

Assignment -3

Module – 3 (Collections, functions and Modules)

1. What is List? How will you reverse a list?

-> Lists are used to store multiple items in a single variable
-> Reversing list by swapping present and last numbers at a time. Using the reversed() and reverse() built-in function

2. How will you remove last object from a list? Suppose list1 is [2, 33, 222, 14, and 25], what is list1 [-1]?

-> To remove last object in list we use pop operation.to pop or delete last object.list1[-1] in Python is the last element of the list. In this case, list1[-1] would be 25, the last element of the list.

3. Differentiate between append () and extend () methods?

-> The append() function adds the full input to the list as a single item.
-> extend() adds each item to the list independently after iterating through each one in the input.

4. Write a Python function to get the largest number, smallest num and sum of all from a list.

Code:

```
l = []
num = int(input('How many numbers: '))
for n in range(num):
    numbers = int(input('Enter number '))
    l.append(numbers)
print("Maximum element in the list is :", max(l), "\nMinimum element in the list is :", min(l))
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/
How many numbers: 5
Enter number 2
Enter number 3
Enter number 4
Enter number 5
Enter number 6
Maximum element in the list is : 6
Minimum element in the list is : 2
>>> |
```

5. How will you compare two lists?

-> `sort()` method sorts the two lists and the `==` operator compares the two lists item by item which means they have equal data items at equal positions.

6. Write a Python program to count the number of strings where the string length is 2 or more and the first and last character are same from a given list of strings.

Code:

```
def match_words(words):
    ctr = 0
    for word in words:
        if len(word) > 1 and word[0] == word[-1]:
            ctr += 1
    return ctr
print(match_words(['abc','xyz','aba','1221']))
```

Output:

```
===== RESTART: C:/User
2
>>> |
```

7. Write a Python program to remove duplicates from a list.

Code:

```
a= [10,20,30,20,10,50,60,40,80,50,40]
dup_items = set()
uniq_items = []
for x in a:
    if x not in dup_items:
        uniq_items.append(x)
        dup_items.add(x)
print(dup_items)
```

Output:

```
>>> |===== RESTART: C:/Users/user/OneDrive/
{40, 10, 80, 50, 20, 60, 30}
>>> |
```

8. Write a Python program to check a list is empty or not.

Code:

```
l = []
if not l:
    print("List is empty")
```

Output:

```
===== RESTART: C:/Users/user/
List is empty
```

9. Write a Python function that takes two lists and returns true if they have at least one common member.

Code:

```
list1=[1,2,3,4,5]
list2=[5,6,7,8,9]
result = False
for x in list1:
    for y in list2:
        if x==y:
            result = True
            print(result)
if result:
    print("List have at least one common member")
else:
    print("List do not have any commom member")
```

Output:

```
===== RESTART: C:/Users/user/OneDri
True
List have at least one common member
>>> |
```

10. Write a Python program to generate and print a list of first and last 5 elements where the values are square of numbers between 1 and 30.

Code:

```
def printValues():  
    l = list()  
    for i in range(1,21):  
        l.append(i**2)  
    print(l[:5])  
    print(l[-5:])  
printValues()
```

Output:

```
>>> | ERROR: You must have at least one commit to push.  
===== RESTART: C:/Users/user/OneDrive/  
[1, 4, 9, 16, 25]  
[256, 289, 324, 361, 400]  
>>> |
```

11. Write a Python function that takes a list and returns a new list with unique elements of the first list.

Code:

```
def unique_list(l):  
    x=[]  
    for a in l:  
        if a not in x:  
            x.append(a)  
    return x  
print(unique_list([1,2,3,3,3,4,5]))
```

Output:

```
>>> |===== RESTART: C:  
[1, 2, 3, 4, 5]  
>>> |
```

12. Write a Python program to convert a list of characters into a string.

Code:

```
def convert(s):
    new=""
    for x in s:
        new += x
    return new
s = ['u','n','n','a','t','i']
print(convert(s))
```

Output:

```
===== RE:
unnati
>>>
```

13. Write a Python program to select an item randomly from a list.

Code:

```
import random

my_list = [1,'a',32,'c',66,'d','e']
print(random.choice(my_list))
```

Output:

```
===== RESTART: C:/Users/user/Python37
e
>>>
```

14. Write a Python program to find the second smallest number in a list.

Code:

```
def find_len(list1):  
  
    length = len(list1)  
    list1.sort()  
    print("second smallest element is:",list1[1])  
list1=[12,45,2,5,22,5,6,40]  
largest = find_len(list1)
```

Output:

```
>>> |===== RESTART: C:/U  
second smallest element is: 5  
>>> |
```

