**Python training Notes:**

**Course Name:** **SCRIPT 307: Basic Python**

**This is Part 1 of the whole training in the duration 17 to 31 July**

**This will be followed with next Part 2 session for Intermediate Python topics in the month of August.**

**Day 4: 20 Jul 2018 - Friday (2 Hrs Session)**

**Expectation Setting ASL (Assisted Self-Learning) 2Hrs session daily**

**And then do self-study and hands on assignments from below learning course link and also the assignments given below here in this document:**

<https://knowledgecenter.persistent.co.in/ViewCourse/pmoc>

***Please visit the following URL to view the collaborative learning group***

<https://persistentuniversity.persistent.co.in/CollaborativeLearningGroup/view.aspx?SkillId=9144>

**Topics Covered:**

Dictionary – Inner list, inner dictionary

Functions for Dictionary

List Comprehension

Dictionary Comprehension

Functions

**\*\*\*\*\*To Do for Day 4:**

Nugget 1 : Introduction to Python & Python Fundamentals

Nugget 2 : Python Basics

Nugget 3 : Python Control Structures

Nugget 4 : Functions & Modules

Subjective Assignment for Nugget 1 to 3 : Only for self Practice

522

1. Complete reading these 4 Nuggets from <https://knowledgecenter.persistent.co.in/ViewCourse/pmoc>

2. Please execute all codes in these 4 Nuggets

3. Start solving assignment at the end of Nuggets

**Try Below Codes:**

**11\_Dictionery\_Demo.py**

dictionaryOne = {}

dictionaryTwo = {'course': 'moc','name': 'python', 'name':'Perl'}

print "dictionaryTwo = ", dictionaryTwo #dictionaryTwo = {'course': 'moc', 'name': 'python'}

print dictionaryOne, dictionaryTwo #tuple of 2 dictionary

print "name = ",dictionaryTwo['name'] #python

print dictionaryTwo.has\_key('name') #True

#dictionaryTwo.

if(dictionaryTwo['name']): print "Name contains value"

else: print "No Value for name"

dictionaryTwo['course'] ='Selfy'

#add new entry

dictionaryTwo['id']=100 #craetes key-value pair 'id':100

print dictionaryTwo

print "=========================================================="

#delete pair

del dictionaryTwo['id']

print "after deletion = ", dictionaryTwo

dictionaryTwo.clear()

print "after claering = ", dictionaryTwo

dictionaryFour = {'name': 'python', 'course': 'moc'}

del dictionaryFour

#dictionaryFour # not accissible as deleted, NameError if try to access

dictionaryTwo = {'name': 'python', 'course': 'moc'}

print dictionaryTwo.items()

#[('course', 'moc'), ('name', 'python')] list of tuple key-value pairs

print dictionaryTwo.keys() #list of keys

print dictionaryTwo.values() #list values

print "-----------------------------------------"

k1 = dictionaryTwo.keys() # ['course', 'name']

k1.sort()

for i in k1: #generic loop to process dictionary ['course', 'name']

print i ,"\t = ",dictionaryTwo[i]

dictionaryTwo[i]= dictionaryTwo[i].upper()

print "-----------------------------------------"

print "Updated dictionaryTwo = ", dictionaryTwo

print "-----------------------------------------"

empData ={'3a':50000,'1a':30000,'2a':40000}

#increment the salary of every person by 5000 and then display updated emp data

print "Original Emp Data = ",empData

k1 = empData.keys()

k1.sort()

totalSal =0

for i in k1:

print "Emp Id : ", i, "\t Salary = ", empData[i]

totalSal +=empData[i]

empData[i]+=10000

print "Updated Emp Data = ",empData

print "Total Sal = ",totalSal

print "-----------------------------------------"

#'5a':75000 add this pair in empData

empData['6a'] = 66000

empData.update({'5a':75000})

str1 = raw\_input("Enter emp ID:salaray in a single line: ")#"7a :77000"

l1 = str1.split(":") #l1 is a list of [id,sal]

print "list1 = ", l1

empData[l1[0]] = l1[1]

print "Updated Emp Data = ",empData

print "-----------------------------------------"

**List Comprehension**

"""

get a new list from a existing list by performing some operatio on

every element of a list

module in Python : a file - contains re usable code - functions/ Classes

"""

import math

l1 = [1,2,3,4,5]

double\_list = []

for i in l1:

double\_list.append(i\*2)

print "Original list = ", l1

print "Double list = ", double\_list

print "-----------------------------------------------------"

