Data Analysis & Visualization Practical Assignment 1

August 5, 2022

Name: Amartya Sinha Roll No: AC-1207

[ ]:
```
# 1.Given below is a dictionary having two keys 'Boys' and 'Girls' and having
# two lists of heights of five Boys and Five Girls respectively as values
# associated with these keys.

#   Original dictionary of lists:
#   {'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}

#   From the given dictionary of lists create the following list of
#   dictionaries:
#   [{'Boys': 72, 'Girls': 63},
#    {'Boys': 68, 'Girls': 65},
#    {'Boys': 70, 'Girls': 69},
#    {'Boys': 69, 'Girls': 62},
#    {'Boys':74,  'Girls' :61}]
```

[ ]:
```
dic1 = {'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}
dic1
```

[ ]:
```
{'Boys': [72, 68, 70, 69, 74], 'Girls': [63, 65, 69, 62, 61]}
```

[ ]:
```
lst = []

for height_ind in range(5):  # using 5 as there are five boys and five girls
    d = {}
    for key in dic1:
        d[key] = dic1[key][height_ind]
    lst.append(d)

lst
```

[ ]:
```
[{'Boys': 72, 'Girls': 63},
 {'Boys': 68, 'Girls': 65},
 {'Boys': 70, 'Girls': 69},
 {'Boys': 69, 'Girls': 62},
 {'Boys': 74, 'Girls': 61}]
```

[ ]:
```
# 2.Given are three lists where L1 has names of n students, L2 has marks and
# L3 has hobbies of n students.Using three lists, create the following
# dictionary with hobbies as keys and (names,marks) as values. In case more
# than one student has same hobby then values must be appended for the same
# key instead of overwriting.

#   E.g.
#   L1=['A','B','C']L2=[100,40,50] L3=['painting','music','painting']

#   then output should be:
#   dict1={painting:(('A',100),('C',50)), 'music':('B',40)}
```

[ ]:
```
L1 = ['Abhishek', 'Aditi', 'Aditya', 'Aman', 'Amartya', 'Amit']
L2 = [75, 80, 84, 75, 82, 79]
L3 = ['cricket', 'painting', 'coding', 'cricket', 'coding', 'cricket']
```

[ ]:
```
dic2 = {}
myzip = zip(L3, L2, L1)

for hb, mrk, nm in myzip:
    # when hobbey key does not exist
    if not hb in dic2.keys():
        dic2[hb] = (nm, mrk)
    # when only one value is for the hobby
    elif type(dic2[hb]) == tuple:
        temp = []
        temp.append(dic2[hb])
        temp.append((nm, mrk))
        dic2[hb] = temp
    # when more than one value exist for the hobby
    else:
        dic2[hb].append((nm, mrk))


for i in dic2.keys():
    dic2[i] = tuple(dic2[i])
dic2
```

[ ]:
```
{'cricket': (('Abhishek', 75), ('Aman', 75), ('Amit', 79)),
 'painting': ('Aditi', 80),
 'coding': (('Aditya', 84), ('Amartya', 82))}
```

[ ]:
```
# 3. Write  two  lambda functions

#  a. One to  arrange a list of names on the last letter of the name i.e.
#     names=['axc','bxbb','xxb','zzxy','zzc']
#     then new sorted list on last letter  is
```

```
#     ['bxbb', 'xxb', 'axc', 'zzc', 'zzxy']

#  b. Second lambda function arranges list on the length of names and store
#     results in dictionary with length as key and value as names
```

[ ]:
```
names = ['Alan', 'Bradman', 'Adan', 'Albert', 'West', 'Will', 'Steven']
```

[ ]:
```
l_letter_sorted = sorted(names, key = lambda x: x[-1])
l_letter_sorted
```

[ ]:
```
['Will', 'Alan', 'Bradman', 'Adan', 'Steven', 'Albert', 'West']
```

[ ]:
```
len_sorted = sorted(names, key=lambda x: len(x))

mydic = {}
for val in len_sorted:
    if len(val) not in mydic.keys():
        mydic[len(val)] = [val]
    else:
        mydic[len(val)].append(val)

mydic
```

[ ]:
```
{4: ['Alan', 'Adan', 'West', 'Will'], 6: ['Albert', 'Steven'], 7: ['Bradman']}
```

[ ]:
```
# alternate method

len_sorted = sorted(names, key=lambda x: len(x))

mydico = {}
result = lambda x: mydico.update({len(x) : [x]} if len(x) not in mydico.keys()␣
 ↪else {len(x) : mydico[len(x)] + [x]})
[result(val) for val in len_sorted]

mydico
```