15. Write a Python program to get unique values from a list

Code:

```
def unique(list1):
    unique_list=[]
    for x in list1:
        if x not in unique_list:
            unique_list.append(x)
    for x in unique_list:
        print(x)
list1 = [10,20,10,30,40,40]
print("The unique value from 1st list ")
unique(list1)
list2 = [1,2,1,1,3,4,3,3,5]
print("The unique value from 2nd list ")
unique(list2)
```

Output:

```
===== RESTART: C:/Users/user/OneDri
The unique value from 1st list
10
20
30
40
The unique value from 2nd list
1
2
3
4
5
.
```


16. Write a Python program to check whether a list contains a sub list.

Code:

```
list = [[1,5,7],[1,2,3],[6,7,8],[5,9,8]]
sub_list = [1,2,3]
if sub_list in list:
    print(" List contain sublist")
else:
    print("List not contain sublist")
```

Output:

```
>> |===== RESTART: C:/Users/user/OneDrive
    |List contain sublist
>> |
```

17. Write a Python program to split a list into different variables.

Code:

```
color = [
    ("Black", "000000", "rgb(0,0,0)"),
    ("Red", "#FF0000", "rgb(255,0,0)"),
    ("Yellow", "FFFF00", "rgb(255,255,0)")]
```

```
var1,var2,var3 = color
```

```
print(var1)
print(var2)
print(var3)
```

Output:

```
>> |===== RESTART: C:/Users/user/OneDrive/Documents
    |('Black', '000000', 'rgb(0,0,0) ')
    |('Red', '#FF0000', 'rgb(255,0,0) ')
    |('Yellow', 'FFFF00', 'rgb(255,255,0) ')
>> |
```

18. What is tuple? Difference between list and tuple.

-> Lists are mutable, which means they can be changed. The tuple is immutable, which means it cannot be changed. The set is mutable, which means that we can change it. However, no elements are duplicated

19. Write a Python program to create a tuple with different data types.

Code:

```
tuplex = ("tuple",False,3.2,1)
print(tuplex)
```

Output:

```
>>>|
===== RESTART: C:/Users/user/OneDrive/
('tuple', False, 3.2, 1)
>>>|
```

20. Write a Python program to create a tuple with numbers.

Code:

```
tuple = (1,2,3,4,5)
print(tuple)
```

```
tuple = (5,)
print(tuple)
```

Output:

```
>>>|
===== RESTART: C:/Users/user
(1, 2, 3, 4, 5)
(5,)
>>>|
```

21. Write a Python program to convert a tuple to a string.

Code:

```
t = ("u","n","n","a","t","i")
str = "".join(t)
print(str)
```

Output:

```
===== RESTART: 1
unnati
>>>
```

22. Write a Python program to check whether an element exists within a tuple.

Code:

```
tuple = ("u","n","n","a","t","i","1","2","3")

print("n" in tuple)

print("5" in tuple)
```

Output:

```
===== RESTART: C:/Users/
True
False
>> |
```

23. Write a Python program to find the length of a tuple.

Code:

```
tuple = ("u","n","n","a","t","i")
```

```
print(tuple)
```

```
print(len(tuple))
```

Output:

```
>>> |
===== RESTART: C:/Users/user
('u', 'n', 'n', 'a', 't', 'i')
6
>>> |
```

24. Write a Python program to convert a list to a tuple.

Code:

```
def convert(list):
    return tuple(list)
```

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

```
print(convert(list))
```

Output:

```
>>> |
===== RESTART: C:/Users/user
(1, 2, 3, 4, 5)
>>> |
```

25. Write a Python program to reverse a tuple.

Code:

```
def Reverse(tuples):
    new_tup = tuples[::-1]
    return new_tup

tuples = ("u","n","n","a","t","i")
print(Reverse(tuples))
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/
('i', 't', 'a', 'n', 'n', 'u')
>>>
```

26. Write a Python program to replace last value of tuples in a list.

Code:

```
l = [(10,20,30),(40,50,60),(70,80,90)]

print([t[:-1] + (100,) for t in l])
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/
[(10, 20, 100), (40, 50, 100), (70, 80, 100)]
>>>
```

27. Write a Python program to find the repeated items of a tuple.

Code:

```
tuple = 2,4,5,6,1,2,3,4,4,7
```

```
print(tuple)
```

```
count = tuple.count(4)
```

```
print(count)
```

Output:

```
===== RESTART: C:/Users/user
(2, 4, 5, 6, 1, 2, 3, 4, 4, 7)
3
>>>
```

