2) Ask marks in 5 subjects and calculate Total Marks and Find percentage.
3) Ask value in Rupees and Convert into Paise.
4) Calculate Area of Square by taking Length from User.
5) Ask number of games played in a tournament. Ask the user number of games won and number of games loss. Calculate number of tie and total Points. (1 win= 4 points, 1 tie =2 points)