# Python

May 22, 2021

**Day 1 -** **{**Printing, Commenting, Debugging, String Manipulation and Variables**}**

**Printing**

**Def:** 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 will be converted into a string before written to the screen.

**Syntax**

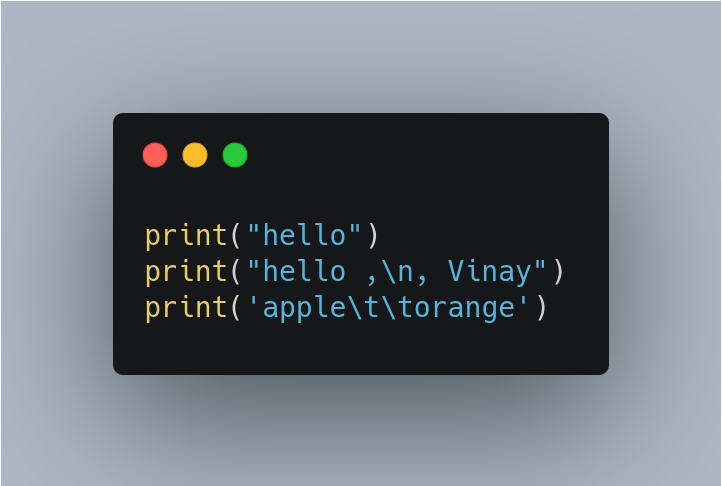
|  |
| --- |
| Print (Object(s), sep=separator, end=end, file=file, flush=flush) |

**Parameter Values**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Object(s) | Any object, and as many as you like. Will be converted to string before printed. |
| sep=’separator’ | Optional. Specify how to separate the objects, if there is more than one. Default is’ ’. |
| end=’end’ | Optional. Specify what to print at the end. Default is ‘/n’ (line feed). |
| file | Optional. An object with a write method. Default is sys. stdout. |
| flush | Optional. A Boolean, specifying if the object is flushed (True) or buffered (False). Default is False. |

For example, of the print input command

**Input**



**Output**



**Note:** Parentheses, you going to mention after the print **() function.** Inside the parentheses we also mentioned that **quotation mark** or **double quotation print (“”).** Furthermore, inside this one we are mentioning to print something, inside the text are what we calling **String print (“Hello World!”).**

**New Line using in print function:**

In above input images we witness the **\n** means that after this you want to print remaining string.

**ex:** print (“Hello World! \nHello World! \nHello World!”)

**Combine the two Strings:**

More about the string method we do want to add the output with the string in further main projects like below example.

**ex:** print (“Bunch of lists” +” Vinay”);

**Note:** While using the space in the editor we must follow some rules, so in further we mostly using the space a lot. Maybe in the if condition and other methods so that the space is commonly used in the Python.

**Input Function**

**Example**

**print (‘Enter your name:’)**

**x = input ()**

**print (‘Hello, ‘ + x)**

**Def**

The input () function allows user input.

**Syntax**

|  |
| --- |
| input (prompt) |

**Parameter Values**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| prompt | A String, representing a default message before the input. |

**Ex2**

Another method of using the Input function and concatenation inside the print function. After finishing this input, firstly it will executes the input function which was inside of the print function and secondly this input takes a string and stores and finally this store of the string will displayed in the print function along with the message.

print("Hello "+input("What is your name?\n: "))

**Note:** There is one site which will help you to understand the line of the process of the code. **Web:** [**https://thonny.org/**](https://thonny.org/)

**Comment**

For commenting we can use # for one line of comment and for more than two lines of comment we will use “”” This one “””.

**Ex:**

"""print("hello")

print("hello ,\n, Vinay")

print('apple\t\torange')"""

"""print("Hello World!\nHello World!\nHello World!")

print('Hello'+' Vinay')"""

# Input Function

**Additional Info**

In addition to input function and also from quiz I had learned that the length of the string. However I’m unable to add the string and the result of the input length, its because of input length is not but a int(integer). So I cleared this typo while looked into the error message, it says TypeError: can only concatenate str (not "int") to str.

**len() function**

**Def**

The len () function returns the number of items in an object.

When the object is a string, the len () function returns the number of characters in the string.