28. Write a Python program to remove an empty tuple(s) from a list of tuples.

Code:

```
L = [(),(),(''), ('a','b'), ('a','b','c'), ('d')]
```

```
L = [t for t in L if t]
```

```
print(L)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive
[('a', 'b'), ('a', 'b', 'c'), 'd']
>>>
```

29. Write a Python program to unzip a list of tuples into individual lists.

Code:

```
l = [(1,2),(3,4),(5,6)]
```

```
result = list(zip(*l))
```

```
print(result)
```

Output:

```
===== RESTART: C:/Users/.../Python36/python.exe
>>> [(1, 3, 5), (2, 4, 6)]
```

30. Write a Python program to convert a list of tuples into a dictionary.

Code:

```
l = [("x", 1), ("x", 2), ("x", 3), ("y", 1), ("y", 2), ("z", 1)]
```

```
d = {}
```

```
for a, b in l:
```

```
    d.setdefault(a, []).append(b)
```

```
print(d)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/
{'x': [1, 2, 3], 'y': [1, 2], 'z': [1]}
.>> |
```

31. How will you create a dictionary using tuples in python?

->

- In the first step, we have created a function that takes tuple and dictionary as an input.
- After this, we have used for loop to use the setdefault() and append the subject name and the subject code.
- Now we have initialized the values of our tuple and declared the resultant dictionary as {}.
- On executing the above program, the expected output is displayed.

32. Write a Python script to sort (ascending and descending) a dictionary by value.

Code:

```
import operator

d={1: 2,3: 4,4: 3,2: 1,0: 0}
print('original dictionary : ',d)

sd=sorted(d.items(),key=operator.itemgetter(1))
print('Ascending order : ',sd)

sd=dict(sorted(d.items(),key=operator.itemgetter(1),reverse=True))
print('Descending order : ',sd)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/Python/PRAC7
original dictionary : {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
Ascending order : [(0, 0), (2, 1), (1, 2), (4, 3), (3, 4)]
Descending order : {3: 4, 4: 3, 1: 2, 2: 1, 0: 0}
>>> |
```

33. Write a Python script to concatenate following dictionaries to create a new one.

Code:

```
dic1 = {1: 10,2: 20}
dic2 = {3: 30,4: 40}
dic3 = {5: 50,6: 60}

dic4 = {}

for d in (dic1,dic2,dic3):
    dic4.update(d)

print(dic4)
```

Output:

```
>>> |===== RESTART: C:/Users/user/OneDrive/Documents/
    |{1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
```

34. Write a Python script to check if a given key already exists in a dictionary.

Code:

```
d = {1: 10,2: 20,3: 30,4: 40,5: 50,6: 60}

def is_key_present(x):
    if x in d:
        print('key is present in the dictionary')
    else:
        print('key is not present in the dictionary')

is_key_present(5)

is_key_present(9)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Do
key is present in the dictionary
key is not present in the dictionary
>>> |
```

35. How Do You Traverse Through A Dictionary Object In Python?

- > Use the `__dict__` attribute to get a dictionary containing the object's properties and values.
- > Use the `dict.items()` method to get a view of the dictionary's items.
- > Use a for loop to iterate over the object's attributes.

36. How Do You Check The Presence Of A Key In A Dictionary?

- > Using the `has_key()` method returns true if a given key is available in the dictionary, otherwise, it returns a false. With the Inbuilt method `has_key()`, use the if statement to check whether the key is present in the dictionary.

37. Write a Python script to print a dictionary where the keys are numbers between 1 and 15.

Code:

```
d = dict()

for x in range(1,16):
    d[x] = x ** 2

print(d)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/Python/PRACTICAL.PY =====
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121, 1
2: 144, 13: 169, 14: 196, 15: 225}
```

38. Write a Python program to check multiple keys exists in a dictionary

Code:

```
student = {  
    'name':'Alex',  
    'class':'V',  
    'roll_id':'2'  
}  
  
print(student.keys() >={'class','name'})  
  
print(student.keys() >={'name','alex'})  
  
print(student.keys() >={'roll_id','name'})
```

Output:

```
===== RESTART: C:/Users/user/One  
True  
False  
True  
>>> |
```

39. Write a Python script to merge two Python dictionaries

Code:

```
d1 = {'a':100,'b':200}
```

```
d2 = {'X': 300,'Y':200}
```

```
d=d1.copy()
```

```
d.update(d2)
```

```
print(d)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Docu
{'a': 100, 'b': 200, 'X': 300, 'Y': 200}
>>>
```

40. Write a Python program to map two lists into a dictionary

Code:

```
keys = ['red','green','blue']

values = ['#FF0000','#00800','#0000FF']

color_dictionary = dict(zip(keys,values))

print(color_dictionary)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/Python/PRi
{'red': '#FF0000', 'green': '#00800', 'blue': '#0000FF'}
>>> |
```

41. Write a Python program to combine two dictionary adding values for common keys.

d1 = {'a': 100, 'b': 200, 'c':300} o d2 = {'a': 300, 'b': 200,'d':400}

Sample output: Counter ({'a': 400, 'b': 400,'d': 400, 'c': 300}).

