Questions on List PART II

Shubham Verma

List Theory: https://lnkd.in/ezAmJb4U

Questions Part I: https://lnkd.in/gfu3x5QH

Linkedin: https://www.linkedin.com/in/shubham-verma-3968a5119

GitHub https://lnkd.in/gky-wyFJ

Credits: W3School for questions

51. Write a python program to dispay following output.

{"title":['a', 'b', 'c'], 'city':['x', 'y', 'z'], 'likes':[1,2,3], 'views':[10,20,30]}

output : [['a', 'x', 1, 10], ['b', 'y', 2, 20], ['c', 'z', 3, 30]]

In [3]:

```
d = {"title":['a', 'b', 'c'], 'city':['x', 'y', 'z'], 'likes':[1,2,3], 'views':[10

def custom_dict_printer(in_dict):
    title, city, likes, views = list(in_dict.values())
    return [list(i) for i in zip(title, city, likes,views) ]

custom_dict_printer(d)
```

Out[3]:

```
[['a', 'x', 1, 10], ['b', 'y', 2, 20], ['c', 'z', 3, 30]]
```

52. Write a Python program to check whether the n-th element exists in a given list.

In [9]:

```
l = [1,2,3,4,5,6,7]

def element_checker_nth_place(in_l, nth_num):
    if nth_num > len(in_l):
        return "element is not present at {}th index".format(nth_num)
    else:
        return "element is present at {}th index".format(nth_num)

element_checker_nth_place(l,8)
```

Out[9]:

```
'element is not present at 8th index'
```

In [10]:

```
element_checker_nth_place(l,7)
```

Out[10]:

```
'element is present at 7th index'
```

53. Write a Python program to find a tuple, the smallest second element value from a list of tuples.

In [12]:

```
l = [(4,5,6), (1,2,3),(7,8,9), (10,0,11)]

def custom_element_extractor(in_l, position):
    return sorted(in_l, key=lambda x:x[position])[0][position]

custom_element_extractor(l,1)
```

Out[12]:

```
0
```

Que: The smallest third element value from a list of tuples.

In [13]:

```
custom_element_extractor(l,2)
```

Out[13]:

```
3
```

54. Write a Python program to create a list of empty dictionaries.

In [19]:

```
def empty_dict_creator(no_of_dict):
    return [{} for x in range(no_of_dict)]

empty_dict_creator(5)
```

Out[19]:

```
[{}, {}, {}, {}, {}]
```

55. Write a Python program to print space-separated elements from elements of list.

In [25]:

```
l = [1,2,3,4,5,6,7,8,9,10]

def space_sep_list_element(in_l):
    print(*in_l)

space_sep_list_element(l)
```

```
1 2 3 4 5 6 7 8 9 10
```

56. Write a Python program to insert a given string at the beginning of all items in a list.

In [29]:

```
l = [1,2,3,4,5]
l2 = ["Ramesh","Suresh", "Mukesh"]
string_insert = "Roll No."
string_insert2 = "First_name: "

def custom_converter_list_element(in_l, in_str):
    l = []
    for i in in_l:
        if type(i) == int:
            l.append(in_str+str(i))
        elif type(i) == str:
            l.append(in_str+i)
        else:
            pass
    return l

custom_converter_list_element(l, string_insert)
```

Out[29]:

```
['Roll No.1', 'Roll No.2', 'Roll No.3', 'Roll No.4', 'Roll No.5']
```

In [30]:

```
custom_converter_list_element(l2, string_insert2)
```

Out[30]:

```
['First_name: Ramesh', 'First_name: Suresh', 'First_name: Mukesh']
```

57. Write a Python program to iterate over two lists simultaneously.

In [31]:

```
l_name= ["Ramesh","Suresh", "Mukesh"]
l_surname = ['Daniels','Hayden', 'Stanford']

def two_list_iterator(in_l1, in_l2):
    for first,second in zip(in_l1,in_l2):
        print(first, second)

two_list_iterator(l_name, l_surname)
```

```
Ramesh Daniels
Suresh Hayden
Mukesh Stanford
```