[ ]:
```
{4: ['Alan', 'Adan', 'West', 'Will'], 6: ['Albert', 'Steven'], 7: ['Bradman']}
```

[ ]:
```
# 4. Write programs in Python using NumPy library to do the following:

#  a. Compute the mean, standard deviation, and variance of a two dimensional
#     random integer arrayalong the second axis.

#  b. Get the indices of the sorted elements of a given array.
#     B= [56, 48, 22, 41, 78, 91, 24, 46, 8, 33]

#  c. Create a 2-dimensional array of size m x n integer elements, also print
```

```
#     the shape, type and datatype of the array and then reshape it into nx m
#     array, n and m are user inputs given at the run time.

#  d. Test whether the elements of a given array are zero, non-zero and NaN.
#     Record the indices of these elements in three separate arrays.
```

[ ]:
```
import numpy as np

myarr = np.random.randint(5, 10, (1, 4))

print("Two Dimensional Random integer Array is", myarr)

print("Its shape is", myarr.shape)

print("\nMean is", np.mean(myarr))
print("Standard Deviation is", np.std(myarr))
print("Variance is", np.var(myarr))
```

```
Two Dimensional Random integer Array is [[7 8 9 6]]
Its shape is (1, 4)

Mean is 7.5
Standard Deviation is 1.118033988749895
Variance is 1.25
```

[ ]:
```
B = np.array([56, 48, 22, 41, 78, 91, 24, 46, 8, 33])

print("Array is", B)
print("Indices of the sorted elements of array B are", np.argsort(B))
```

```
Array is [56 48 22 41 78 91 24 46  8 33]
Indices of the sorted elements of array B are [8 2 6 9 3 7 1 0 4 5]
```

[ ]:
```
m = int(input("Enter m:"))
n = int(input("Enter n:"))

myarray = np.random.randint(5,10, (m,n))

print("Array of size ({0}, {1}):\n {2}".format(m, n, myarray))
print("Its shape is", myarray.shape)

reshaped_arr = np.reshape(myarray, (n, m))

print("\nReshaped Array of size ({0}, {1}):\n {2}".format(n, m, myarray))
print("Its shape is", reshaped_arr.shape)
```

```
Array of size (3, 4):
 [[7 9 9 8]
```

```
 [7 6 6 5]
 [9 5 8 9]]
Its shape is (3, 4)

Reshaped Array of size (4, 3):
 [[7 9 9 8]
 [7 6 6 5]
 [9 5 8 9]]
Its shape is (4, 3)
```

[ ]:
```
arr_is_back = np.array([1, np.nan, 0, 5, 8, 34, 0, 3, np.nan])

print("Indices of Non Zero Elements:\n", np.argwhere(arr_is_back != 0))
print("\nIndices of Zero Elements:\n", np.argwhere(arr_is_back == 0))
print("\nIndices of NaN Elements:\n", np.argwhere(np.isnan(arr_is_back)))
```

```
Indices of Non Zero Elements:
 [[0]
 [1]
 [3]
 [4]
 [5]
 [7]
 [8]]

Indices of Zero Elements:
 [[2]
 [6]]

Indices of NaN Elements:
 [[1]
 [8]]
```

[ ]:
```
# 5. Generate a 2D array having values from 1 to 100  of shape (4,5).

#   I. Multiply all elements that are greater than 50 by 2
#  II. Extract 2x2 lower-right matrix from the given array and find its
#      largest element
```

[ ]:
```
another_array = np.random.randint(1,100, (4,5))
print("Array of shape (4,5) with values from 1 to 100:\n",another_array)
```

```
Array of shape (4,5) with values from 1 to 100:
 [[88  6 66  2 90]
 [77 34  4  3 98]
 [44 30 15 61 60]
 [43 27 66 88 68]]
```

[ ]:
```
another_array[another_array > 50] = 2*another_array[another_array > 50]

print("Elements Greater than 50 after multiplied by 2:\n",another_array)
```

```
Elements Greater than 50 after multiplied by 2:
 [[176   6 132   2 180]
 [154  34   4   3 196]
 [ 44  30  15 122 120]
 [ 43  27 132 176 136]]
```

[ ]:
```
small_matrix = another_array[-2:, -2:]

print("2X2 lower-right matrix from the original array:\n", small_matrix)
print("\nIts maximum element is", np.amax(small_matrix))
```

```
2X2 lower-right matrix from the original array:
 [[122 120]
 [176 136]]

Its maximum element is 176
```