**Syntax**

|  |
| --- |
| len(object) |

**Parameter Values**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| object | Requires, An object. Must be a sequence or a collection. |

**Python Variables**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

**Assigning Values to Variables**

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

The operand to the left of the = operator is the name of the variable and the operand to the right of the = operator is the value stored in the variable. For example -

counter = 100 # An integer assignment

miles = 1000.0 # A floating point

name = "John" # A string

print counter

print miles

print name

**Note**

I’m unable to solve the challenge in the final quiz which is 1-4, below we can view the example. I’m unable to think out of the box as she said, moreover this code is little logical to solve it and I need to assign another variable to get the proper result.

#

a = input("a: ")

b = input("b: ")

#

####################################

#Write your code below this line

c=a

a=b

b=c

#Write your code above this line

####################################

#

print("a: " + a)

print("b: " + b)

**Output**

**a: 2**

**b: 5**

**a: 5**

**b: 2**

**Rules that one must follow while assigning to the variable**

1. We can assign the variable what ever we like to, but single letter would be confusing after 6 or 12 months when we are exploring to the python application we will get confused, example l=len(name).
2. The another best way to assign the variable is with underscore “\_” example “**user\_first\_name**”, but remember, spacing in python will be really sensitive for example “**user first name**” like this the python will be trow an error.
3. We can also assign the variable with numbers like lenght1 and lenght2 so on, but python will get confused after assigning the number after a variable for example 1lenght, 2lenght so on.
4. Privileged words we not going too assigned as a variable for example input = input() or print = print().

**:: The End ::**

**Day - 2**

Data Types, Numbers, Operations, Type Conversion, f-Strings

**Built-in Data Types**

In programming, data type is an important concept.

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

Text Type: str

Numeric Types: int, float, complex

Sequence Types: list, tuple, range

Mapping Type: dict

Set Types: set, frozenset

Boolean Type: bool

Binary Types: bytes, bytearray, memoryview

**Getting the Data Type**

You can get the data type of any object by using the type () function

**Example**

Print the data type of the variable x:

x = 5  
print (type(x))

**Setting the Data Type**

In Python, the data type is set when you assign a value to a variable:

Moreover, in this table I’m also declaring the setting the specific data type

|  |  |  |
| --- | --- | --- |
| **Example** | **Data Type** | **Specific Type** |
| x = "Hello World" | Str | Str (x) |
| x = 20 | Int | Int (x) |
| x = 20.5 | Float | Float (x) |
| x = 1j | Complex | Complex (x) |
| x = ["apple", "banana", "cherry"] | List | List (x) |
| x = ("apple", "banana", "cherry") | Tuple | Typle (x) |
| x = range(6) | Range | Range (x) |
| x = {"name" : "John", "age" : 36} | Dict | Dict (x) |
| x = {"apple", "banana", "cherry"} | Set | Set (x) |
| x = frozenset({"apple", "banana", "cherry"}) | Frozenset | Frozenset (x) |
| x = True | Bool | Bool (x) |
| x = b"Hello" | Bytes | Bytes (x) |
| x = bytearray(5) | Bytearray | Bytearray (x) |
| x = memoryview(bytes(5)) | memoryview | Memoryview (x) |

**Python String**

**Strings:** Strings in python are surrounded by either single quotation marks, or double quotation marks.

‘hello’ is the same as “hello”

You can display a string literal with the print () function.

print (“Hello”)

print (‘Hello’)

**Assign String to a Variable**

Assigning a string to a variable is done with the variable name followed by an equal sign and the string:

a = “Hello”

print (a)

**Multiline Strings**

You can assign a multiline string to a variable by using three quotes:

**Note:** You can use three double quotes:

a = """Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua."""  
print(a)

Or three single quotes:

a = '''Lorem ipsum dolor sit amet,  
consectetur adipiscing elit,  
sed do eiusmod tempor incididunt  
ut labore et dolore magna aliqua.'''  
print(a)

**Note:** in the result, the line breaks are inserted at the same position as in the code.

**String of Arrays**

Like many other popular programming languages, strings in Python are arrays of bytes representing Unicode characters.

However, Python does not have a character data type, a single character is simply a string with a length of 1.

Square brackets can be used to access elements of the string.

**Note:** Get the character at position 1 (remember that the first character has the position 0):

a = "Hello, World!"  
print(a[1])

**Looping Through a String**

Since strings are arrays, we can loop through the characters in a string, with a for loop.