58. Write a Python program to move all zero digits to end of a given list of numbers.

In [34]:

```
l = [1,3,0,9,0,4,7,0,6,0,5,8,0,0,2]

def zero_arranger_list(in_l):
    l = []
    l_zero = []
    for i in in_l:
        if i == 0:
            l_zero.append(i)
        else:
            l.append(i)
    l.extend(l_zero)
    return l

zero_arranger_list(l)
```

Out[34]:

```
[1, 3, 9, 4, 7, 6, 5, 8, 2, 0, 0, 0, 0, 0, 0]
```

59. Write a Python program to find the list in a list of lists whose sum of elements is the highest.

In [14]:

```
l = [[1,2,3], [4,5,6], [10,11,12], [7,8,9]]

def custom_sum_sublist_highest(in_l):
    return max(in_l, key=sum)

custom_sum_sublist_highest(l)
```

Out[14]:

```
[10, 11, 12]
```

60. Write a Python program to find all the values in a list are greater than a specified number.

In [16]:

```
list1 = [1,2,3,4,5,6,7,8,9,10]

def custom_list_greater_element(in_l, number):
    l = []
    for i in in_l:
        if i > number:
            l.append(i)
    return l

custom_list_greater_element(list1,5)
```

Out[16]:

```
[6, 7, 8, 9, 10]
```

61. Write a Python program to extend a list without append.

In [17]:

```
l1 = [1,2,3]
l2 = ["a","b", "c"]

def custom_extend(in_l1, in_l2):
    return in_l1+in_l2

custom_extend(l1,l2)
```

Out[17]:

```
[1, 2, 3, 'a', 'b', 'c']
```

62. Write a Python program to remove duplicates from a list of lists.

In [22]:

```
import itertools
l = [[10, 20], [40], [30, 56, 25], [10, 20], [33], [40]]

def dup_remover_list(in_l):
    in_l.sort()
    return list(x for x,_ in itertools.groupby(in_l))

dup_remover_list(l)
```

Out[22]:

```
[[10, 20], [30, 56, 25], [33], [40]]
```

63. Write a Python program to find the items starts with specific character from a given list.

In [23]:

```
l = ["abc", 'apq', 'bpq','pqr', 'abz']

def custom_startswith_list(in_l, char):
    l= []
    for i in in_l:
        if i.startswith(char):
            l.append(i)
    return l

custom_startswith_list(l,'a')
```

Out[23]:

```
['abc', 'apq', 'abz']
```

64. Write a Python program to check whether all dictionaries in a list are empty or not.

In [25]:

```
d1 = [{},{},{}]
d2 = [{},{'k':'v'},{}]

def empty_dict_list_checker(in_l):
    return all(not x for x in in_l)

empty_dict_list_checker(d1)
```

Out[25]:

```
True
```

In [26]:

```
empty_dict_list_checker(d2)
```

Out[26]:

```
False
```

65. Write a Python program to flatten a given nested list structure.

In [30]:

```
l_nested = [0, 10, [20, 30], 40, 50, (60, 70, 80), {90, 100, 110, 120},{'k':'v'}]

def nested_to_flat_list(in_list):
    l=[]
    for i in in_list:
        if type(i) in [list,set,tuple]:
            for j in i:
                l.append(j)
        elif type(i) == dict:
            for j,k in i.items():
                l.append(j)
                l.append(k)
        else:
            l.append(i)
    return l

nested_to_flat_list(l_nested)
```

Out[30]:

```
[0, 10, 20, 30, 40, 50, 60, 70, 80, 120, 90, 100, 110, 'k', 'v']
```

66. Write a Python program to remove consecutive duplicates of a given list.

In [31]:

```
from itertools import groupby

l = [1,1,2,1,2,2,3,3,4,4,4,4,5,4,6,6,7,7,7,1]

def compress_list(in_l):
    return [x for x, group in groupby(in_l)]

compress_list(l)
```

Out[31]:

```
[1, 2, 1, 2, 3, 4, 5, 4, 6, 7, 1]
```

67. Write a Python program to pack consecutive duplicates of a given list elements into sublists.

In [33]:

```
from itertools import groupby
```

```
l = [1,1,2,1,2,2,3,3,4,4,4,4]

def pack_duplicate_list(in_l):
    return [list(group) for x, group in groupby(in_l)]

pack_duplicate_list(l)
```

Out[33]:

```
[[1, 1], [2], [1], [2, 2], [3, 3], [4, 4, 4, 4]]
```

68. Write a Python program to create a list reflecting the modified run-length encoding from a given list of integers or a given list of characters.

