Print()

The print() function prints the specified message to the screen, or other standard output device.

- The message can be a string, or any other object, the object will be converted into a string before written to the screen.

  Syntax

  ```
  print(value(s), sep= ' ', end = '\n', file=file, flush=flush)
  ```

Parameters:

- **value(s) :** Any value. Will be converted to string before printed
- **sep= ‘separator’ :** (Optional) Specify how to separate the objects, if there is more than one.
- **end= ‘end’:** (Optional) Specify what to print at the end.Default : ‘\n’
- **file :** (Optional) An object with a write method. Default :sys.stdout
- **flush :** (Optional) A Boolean, specifying if the output is flushed (True) or buffered (False). Default: False

Example : 1 print("Hello", "how are you?")

2. x = ("apple", "banana", "cherry")
   print(x)

3. print("Hello", "how are you?", sep="---")

String Literals

String literals in python’s print statement are primarily used to format or design how a specific string appears when printed using the print() function.

- **\n :** This string literal is used to add a new blank line while printing a statement.
- **“” :** An empty quote (“”) is used to print an empty line.

```
print("Hi I am studying \n in chitkara university.")
```

input ()

input (): This function first takes the input from the user and converts it into a string. The type of the returned object always will be <type ‘str’>. It does not evaluate the expression it just returns the complete statement as String. For example, Python provides a built-in function called input which takes the input from the user. When the input function is called it stops the program and waits for the user’s input. When the user presses enter, the program resumes and returns what the user typed.

```
val = input("Enter your value: ")
print(val)
```

Exponentiation Operator **

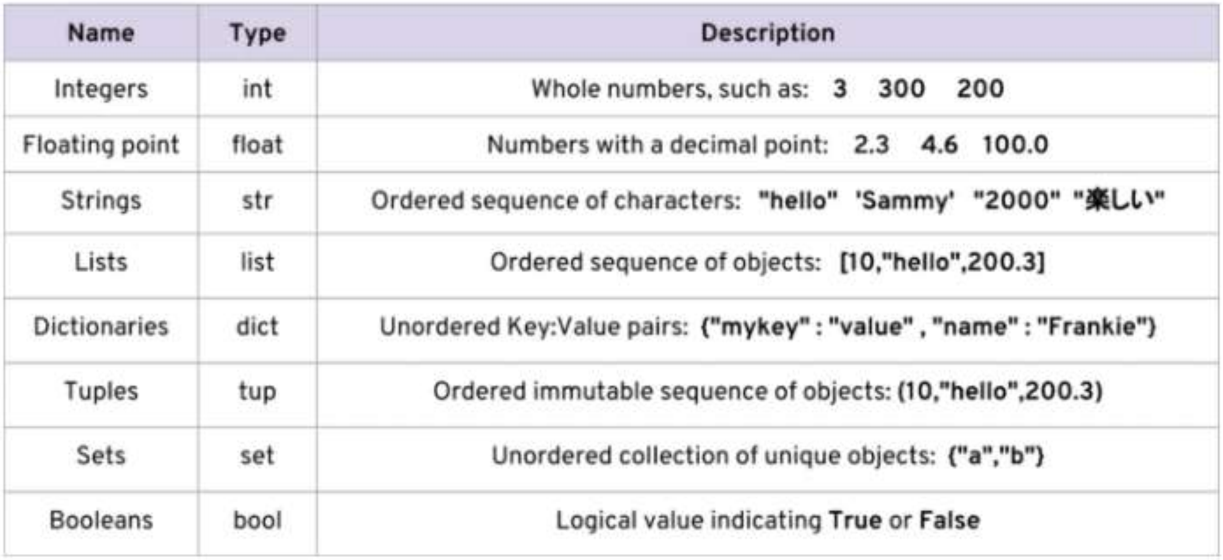
```
a = 2
b = 5

c = a**b
print(c)

z = 2 * (4 ** 2) + 3 * (4 ** 2 - 10)
print(z)
```

Data Types in Python:

- **Numbers:** The Number data type is used to stores numeric values.
- **String:** The string data type is used to stores the sequence of characters.
- **Tuple:** Tuple data type is used to stores a collection of different data types of elements, and it is immutable.
- **List:** List data type is used to store the collection of different data types of elements, and it is mutable.
- **Set:** Set data type is used to store different data types of elements; it is mutable and stores unique elements.
- **Dictionary:** Dictionary data type is used to store a collection of different data types of elements in the form of key-value pairs; it is mutable and stores the unique key.



Name	Type	Description
Integers	int	Whole numbers, such as: **3 300 200**
Floating point	float	Numbers with a decimal point: **2.3 4.6 100.0**
Strings	str	Ordered sequence of characters: **"hello" 'Sammy' "2000" "楽しい"**
Lists	list	Ordered sequence of objects: **[10,"hello",200.3]**
Dictionaries	dict	Unordered Key:Value pairs: **{"mykey" : "value" , "name" : "Frankie"}**
Tuples	tup	Ordered immutable sequence of objects: **(10,"hello",200.3)**
Sets	set	Unordered collection of unique objects: **{"a","b"}**
Booleans	bool	Logical value indicating **True** or **False**

Rules for Python variables:

- 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 (age, Age and AGE are three different variables)