Code:

```
from collections import Counter

d1={'a':100,'b':200,'c':300}
d2={'a':300,'b':200,'c':400}

d = Counter(d1)+ Counter(d2)

print(d)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents
Counter({'c': 700, 'a': 400, 'b': 400})
>>> |
```

42. Write a Python program to print all unique values in a dictionary.

Code:

```
L = [{'V': 'S01'}, {'V': 'S02'}, {'VI': 'S01'}, {'VI': 'S05'}, {'VII': 'S05'}, {'V': 'S09'}, {'VIII': 'S07'}]
```

```
print('original List:', L)
```

```
u_value = set(val for dic in L for val in dic.values())
```

```
print('unique value:', u_value)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/Python/ass.py =====
original List: [{'V': 'S01'}, {'V': 'S02'}, {'VI': 'S01'}, {'VI': 'S05'}, {'VII': 'S05'}, {'V': 'S09'}, {'VIII': 'S07'}]
unique value: {'S05', 'S02', 'S07', 'S09', 'S01'}
> |
```

43. Why Do You Use the Zip () Method in Python?

-> Zip is an in-built function in Python used to iterate over multiple iterables.

44. Write a Python program to create and display all combinations of letters, selecting each letter from a different key in a dictionary. Sample data: {'1': ['a','b'], '2': ['c','d']}

Code:

```
import itertools
```

```
d = {'1': ['a', 'b'], '2': ['c', 'd']}
```

```
for combo in itertools.product(*[d[k] for k in sorted(d.keys())]):  
    print(''.join(combo))
```

Output:

```
===== RESTART: C:/Users/  
ac  
ad  
bc  
bd  
>>>
```

45. Write a Python program to find the highest 3 values in a dictionary

Code:

```
from heapq import nlargest
```

```
my_dict = {'a':500,'b':5874,'c':560,'d':400,'e':5874}
```

```
three_largest = nlargest(3,my_dict,key=my_dict.get)
```

```
print(three_largest)
```

Output:

```
===== RESTART: C:/Users/user/  
['b', 'e', 'c']  
>>>
```

46. Write a Python program to create a dictionary from a string.

Code:

```
from collections import defaultdict, Counter
```

```
str1 = 'unnatipargi'
```

```
my_dict = {}
```

```
for letter in str1:
```

```
    my_dict[letter] = my_dict.get(letter, 0) + 1
```

```
print(my_dict)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/Documents/Python/pra2.p
{'u': 1, 'n': 2, 'a': 2, 't': 1, 'i': 2, 'p': 1, 'r': 1, 'g': 1}
>
```

47. Write a Python function to calculate the factorial of a number (a nonnegative integer)

Code:

```
def factorial(n):  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n - 1)  
  
n = int(input("Enter number to calculate factorial: "))  
  
print(factorial(n))
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/  
Enter number to calculate factorial: 5  
120  
>
```

48. Write a Python function to check whether a number is in a given range.

Code:

```
def test_range(n):  
    if n in range(3, 9):  
        print("%s is in the range" % str(n))  
    else:  
        print("The number is outside the given range.")  
test_range(5)
```

Output:

```
===== RESTART: C:/Users/a  
5 is in the range  
>> |
```

49. Write a Python function to check whether a number is perfect or not.

Code:

```
def perfect_number(n):  
    sum=0  
    for x in range(1,n):  
        if n%x==0:  
            sum += x  
    return sum == n  
print(perfect_number(6))
```

Output:

```
===== RESTART: C  
True  
>> |
```

50. Write a Python function that checks whether a passed string is palindrome or not.

Code:

```
def isPalindrome(s):  
    return s==s[::-1]
```

```
s = 'aba'
```

```
ans = isPalindrome(s)
```

```
if ans:
```

```
    print('yes')
```

```
else:
```

```
    print('no')
```

Output:

```
| yes  
> |
```

51. How Many Basic Types Of Functions Are Available In Python?

-> There are basically 2 type of functions:

1. User define function
2. Built-in functions or library function

52. How can you pick a random item from a list or tuple?

-> Create a tuple and add some dummy data to it. Generate a random item from the tuple using `random.choice()` method by passing the input tuple as an argument to the `choice()` function.