In [35]:

```
from itertools import groupby

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

def encoding_list(in_l):
    return [[len(list(group)),x] for x, group in groupby(in_l)]

encoding_list(l)
```

Out[35]:

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

69. Write a Python program to split a given list into two parts where the length of the first part of the list is given.

In [39]:

```
l = [1,2,3,4,5,6,7,8,9,10]

def list_splitter(in_l, length_at_split):
       return [in_l[:length_at_split], in_l[length_at_split:]]

list_splitter(l,5)
```

Out[39]:

```
[[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]]
```

70. Write a Python program to remove the K'th element from a given list

In [5]:

```
l = [1,2,3,4,5,6,7,8,9,10]

def kth_element_remove_list(in_l, kth_element):
    in_l.pop(kth_element-1)
    return in_l

kth_element_remove_list(l, 10)
```

Out[5]:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

71. Write a Python program to insert an element at a specified position into a given list.

In [6]:

```
l = [1,2,3,4,5,6,7,8,9,10]

def list_insert(in_l, element, position):
    in_l.insert(position-1, element)
    return in_l

list_insert(l,"Shubham", 1)
```

Out[6]:

```
['Shubham', 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

72. Write a Python program to extract a given number of randomly selected elements from a given list.

In [8]:

```
import random
l = [1,2,3,4,5,6,7,8,9,10]

def random_selection_list(in_l, no_of_element):
    return random.sample(in_l, no_of_element)

random_selection_list(l,5)
```

Out[8]:

```
[2, 4, 5, 9, 3]
```

73. Write a Python program to round every number of a given list of numbers and print the total sum multiplied by the length of the list.

In [9]:

```
l = [22.4, 4.0, -16.22, -9.1, 11.0, -12.22, 14.2, -5.2, 17.5]

def custom_list_sum_round(in_l):
    sum = 0
    for i in in_l:
        sum += round(i)
    return sum*len(in_l)

custom_list_sum_round(l)
```

Out[9]:

```
243
```

74. Write a Python program to round the numbers of a given list, print the minimum and maximum numbers.

In [13]:

```
l = [22.4, 4.0, -16.22, -9.1, 11.0, -12.22, 14.2, -5.2, 17.5]

def min_max_round_list(in_l):
    l = sorted([round(i) for i in in_l])
    return "Minimun: {} and Maximum: {}".format(l[0],l[-1])

min_max_round_list(l)
```

Out[13]:

```
'Minimun: -16 and Maximum: 22'
```

75. Write a Python program to create a multidimensional list (lists of lists) with zeros. for example: Multidimensional list: [[0, 0], [0, 0], [0, 0]]

In [1]:

```
def return_multi_dim_list_zeros(n_row, n_col):
    l = []
    for i in range(n_row):
        l.append([])
        for j in range(n_col):
            l[i].append(0)
    return l

return_multi_dim_list_zeros(3,2)
```

Out[1]:

```
[[0, 0], [0, 0], [0, 0]]
```

76. Write a Python program to create a 3X3 grid with numbers. For example 3X3 grid with numbers: [[1, 2, 3], [1, 2, 3], [1, 2, 3]]

In [3]:

```
def custom_grid_numbers(n_row, n_col):
    l = []
    for i in range(n_row):
        l.append([])
        for j in range(1, n_col+1):
            l[i].append(j)
    return l

custom_grid_numbers(3,3)
```

Out[3]:

```
[[1, 2, 3], [1, 2, 3], [1, 2, 3]]
```

77. Write a Python program to read a matrix from console and print the sum for each column. Accept matrix rows, columns and elements for each column separated with a space(for every row) as input from the user.

In [24]:

