Beginner:

[ ]Introduction of python

- [ ]why python and the usage of python

[ ]Installation python & vs code

[ ]Data type

primitive integer string Boolean float

Non primitive list Arrays files Tuples Sets

-don’t need to declare type

[ ]Operations + - * / % **(exponential) //(floor division)

[ ]Output print

[ ]String " " or ' ' print \' \" \\ \n (next line) \t

Operation 'hi ' * 3 => hi hi hi "1" + "2" => "12

[ ]Variable

-can’t start with numbers can use alphabet & underscore & numbers

-case sensitive

[ ]Input int(input()) float(input()) input()=> str

x = Input('comment')

-import & python name spaces

[ ]Type conversation

[ ]In-place operations-- ++ =* =/ =× =+

== [ ]comparison

= assignment=! >= <= > <

[ ]if Condition :

Statement

else:

Statement

-indentation - nested

-Can have 1 else for every if but unlimited if

-elif (else if) can have else

[ ]Logical operations and or not

[ ]Bitwise operations & | != ^(XOR)

[ ]Loop while, for, for in(foreach)

python doesn't have do while

diff for & while not/knowing iteration

break continue

[ ]List[,]

-support diff type in the same list

-starts iterating from 0

-append(x) insert (x,index) extend=+ ([])

-support negative number (counting from end)

-2Dlist

-string as list x="hello" x[1]=>e space has index

-list operations* +

-slices [start index, end index] includes start but not the end x=[2,5,3,1,6,8,9] x[1:4] 5 3 1

x[,2] 2 5 x[3,] 3 1 6 x[::2] 2 3 6 x[1:6:3] 5 6

list [1,2,3,4,5] =

list[-1::] [5] list[:-1:] [1,2,3,4] list[::-1[5,4,3,2,1] [

list[-1::] --> list only with last element

list[:-1:] --> all list except last element

list[::-1] --> list in backwards order

[ ]Range(start, end, steps)

-range(9) 0,1,2,3,4,5,6,7,8 range(3,8) 3,4,5,6,7

[ ]Function str(10) print("hi")

def func_name(parameter1,parameter2)

-return & after return doesn’t execute

-diff between parameter & argument

[ ]List method

-Len() length of list

append() Adds an element at the end of the list

clear() Removes all the elements from the list copy() Returns a copy of the list

count() Returns the number of elements with the specified value -list.count(item)

extend() Add the elements of a list (or any iterable), to the end of the current list

index() Returns the index of the first element with the specified value

insert() Adds an element at the specified position

max(list): Returns the maximum value.

min(list): Returns the minimum value.

pop() Removes the element at the specified position

remove() Removes the first item with the specified value

reverse() Reverses the order of the list

sort() Sorts the list

[ ]String function

-print("{x} {y}.format(x,y"(

-print("{0}{1}.format("hello","world("

.join .replace .startswith .endswith .lower .upper .split

[ ]Comment # octothorpe

[ ]Docstrings (documentation strings) .

"""are similar to comments, in that they're designed to explain code. But! they're more specific and have a different syntax"""

-Docstring of a function can be accessed by doc attribute.

-print(max.doc)

Intermediate:

[ ] Keywords and Identifiers

[ ] Statements

conditional list if break continue assignment loop while for compound indentation iteration assert

-A compound statement basically is a group of statements executed as a unit. It is made up of:

a header line (followed by a colon :), and

a body containing a sequence of statements at the same level of indentation.

[ ] Python pass

-empty statement(do nothing)

[ ] Python Namespaces

-a collection of currently defined symbolic names along with information about the object that each name references. You can think of a namespace as a dictionary in which the keys are the object names, and the values are the objects themselves.

[ ] Python Recursion

-base case

-factorial

-indirect (odd even example)

[ ] Anonymous Function

-lambada

-filter and map

[ ] Python function

-generator

-decorator

-*args is tuple **kwargs is a dictionary

[ ] Global, Local and Nonlocal

[ ] Python Global Keyword

[ ] Python Modules

-a file containing Python statements and definitions .py

-A python “module” consists of a unit namespace, with the locally extracted variables.

-Python there also are built-in modules, such as sys, which are written in C

[ ] Python Package

-A package is a folder with a bunch of modules

-To be considered a package, that folder must contain an init.py file

[ ] Python List

-comprehension

[ ] Python Tuple

-immutable

-words = ("spam", "eggs", "sausages") word[0]

-word[1]="milk" error

-When a data type has parentheses around it doesn't mean it is a Tuple:

Not Tuples: (5) ("string")

To create a tuple with single element, place comma at the end before the closing parentheses or you can also create one without parentheses as long as there is a comma: (5,) ("string" ,) 42, "string",

However, an empty parenthesis is an empty tuple. tup = ( )

And they are smaller in size than lists.

And we actually do not need parentheses to create a tuple.

names = "Saniya","Dan"

But we can mute through a tuple by this way.

my_tuple = (1,2,3,4,5,6)

my_tuple = list(my_tuple)

my_tuple[1] = "Two"

print(tuple(my_tuple))

-unpacking

numbers = (1, 2, 3)

a, b, c = numbers

a,b = [1,2]

a, b = b, a

-a, b, *c, d = [1, 2, 3, 4, 5, 6, 7, 8, 9]

[ ] Python Set

-mutable

-unordered (can't be indexed)

-faster support of in than list

num_set = {1, 2, 3, 4, 5}

-Set functions add() remove() len() discard()

-sets for math sets

a.union(b)   a|b

a.difference(b) a-b

a.symmetric_difference(b) a^b

Intersection a&b

[ ] Python Dictionary

-map(key,value) & mutable

-can use strings, integers, booleans, and any other immutable type as dictionary keys.

-Immutable objects are those that can't be changed

-error => bad_dict = { [1, 2, 3]: "one two three", }

-ages = {"Dave": 24, "Mary": 42, "John": 58}

-dict func in not in len() get(key) get(key,"key not found)

*

Lists --> [ ]

Tuples --> ( )

Dictionary --> { }

Set --> {}