**Python training Notes:**

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

**Day 2: 12 Sep 2017**

**Expectation Setting : ASL (Assisted Self-Learning) 3Hrs session daily**

**And then do self-study and hands on assignments form below learning course link:**

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

**Micro Learning Online link :** <https://persistentuniversity.persistent.co.in/microlearning/Course/837>

Welcome to Micro Learning!!

You can now complete a course by learning small units at a time, i.e. learning Micro Nuggets. Take out 15 minutes time daily and complete one micro nugget and a short quiz. Click on Complete button to ensure the completion and to get access to next Micro Nugget on next day.

Use the link given above to visit the Micro Learning Site to view the next micro nugget.

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

<https://persistentuniversity.persistent.co.in/CollaborativeLearningGroup/View.aspx?SkillID=8451>

s=’ABC’

revStr = s[::-1] #reversed string

List

AList = [‘a’,1,2,3]

AList[-1] #o/p = 3

List Tuple

Set

listOne =[1,2,3]

s = set(listOne)

OR

S2 = {1,2,3,4}

S3={} # empty dictionary

Error

TypeError

NameError

IndexError

KeyError : trying to access key that does not exists in disctionary, gives KeyError

SyntaxError

**\*\*\*\*\*To Do for Day2:**

Nugget 1 : Introduction to Python & Python Fundamentals

Nugget 2 : Python Basics

Nugget 3 : Python Control Structures

663

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

1. Complete reading above Nuggets

2. Please excute all codes in these Nuggets

3. Start solving assignment at the end of Nuggets

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

**Try Below Codes:**

**4\_user\_input.py**

#keyborad input

x = raw\_input("Please enter ur name :") #input is "" string str = "55" num = int(str)

print "Hello ", x

print "Welcome to Python"

print 5\*x

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

y = input("Enter your name") #input is not "" string

print "Hello", y # error, so take input as "ABC" it will work use input for numeric input

**6\_List\_Tuple.py**

#List data tuple data

#list ---->mutable

listOne= [123, 'abc', 4.56]

print listOne

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

#Tuple --->immutable

tupleOne= (123, 'abc', 4.56)

print tupleOne

#inner list

listTwo= [4.56, ['inner', 'list']]

print listTwo

#inner Tuple

tupleTwo = (123, 'abc', 4.56, ['inner', 'tuple'], 7-9j,[10,20])#tuple having one of the elements as mutable list

print tupleTwo

print "First element of list = ", listOne[0]

print "First element of tuple = ", tupleOne[0]

print listTwo[0:1]

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

listFour= [123, 'abc', 4.56, ['inner', 'list']]

print "3rd index inner list = ", listFour [3]

print "3rd index inner list = ", listFour [6] #run time error

print "first index element of inner list = " , listFour [3][1]

**7\_List\_Update.py**

#Updating List data:

#Example 1: Replacing a value in List

#append insert remove pop extend reverse

listOne=[100,200,'hello']

print listOne

listOne[2] =200

print listOne

print "last element = ",listOne[-1]

listOne.append(500)

print listOne

listThree= [123, 'abc', 4.56]

listThree.insert(2,'c')

print listThree

listThree.remove(4.56)

print listThree

#pop

listSix= [123, 'abc', 4.56]

print listSix.pop()

print "Modified List =", listSix

listSix.pop(1)

#listSix.pop(5) run time error IndexError

print "Modified List =", listSix

listTwo= [123, 'abc', 4.56]

print 'Original list = ', listTwo

listTwo.reverse()

print "reversed list =", listTwo

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

listOne= [123, 'abc', 4.56]

listOne.append([1, 2])

print "Extended list = ",listOne

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

print 'abc' in listTwo

print 'abc' not in listTwo

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

number\_list = [43, -1.23, -2]

string\_list = ['hello', 'world']

print string\_list + number\_list

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

listX =[10,0,44,55,77]

print "Original X list = ",listX

listX.sort()

print "Sorted X list = ",listX

**8\_Tuple\_del.py**

"""Tuple example"""

tupleOne= (123, 'abc', 4.56) #tuple defination tupleOne

tupleTwo= (789, 'def', 2.24) #tuple defination tupleTwo

tupleThird= tupleOne[0], tupleTwo[1] #Tuples are immutable which means they cannot be updated or change values of

#tuple values.But they allow to take portions of Tuples to create a new Tuple.

print "tupleOne = ",tupleOne #tupleOne = (123, 'abc', 4.56)

print "tupleTwo = ",tupleTwo #tupleTwo = (789, 'def', 2.24)

print "New tupleThird = ",tupleThird #New tupleThird = (123, 'def')

del tupleThird #Removing individual tuple elements is not possible though it is possible to