```
def matrix_console_value(n_row, n_col):
    matrix = [[0]*n_col for row in range(n_row)]
    print('Input number of elements in a row seprated by space')
    for row in range(n_row):
        lines = list(map(int, input().split()))
        for column in range(n_col):
            matrix[row][column] = lines[column]

    sum = [0]*n_col
    print("sum for each column:")
    for column in range(n_col):
        for row in range(n_row):
            sum[column] += matrix[row][column]
        print((sum[column]), ' ', end = '')

matrix_console_value(2,2)
```

```
Input number of elements in a row seprated by space
1 2
3 4
sum for each column:
4  6
```

78. Write a Python program to Zip two given lists of lists.

Original lists:

[[1, 3], [5, 7], [9, 11]]

[[2, 4], [6, 8], [10, 12, 14]]

Zipped list:

[[1, 3, 2, 4], [5, 7, 6, 8], [9, 11, 10, 12, 14]]

In [25]:
```python
l1 = [[1, 3], [5, 7], [9, 11]]
l2 = [[2, 4], [6, 8], [10, 12, 14]]

def custom_zip(in_l1, in_l2):
    return list(map(list.__add__, in_l1, in_l2))

custom_zip(l1,l2)
```

Out[25]:
```
[[1, 3, 2, 4], [5, 7, 6, 8], [9, 11, 10, 12, 14]]
```

79. Write a Python program to count number of lists in a given list of lists.

In [26]:
```python
l = [[1, 3], [5, 7], [9, 11], [13, 15, 17]]

def list_counter(in_l):
    count = 0
    for i in in_l:
        if type(i) == list:
            count+=1
    return count

list_counter(l)
```

Out[26]:
```
4
```

80. Write a Python program to find the list with maximum and minimum length.

In [28]:
```python
l = [[0], [1, 3], [5, 7], [9, 11], [13, 15, 17]]

def list_max_min_length(in_l):
    max_length = (max(len(x) for x in in_l), max(in_l, key = len))
    min_length = (min(len(x) for x in in_l), min(in_l, key = len))
    return "Max length and list {} and Min length and list {}". format(max_length,

list_max_min_length(l)
```

Out[28]:
```
'Max length and list (3, [13, 15, 17]) and Min length and list (1, [0])'
```

81. Write a Python program to count the number of sublists contain a particular element.

In [3]:
```python
l = [['A', 'B'], ['A', 'C'], ['A', 'D', 'E'], ['B', 'C', 'D']]
l1 =[[1, 3], [5, 1], [1, 11], [1, 15, 7]]

def count_element_in_sublist(in_l, element):
    count=0
    for i in in_l:
        if element in i:
            count+=1
    return count

count_element_in_sublist(l,'A')
```

Out[3]:

```
3
```

In [4]:

```
count_element_in_sublist(l1,1)
```

Out[4]:

```
4
```

82. Write a Python program to sort each sublist in a given list of lists.

In [14]:

```
l = [[2], [0], [1, 3], [0, 7], [9, 11], [13, 15, 17]]

def custom_sort_sublist(in_l):
    return sorted(in_l, key=lambda x: x[0])

custom_sort_sublist(l)
```

Out[14]:

```
[[0], [0, 7], [1, 3], [2], [9, 11], [13, 15, 17]]
```

83. Write a Python program to sort a given list of lists by length and value.

In [16]:

```
l = [[2], [0], [1, 3], [0, 7], [9, 11], [13, 15, 17]]

def custom_sort_sublist(in_l):
    in_l.sort()
    in_l.sort(key=len)
    return in_l

custom_sort_sublist(l)
```

Out[16]:

```
[[0], [2], [0, 7], [1, 3], [9, 11], [13, 15, 17]]
```

84. Write a Python program to remove sublists from a given list of lists, which contains an element outside a given range

In [18]:

```
l = [[2], [0], [1, 2, 3], [0, 1, 2, 3, 6, 7], [9, 11], [13, 14, 15, 17]]

def custom_sublist_remover(in_l, a,b):
    l = [i for i in in_l if min(i)>=a and max(i)<=b]
    return l

custom_sublist_remover(l,13,17)
```

Out[18]:

