

Chapter 1 : Modules, Comments & pip

Let's write our very first python program.
Create a file called `hello.py` and paste the below code in it.

`print ("Hello World")` → print is a function (More later)

Execute this file (by file) by typing `python hello.py` and you will see Hello World printed on the screen.

Modules

A module is a file containing code written by somebody else (usually) which can be imported and used in our programs.

Pip

Pip is the package manager for python. You can use pip to install a module on your system.

→ pip install flask installs flask module.

Types of modules

There are two types of modules in Python

- 1> Built in modules → Pre installed in python
- 2> External modules → Need to install using pip.

Some examples of built in modules are os, abc, etc.
Some examples of external modules are tensorflow, flask etc.

Using Python as a Calculator

We can use python as a calculator by typing "python" + ↵ on the terminal

↳ This opens REPL

or Read Evaluate Print Loop

Comments

Comments are used to write something which the programmer does not want to execute.

↳ Can be used to mark author name, date etc.

Types of Comments

There are two types of comments in Python

1. Single line comments → Written using #
2. Multi line comments → Written using """ command """

Chapter 2 - Variables and Data types

A variable is the name given to a memory location in a program. For example

a = 30

b = "Harry"

c = 71.22

→ Variables = Container to store a value.

→ Keywords = Reserved words in Python
Identifiers = class/function/variable name

Data Types

Primarily there are following data types in Python

1. Integers

2. Floating point numbers

3. Strings

4. Booleans

5. None

Python is a fantastic language that automatically identifies the type of data for us.

a = 71

⇒ Identifies a as class int

b = 88.44

⇒ Identifies b as class <float>

name = "Harry"

⇒ Identifies name as class <str>

Rules for defining a Variable name → Also applies to other Identifiers

→ A variable name can contain alphabets, digits and underscores.

→ A variable name can only start with an alphabet and underscore.

→ A variable name can't start with a digit

→ No white space is allowed to be used inside a variable name.

Examples of a few variable names are :-
 harry, one8, Seven, _Seven etc.

Operators in Python

Following are some common operators in Python :

- 1, Arithmetic operators $\Rightarrow +, -, *, /$ etc.
- 2, Assignment operators $\Rightarrow =, +=, -=$ etc.
- 3, Comparison operators $\Rightarrow ==, >, \geq, <, \leq, !=$ etc.
- 4, Logical operators $\Rightarrow \text{and}, \text{or}, \text{not}$

`type()` function and Typecasting

`type` function is used to find the data type of a given variable in Python.

`a = 31`

`type(a) \Rightarrow class <int>`

`b = "31"`

`type(b) \Rightarrow class <str>`

A number can be converted into a String and vice versa (if possible)

There are many functions to convert one data type into another

`str(31) \Rightarrow "31"` \Rightarrow Integer to String Conversion

`int("32") \Rightarrow 32` \Rightarrow String to Integer Conversion

`float(32) \Rightarrow 32.0` \Rightarrow Integer to Float Conversion

... and so on

Here "31" is a string literal and 31 a numeric literal.

input() function

This function allows the user to take input from the keyboard as a string

`a = input("Enter name")` \Rightarrow If a is "harry", the user entered harry

It is important to note that \Rightarrow If a is "34" user the output of input is always user entered 34 a string (even if the number is entered)

Chapter 3 - Strings

String is a data type in Python.

String is a sequence of characters enclosed in quotes.

We can primarily, write a string in these three ways

1. Single quoted strings → $a = 'Harry'$
2. Double quoted strings → $b = "Harry"$
3. Triple quoted strings → $c = """Harry"""$

String Slicing

A String in Python can be sliced for getting a part of the string.

Consider the following string:

$\text{name} = "H\boxed{a}rry" \Rightarrow \text{length} = 5$

0 1 2 3 4
 (-5) (-4) (-3) (-2) (-1)

The index in a string starts from 0 to $(\text{length}-1)$ in Python. In order to slice a string, we use the following syntax:

$sl = \text{name}[\text{ind_start} : \text{ind_end}]$

first index included \rightarrow last index is not included

$sl[0 : 3]$ returns "Hai" → characters from 0 to 3

$sl[1 : 3]$ returns "ai" → characters from 1 to 3

Negative Indices : Negative Indices can also be used as shown in the figure above. -1 corresponds to the $(\text{length}-1)$ index, -2 to $(\text{length}-2)$

Slicing with skip value

We can provide a skip value as a part of our slice like this :

Word = "amazing"

word [1:6:2] → 'mzn'

Other advanced slicing techniques

Word = "amazing"

word [:7] → word [0:7] → 'amazing'

word [0:] → word [0:7] → 'amazing'

String functions

Some of the mostly used functions to perform operations on or manipulate strings are :

- 1> len() function → This function returns the length of the string

len ("Harry") → returns 5

- 2> string.endswith("rry") → This function tells whether the variable string ends with the string "rry" or not. if string is "Harry", it returns true for "rry" since Harry ends with rry

- 3> string.count("c") → Counts the total number of occurrence of any character

- 4> string.capitalize() → This function capitalizes the first character of a given string.

5. `String::find(word)` - This function finds a word and returns the index of first occurrence of that word in the string.

6. `String::replace(oldword, newword)` - This function replaces the oldword with newword in the entire string.

Escape Sequence Characters

Sequence of characters after backslash \ → Escape Seq characters

Escape sequence character comprises of more than one characters but represents one character when used within the strings.

Examples \n, \t, \\", \\ etc.
newline Tab Single quote → backslash.

Chapter 4 - Lists and Tuples

Python Lists are containers to store a set of values of any data type

```
friends = ["Apple", "Akash", "Rohan", 7, False]
```

↓
str()
↓
int()
↑
bool()

Can store value of any datatype

List Indexing

A List can be indexed just like a String

```
L1 = [7, 9, "Harry"]
```

$L1[0] \Rightarrow 7$

$L1[1] \Rightarrow 9$

$L1[70] \Rightarrow \text{Error}$

$L1[0:2] \Rightarrow [7, 9] \Rightarrow \text{List Slicing}$

List Methods

Consider the following list :

```
L1 = [1, 8, 7, 2, 21, 15]
```

1. $L1.\text{Sort}()$: updates the list to [1, 2, 7, 8, 15, 21]

2. $L1.\text{reverse}()$: updates the list to [15, 21, 2, 7, 8, 1]

3. $L1.\text{append}(8)$: adds 8 at the end of the list

4. $L1.\text{insert}(3, 8)$: This will add 8 at 3 index

5, `L1.pop(2)`: Will delete element at index 2 and return its value

6, `L1.remove(2)`: Will remove 2 from the list.

Tuples in Python

A tuple is an immutable data type in python.

↳ Cannot change

`a = ()` ⇒ Empty tuple

`a = (1,)` ⇒ Tuple with only one element needs a comma

`a = (1, 7, 2)` ⇒ Tuple with more than one element

Once defined, a tuple's elements can't be altered or manipulated.

Tuple methods

Consider the following tuple

`a = (1, 7, 2)`

1, `a.count(1)`: `a.count(1)` will return number of times 1 occurs in a.

2, `a.index(1)`: `a.index(1)` will return the index of first occurrence of 1 in a.