#delete an entire tuple with del function

#print tupleThird #run time error

**9\_ListDeepCopy.py**

import copy

listOne = [[1, 'hello'],[2, 'world']]

newlist =listOne #shallow copy

print "listOne = ", listOne #listOne = [[1, 'hello'], [2, 'world']]

print "newlist = ", newlist #newlist = [[1, 'hello'], [2, 'world']]

print "id of listOne = ", id(listOne) #id of listOne = 50857416

print "id of newlist = ", id(newlist) #id of newlist = 50857416

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

listTwo = copy.deepcopy(listOne) #deepcopy() method of copy module used to obtain deep copy of the object

#creating a new container but containing references to completely new copies (references)

print "listTwo = ", listTwo #listTwo = [[1, 'hello'], [2, 'world']]

print "id of listOne = ", id(listOne) #id of listOne = 50857416

print "id of listTwo = ", id(listTwo) #id of newlist = 50529736

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

listOne.append(500)

print "Changed listOne =", listOne #Changed listOne = [[1, 'hello'], [2, 'world'], 500]

print "UnChanged listTwo =", listTwo #UnChanged listTwo = [[1, 'hello'], [2, 'world']]

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

tupleTwo = tuple(listTwo) #you to create a tuple from a existing list by using tuple() function

print "Tuple Two = ",tupleTwo

print "Changed listTwo =", listTwo

l1 = list(tupleTwo)

print l1

**10\_Set\_demo.py**

#set is unordered collection of elements

listA=[1,2,3]

#create a set

s= set(listA)

print "Set = ", s

s1 ={1,2,3,3} #unique elements 1,2,3 taken

print "s1 = ", s1

print "One element = ", s1[0]

#set union

setOne = set([1, 2, 3])

setTwo = set([3, 4, 5])

print setOne | setTwo #returns me other set with unique elements set([1,2,3,4,5])

#set intersection

print setOne & setTwo #returns me other set with onlyu common elements set([3])

# iter.py

str = "formidable" #str[0]

for i in str: #for loop traverses all elements in a sequence

print i,

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

it = iter(str) #iter is predenfined function

print it.next()

print it.next()

print it.next()

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

print list(it)

'''

#set

s ={1,2,3,4,5} #not sequence type, unique elements

OR

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

s1= set(l1)

it = iter(s) #iterable object

**11\_Dictionery\_Demo.py**

dictionaryOne = {}

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

print dictionaryOne, dictionaryTwo

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

print dictionaryTwo.has\_key('name')

#add new entry

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

print dictionaryTwo

#delete pair

del dictionaryTwo['id']

print "after deletion = ", dictionaryTwo

dictionaryTwo.clear()

print "after claering = ", dictionaryTwo

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

del dictionaryFour

#sprint 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 "-----------------------------------------"

for i in dictionaryTwo.keys():

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

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

**12\_if\_demo.py**

#Example : if loop

a=20

b=20

if (a<b):print "a is less than b"

elif(a>b):

print "a is greater than b"

else:

print "a is less than b"

12\_while\_demo.py

#Example: while loop

count = 0

while (count < 5):

print 'the index is:', count

count = count + 1

print "END!!!!"

**14\_List\_Comprehension.py**

#List Comprehension

listOne=[1,2,3]

new\_listOne =[x\*2 for x in listOne]

#comprehenstion to operate on every element

print "list one =", listOne #[1,2,3]

print "new list =", new\_listOne #[2,4,6]

str ="abc"

print str.upper()

words =["abc","xyz", "lmn","aaaa","bbb"]

upperwords =[i.upper() for i in words]

print "Original words = ",words

print "Upper words =",upperwords

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

square\_list = [x\*\*2 for x in listOne]

print "Square list =",square\_list

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

#dictionary comprehension

dictionaryOne = [1, 2, 3, 4] #it is a list

dictionaryTwo = {x: x \* x for x in dictionaryOne }

print dictionaryTwo

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

#d1 ={'a1':100,'b1':200}

"""

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

revB = [reverse a]

print "Rev B= ", revB

for i in listOne:

listTwo.append(i\*\*2)

"""

**13\_for\_demo.py**

#for loop

x = [1.0, 2.0, 3.0] #access first element x[0] o/p 1.0

for n in x:

print(n)

n=n\*\*2 #modifying n does not change X list element

print x

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

# for and range

x = [1, 3, -7, 4, 9, -5, 4]

for i in range(len(x)): #len(x) is 7 range(7) =[0,1,2,3,4,5,6]

#i is indexes of your list

print i ," = ", x[i]

"""

if x[i] < 0:

print("Found a negative number at index ", i)

x.append(1,2)

"""

**Assignments to do:**

**Assignment on ----->List sorting**

1. sort the list of following names by an ascending order of number of letters in each name

unsortedList = ['Aaaa', 'bb', 'cccccccc', 'zzzzzzzzzzzz']

1. Define dictionary as

emp ={'1a':30000,'2a':40000}

1. print all employee details in sorted keys order
2. Increment the salary of every person by 5000 and then print the updated emp
3. Define num list as num = [10,45,34,88]

Print addition of all these elements