```
[[13, 14, 15, 17]]
```

85. Write a Python program to scramble the letters of string in a given list.

In [33]:

```
#pip install python_string_utils
```

In [35]:

```
import string_utils
l = ['Python', 'list', 'exercises', 'practice', 'solution']
```

```python
def string_list_scramble(in_l):
    l=[]
    for i in in_l:
        l.append(string_utils.shuffle(i))
    return l

string_list_scramble(l)
```

Out[35]:

```
['htoPny', 'list', 'eseesxcir', 'rtiacpce', 'ooulnsit']
```

86. Write a Python program to find the maximum and minimum values in a given heterogeneous list.

In [38]:

```python
l =['Python', 3, 2, 4, 5, 'version']

def max_min_list_hetrogeneous(in_l):
    l = []
    for i in in_l:
        if type(i) == int:
            l.append(i)
    return "maximum value: {} and minimum value: {}".format(max(l), min(l))

max_min_list_hetrogeneous(l)
```

Out[38]:

```
'maximum value: 5 and minimum value: 2'
```

87. Write a Python program to extract common index elements from three given list.

In [42]:

```python
l1 = [0,1,2,3,4,5,6]
l2 = [0,2,3,4,4,6,6]
l3 = [0,1,2,3,4,5,6]

def common_index_element(in_l1,in_l2, in_l3):
    l=[]
    for i,j,k in zip(in_l1,in_l2, in_l3):
        if i==j==k:
            l.append(i)
    return l

common_index_element(l1,l2,l3)
```

Out[42]:

```
[0, 4, 6]
```

88. Write a Python program to extract specified size of strings from a give list of string values

In [44]:

```python
l = ['my', "name","is","samrinjeet","and","i","live", "in","chandigarh"]

def string_from_list_len(in_l, length):
    l=[]
    for i in in_l:
        if len(i) == length:
            l.append(i)
    return l

string_from_list_len(l,2)
```

Out[44]:

```
['my', 'is', 'in']
```

89. Write a Python program to extract specified number of elements from a given list, which follows each other continuously.

In [47]:

```python
from itertools import groupby
l = [1,22,22,22,22,3,3,4,4,4,4]

def custom_len_number(in_l, no_of_times_repeting):
    return [i for i, j in groupby(in_l) if len(list(j)) == no_of_times_repeting]

custom_len_number(l,4)
```

Out[47]:

```
[22, 4]
```

90. Write a Python program to compute average of two given lists.

In [54]:

```python
l1= [i for i in range(10)]
l2= [i for i in range(11,20)]

def average_two_list(in_l1, in_l2):
    return round((sum(in_l1)+sum(in_l2))/(len(in_l1)+len(in_l2)),3)

average_two_list(l1,l2)
```

Out[54]:

```
9.474
```

91. Write a Python program to count integer in a given mixed list.

In [55]:

```python
l = [1, 'abcd', 3, 1.2, 4, 'xyz', 5, 'pqr', 7, -5, -12.22]

def int_counter_mixed_list(in_l):
    count=0
    for i in in_l:
        if type(i) == int:
            count+=1
    return count

int_counter_mixed_list(l)
```

Out[55]:

```
6
```

92. Write a Python program to remove a specified column from a given nested list.

In [56]:

```python
l = [[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]]

def custom_column_remover(in_l, col_number):
    for i in in_l:
        del i[col_number-1]
    return in_l

custom_column_remover(l,1)
```

Out[56]:
```
[[2, 3, 4], [6, 7, 8], [10, 11, 12], [14, 15, 16]]
```

In [58]:
```
l = [[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]]
custom_column_remover(l,2)
```

Out[58]:
```
[[1, 3, 4], [5, 7, 8], [9, 11, 12], [13, 15, 16]]
```

93. Write a Python program to extract a specified column from a given nested list.

In [62]:
```
l = [[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]]

def custom_column_extractor(in_l, column_number):
    return [i.pop(column_number-1) for i in in_l]