53. How can you pick a random item from a range?

-> Random integer values can be generated with the `randint()` function it represent item from a range.

54. How can you get a random number in python?

-> `Random.uniform()` → returns a random floating number between the two specified numbers (both included). `random.randint()` → returns a random integer number selected element from a range (both included).

55. How will you set the starting value in generating random numbers?

-> Use the `seed()` method to customize the start number of the random number generator.

56. How will you randomizes the items of a list in place?

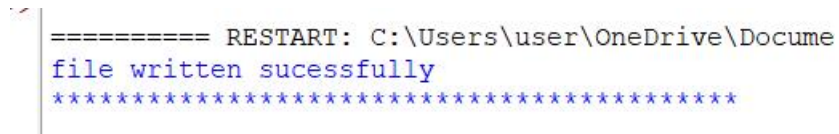
-> Python number method `shuffle()` randomizes the items of a list in place.

57. Write a Python program to read a random line from a file.

Code:

```
import random
file=open('top1','r')
print(file.read())
print('file written sucessfully')
file.close()
print('*****')
```

Output:

A screenshot of a terminal window with a dark background. The text is displayed in a light blue/cyan color. The output consists of three lines: the first line is partially cut off and shows '==== RESTART: C:\Users\user\OneDrive\Docume', the second line is 'file written sucessfully', and the third line is a line of 11 asterisks '*****'.

```
==== RESTART: C:\Users\user\OneDrive\Docume
file written sucessfully
*****
```

58. Write a Python program to convert degree to radian.

Code:

```
pi=22/7
degree = float(input("Input degrees: "))
radian = degree*(pi/180)
print(radian)
```

Output:

```
===== RESTART: C:/User
Input degrees: 120
2.0952380952380953
>> |
```

59. Write a Python program to calculate the area of a trapezoid.

Code:

```
base_1 = 5
base_2 = 6
height = float(input("Height of trapezoid: "))
base_1 = float(input('Base one value: '))
base_2 = float(input('Base two value: '))
area = ((base_1 + base_2) / 2) * height
print("Area is:", area)
```

Output:

```
===== RESTART: C:/Users/us
Height of trapezoid: 6
Base one value: 10
Base two value: 5
Area is: 45.0
>> |
```


60. Write a Python program to calculate the area of a parallelogram.

Code:

```
base = float(input('Length of base: '))
height = float(input('Measurement of height: '))
area = base * height
print("Area is: ", area)
```

Output:

```
===== RESTART: C:/Users/use
Length of base: 5
Measurement of height: 6
Area is: 30.0
>> |
```

61. Write a Python program to calculate surface volume and area of a cylinder.

Code:

```
pi=22/7
height = float(input('Height of cylinder: '))
radian = float(input('Radius of cylinder: '))
volume = pi * radian * radian * height
surface_area = ((2*pi*radian) * height) + ((pi*radian**2)*2)
print("Volume is: ", volume)
print("Surface Area is: ", sur_area)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive/D
Height of cylinder: 5
Radius of cylinder: 6
Volume is:  565.7142857142857
Surface Area is:  414.8571428571429
>> |
```

62. Write a Python program to returns sum of all divisors of a number.

Code:

```
def sum_div(number):  
    divisors = [1]  
    for i in range(2, number):  
        if (number % i)==0:  
            divisors.append(i)  
    return sum(divisors)  
print(sum_div(8))  
print(sum_div(12))
```

Output:

```
===== RESTART: C:/User  
7  
16  
>> |
```

63. Write a Python program to find the maximum and minimum numbers from the specified decimal numbers.

Code:

```
from decimal import *
data = list(map(Decimal, '2.45 2.69 2.45 3.45 2.00 0.04 7.25'.split()))
print("Maximum: ", max(data))
print("Minimum: ", min(data))
```

Output:

```
===== RESTART: C:/Users/us
Maximum:  7.25
Minimum:  0.04
>>> |
```

64. Write a Python program to combine values in python list of dictionaries. Sample data: [{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount': 300}, {'item': 'item1', 'amount': 750}]

Code:

```
from collections import Counter
```

```
item_list = [{'item': 'item1', 'amount': 400}, {'item': 'item2', 'amount': 300}, {'item': 'item1', 'amount': 750}]
```

```
result = Counter()
```

```
for d in item_list:  
    result[d['item']] += d['amount']
```

```
print(result)
```

Output:

```
===== RESTART: C:/Users/user/OneDrive  
Counter({'item1': 1150, 'item2': 300})  
>>
```