#alternate solution - List Comprehension

double\_list1 = []

double\_list1=[i\*2 for i in l1] #[2,4,6,8,10]

print "Original list = ", l1

print "Double list = ", double\_list1

print "-----------------------------------------------------"

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

squared\_elements =[]

squared\_elements = [i\*\*2 for i in nums]

print "Squared list = ", squared\_elements

print "-----------------------------------------------------"

squared\_root\_elements=[]

squared\_root\_elements = [math.sqrt(i) for i in squared\_elements]

print "Square rooted elemnts list = ", squared\_root\_elements

squared\_root\_elements\_int = [int(math.sqrt(i)) for i in squared\_elements]

print "Square rooted elemnts list in int= ", squared\_root\_elements\_int

print "-----------------------------------------------------"

#print "Square root of 25 = ", math.sqrt(25)

num2 = [1,2,3,4,5,77,77,88,99]

#get a list of squared elements which are less than 10 in num2

squared1 =[]

squared1 = [i\*\*2 for i in num2 if i <10]

print "Squared elements = ", squared1

**Dictionary Comprehension**

#Dictionary Comprehension

list1 = [1,2,3]

dict1 = {i : i\*\*2 for i in list1} #{1:1, 2:4, 3:9}

print "Orginal list = ", list1

print "Dictionary ", dict1

print "--------------------------------------------"

dict1 = {i : i\*\*2 for i in list1 if i<3} #{1:1, 2:4}

print "Orginal list = ", list1

print "Dictionary ", dict1

=====================================

**Functions**

**1\_Func1.py**

def func2(): #function definations

print "In Func2...."

def func1():

print "In func1....."

func2()

return "Hello"

'''

print "Start of the script"

func2() #function call

print "END!!"

'''

ret = func1() #calling/invoking a function

print "retr value of function call = ", ret #default return value of a function is None

print "END!!!!!"

**2\_Func1\_return.py**

def func2():

print "In Func2...."

def func1():

print "In func1....."

func2()

#return "SUCCESS" #return string

#return ['xyz',1,3.145] #return LIST

result = func1()

print result

**3\_Func1\_parameter.py**

#Keyword parameters

def hello(name): #catching place -formal argument name="Python"

"""Parameter passing"""

print "Hello "+name

result = hello() #TypeError: hello() takes exactly 1 argument (0 given)

#result = hello("Python") #invoking place

print result

**4\_Func1\_parameter\_by\_name.py**

#2)parameter passing by names

def greet(title, name): #keyword arguments - formal/catching arguments

print "Hello "+title+ " "+name

#greet("ABC", "Mr.")

greet(name="ABC", title="Mr.") #sequence can be anything with these named parameter

#greet(name=[1,2,3,4])

**5\_Func1\_parameter\_dafault.py**

#3) fixed/default value parameter

def calculate\_tax(cost, rate=0.2): #1)cost = 1000 rate=0.2 2)cost = 1000 rate=0.5

return cost+(cost\*rate)

print "Tax with Default rate 0.2 = ",calculate\_tax(1000) #default rate 0.2 will be considered

print "Tax with rate 0.5 passed value = ",calculate\_tax(1000,0.5) #passes rate=0.5 will be considered

**5\_Func1\_parameter\_by\_name.py**

#parameter by name: Keyword arguments

def greet(title, name):

print "Hello "+title+ " "+name

print "Lower case string = ",name.lower()

greet("Sangita", "Mrs.") ##Hello Sangita Mrs.

greet(name="ABC", title="Mr.") #keyword arguments passing to a function call

#greet(title="Mr.", "NIL") # "NIL" as normal parameter (non-keyword argument)---this can not be passed after keyword argument

#Syntax error

#greet("NIL", title="Mr.") #TypeError: greet() got multiple values for keyword argument 'title'

#greet("Mr", name="NIL", value =100)#Syntax error

**Assignments to do:**

Q1. Accept 2 numbers from keyboard. Pass these as keyworded arguments and let function return the addition answer.

Q2. Covert all words from a list to upper case using List comprehension

Save the solutions in a folder: **Assignments\Day4**

**Assignments\Day4 --🡪**

**Q1.py**