custom_column_extractor(l,2)
```

Out[62]:
```
[2, 6, 10, 14]
```

94. Write a Python program to rotate a given list by specified number of items to the right.

In [65]:
```
l = [1,2,3,4,5,6,7,8,9,10]

def list_right_rotate(in_list, position):
    return in_list[-(position-1):]+in_list[:-position]

list_right_rotate(l,4)
```

Out[65]:
```
[8, 9, 10, 1, 2, 3, 4, 5, 6]
```

95. Write a Python program to rotate a given list by specified number of items to the left direction.

In [67]:
```
l = [1,2,3,4,5,6,7,8,9,10]

def list_left_rotate(in_list, position):
    return in_list[(position-1):]+in_list[:position]

list_left_rotate(l,4)
```

Out[67]:
```
[4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4]
```

96. Write a Python program to find the item with maximum occurrences in a given list.

In [68]:
```
l = [1,1,1,1,1,1,1,2,2,2,3,3,3,4,4,4,5,5,5]

def max_occur_element_list(in_l):
    count_ele=0
    for i in in_l:
        occurance_ele=in_l.count(i)
        if occurance_ele>count_ele:
            count_ele=occurance_ele
            result = i
    return result
```

```
max_occur_element_list(l)
```

Out[68]:

```
1
```

97. Write a Python program to check whether a specified list is sorted or not.

In [72]:

```
l = [1,2,3,4,5,6,7,8,9,10]
l1= [1,3,2,4,5,6,7,8,9,10]
l2 = [10,9,8,7,6,5,4,3,2,1]

def sorted_list_checker(in_l):
    if in_l == sorted(in_l) or in_l == sorted(in_l, reverse=True):
        return True
    else:
        return False

sorted_list_checker(l)
```

Out[72]:

```
True
```

In [71]:

```
sorted_list_checker(l1)
```

Out[71]:

```
False
```

In [73]:

```
sorted_list_checker(l2)
```

Out[73]:

```
True
```

98. Write a Python program to extract the nth element from a given list of tuples.

In [74]:

```
l = [('Greyson Fulton', 98, 99), ('Brady Kent', 97, 96), ('Wyatt Knott', 91, 94),

def nth_element_list_tuple_extractor(in_l, nth_element):
    l = []
    for i in in_l:
        l.append(i[nth_element])
    return l

nth_element_list_tuple_extractor(l,0)
```

Out[74]:

```
['Greyson Fulton', 'Brady Kent', 'Wyatt Knott', 'Beau Turnbull']
```

99. Write a Python program to check if the elements of a given list are unique or not.

In [75]:

```
l = [1,1,2,2,3,3,4,4]
l1 = [1,2,3,4,5,6,7,8]

def unique_all_element_list_checker(in_l):
    if len(in_l) == len(set(in_l)):
        return True
    else:
        return False
```

```
unique_all_element_list_checker(l)
```

Out[75]:

```
False
```

In [76]:

```
unique_all_element_list_checker(l1)
```

Out[76]:

```
True
```

100. Write a Python program to sort a list of lists or tuples by a given index of the inner list or tuple.

In [77]:

```
l = [('Greyson Fulton', 98, 99), ('Brady Kent', 97, 96), ('Wyatt Knott', 91, 94), (

def custom_sort_list_by_index(in_l, index_inner_list):
    return sorted(in_l, key=lambda x:x[index_inner_list])

custom_sort_list_by_index(l,0)
```

Out[77]:

```
[('Beau Turnbull', 94, 98),
 ('Brady Kent', 97, 96),
 ('Greyson Fulton', 98, 99),
 ('Wyatt Knott', 91, 94)]
```

In [78]:

```
custom_sort_list_by_index(l,2)
```

Out[78]:

```
[('Wyatt Knott', 91, 94),
 ('Brady Kent', 97, 96),
 ('Beau Turnbull', 94, 98),
 ('Greyson Fulton', 98, 99)]
```

In [79]:

```
custom_sort_list_by_index(l,1)
```

Out[79]:

```
[('Wyatt Knott', 91, 94),
 ('Beau Turnbull', 94, 98),
 ('Brady Kent', 97, 96),
 ('Greyson Fulton', 98, 99)]
```