Day 1:

Points to be noted:

Python uses indentation to show a block structure unlike other languages which may use ({})

Eg: if(a<b) {

//do something; -> in C, C++, Java

}

While in Python

if(a<b):

#do something

Here the : (colon) represents start of the block and the indentation inside if loop means it’s inside if loop

**Printing Hello World** :

It’s simple just use the print function

>>>print(“Hello LTI”)

>>>Hello LTI

Here the print function tells the compiler to print. We can print anything inside print function, here we have used string. In python by default it prints in a new line. You can also specify new line like this >>>print(“Hello”, ”\n” ,”LTI”) which is like other programming languages.

If you want to print without the newline which is there by default, then you can give something like this

>>>print(“Hello ”,end=” “) – *notice the end keyword part*

>>>print(“LTI”)

>>> Hello LTI

You can give anything inside the end part, now we used a whitespace, so it does not end at whitespace.

Another parameter is sep >>>print(“Hello”, “LTI” , sep = “-“) o/p -> Hello-LTI

From this we can infer that comma spaced values inside print function, they’re defaulted by a single whitespace. For escape characters please refer this link: <https://www.w3schools.com/python/gloss_python_escape_characters.asp>

**Data Types:**

Numeric data types :

* Supports large integers by default.
* Sample integer >>> var = 10 o/p -> 10 is considered as Int and stored in var. In python the right part of the equal to defines the type of data not the variable.
* For any float, you can declare as >>>var = 10.2 now this var variable has become float/decimal.

For string data types:

Simply declare >>>var = “pineapple” or >>>var = ‘pineapple’

Suppose you want the type of a variable then use type function

Eg: >>>type(var) o/p -> <class 'str'>

If you want a variable to be Null or None, then we can simply use

>>>var = None

**Operators:**

**Arithmetic operators:**

+ -> for addition >>>var = 5 + 5

- -> for subtraction >>>var = 5-5

\* -> for multiplication >>> var = 5\*2

/ -> for float division >>> var = 10/2 o/p-> 5.0

// -> for Int division >>>var = 10//2 o/p -> 5

% -> returns remainder >>>var = 5%2 o/p -> 1

\*\* -> Power of a number >>>var = 5\*\*2 o/p -> 25

**Comparison operators:** (returns Boolean)

> - greater than >>>(var>10)

< - less than >>>(var<10)

<= - less than or equal to (var<=10)

>= - greater than or equal to (var>=10)

!= - Not equal to (var!=10)

== - equals (var==10)

**Logical operators:**

and - True if both true (var>10 and var<20)

or – True if either one is true (var>10 or var<20)

not – True if false (var not None)

**Bit-wise operators:**

& - bit wise AND (var & 1010)

| - bit wise OR (var | 1010)

^ - bit wise XOR (var ^ 1010)

**Strings:**

Python strings are immutable that is they can’t be changed after they are created. The characters of a string can be indexed like an array

>>>var = “Good”

>>>var[0]

>>>G

Python uses zero-based indexing and when index is out of bounds then it throws an error. We can find the length of the string using len function

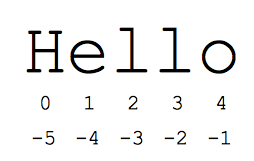
>>>len(var)

>>>4

*Tip : Don’t use variable name as len*

Another useful property of strings is string slicing. It is used to return the subpart of your string . It is also present in list/arrays . The below image specifies the indices of string from left and right.

Eg: >>>s = “Hello”



>>>s[1:4] returns “ell” that is start at index 1 and up to index 4 (not inclusive)

>>>s[2:3] returns “l”

>>>s[:2] returns “He” if start index is not specified it considers it at as s[0:2]

>>>s[2:] returns “llo” similarly as above

Negative index slicing is the same but acts in reverse

>>>s[-4:-1] returns “ell” you can see with image for easy understanding.

>>>s[-2:] returns “lo” starts at -2 goes till end

>>>s[-1] returns the last character of the string.

There are many predefined string methods such as:

>>>s.lower() – to convert to lowercase

>>>s.split(‘delimiter’) – to split strings to arrays

>>>s.isdigit() – Returns true if all characters are numbers

The % operator in string:

>>>st = “%d candles where there, meaning I celebrated my %s anniversary.” % (5, “fifth”)

>>>5 candles where there, meaning I celebrated my fifth anniversary.

The elements after % operator are filled respectively in those placeholders (%d, %s etc). %d denotes it’s an integer similarly for others.

**Getting input:**

Getting input from users in python is pretty easy it uses the input() method

>>>ip = input()

if you’re getting array as input then

>>>ip = list(map(int,input().split()) - *don’t worry if this doesn’t make sense, we will cover this in later sessions.*

**Type conversion:**

This is one of the basic and important ones to know in python. Suppose you want to convert string to int or int to string (for any purpose) these become quite handy.

Converting str to int

>>>S = “5”

>>>S = int(S)

>>>S = str(S) #similarly for string.

**Other useful concepts:**

# - used for commenting in python

Suppose you’re code is to big for single line and reduces readability then we can use backslash \

>>>st = “We'll walk this road together, through the storm Whatever weather, cold or warm Just letting you know that, you're not alone”

>>>st = “We'll walk this road together, through the storm \  
 Whatever weather, cold or warm \  
 Just letting you know that, you're not alone”

Assignments for day 1:

1. Print the following image  it contains 6 equal to symbols and 3 greater than and less than symbols. Try not to hard code. Try to achieve only using one print statement.
2. Given the following string s of length n, which may represent different email ids of gmail , write a program to extract their id.

rossumvan@gmail.com - rossumvan (output)

stevejobs@gmail.com – stevejobs (output)

1. Write a program to remove “$” signs from the given string . These $ signs only occur at the start and end of string

$$$password$$$ - password (output)

Password123$$$ - Password123 (output)

1. Given a number n, print 2\*n but multiplication and addition operators are not allowed

If n = 10 output = 20 (*achieve 20 without multiplication and addition operators*)

If n = 23 output = 46

(*try bitwise operators*)

1. Given a decimal number , write a program to round off the decimal to nearest integer.

N = 16.5 , output = 17

N = 16.556 , output = 16.56

*Tip : Look for already existing methods*