Quiz 2 - Brandan Owens and Loan Pham

Q.1 Create the following lists

In [1]:

```
import math
import numpy as np
import random
import pandas as pd
from scipy import stats
```

In [2]:

```
# (a)  set the random seed to be 50

np.random.seed(50)
np.random.randn(50)
```

Out[2]:

```
array([-1.56035211, -0.0309776 , -0.62092842, -1.46458049,  1.41194612,
       -0.47673214, -0.78046921,  1.07026774, -1.2822926 , -1.3274789 ,
        0.12633764,  0.86219372,  0.69673696, -0.33456518, -0.99752606,
        1.59890829,  3.31407535,  0.98777046,  0.12386626,  0.74278539,
       -0.39395585,  0.14811582, -0.41223445, -0.16071506,  0.13953147,
        0.28546937, -0.28126199,  1.71090732, -0.14976664,  0.69030672,
        1.09520951,  1.3384087 , -1.36898167,  0.48642763,  0.75352168,
        0.36346459, -0.31471048,  1.37328117, -0.62441716,  0.375754  ,
       -0.20041632,  0.74303806,  0.85736196, -1.50618929, -1.66635217,
       -0.2189948 , -0.35885843,  0.37852769,  0.68421537, -1.16785607])
```

In [3]:

```
# (b) create an 8x6 array of random integers. The integers should range from 5 to 56.
num = np.random.randint(5,56, size=(8,6))
num
```

Out[3]:

```
array([[48, 40, 43, 32, 55, 33],
       [36, 45, 20, 10, 52,  9],
       [ 9, 23, 52, 47, 22, 31],
       [13, 51, 16, 32, 32, 12],
       [46,  6, 13, 40,  8, 26],
       [47,  8, 11, 20, 36, 27],
       [ 5, 55, 16, 35, 38, 54],
       [19, 31, 27, 51, 42, 11]])
```

In [4]:

```
# (c)  access the element in the second row and the third column.
num [1,2]
```

Out[4]:

```
20
```

In [5]:

```
# (d) create an array by accessing the final column. Reverse the array.
x = num[:,-1]
print(x)
np.flip(x)
```

```
[33  9 31 12 26 27 54 11]
```

Out[5]:

```
array([11, 54, 27, 26, 12, 31,  9, 33])
```

In [6]:

```
# (e)  create an array by finding the max of each column.
num_max = np.max(num, axis = 1)
num_max
```

Out[6]:
```
array([55, 52, 52, 51, 46, 47, 55, 51])
```

In [7]:
```
# (f)  reshape the array into 1x48 array.
num_reshaped = num.reshape(1,48)
num_reshaped
```

Out[7]:
```
array([[48, 40, 43, 32, 55, 33, 36, 45, 20, 10, 52,  9,  9, 23, 52, 47,
        22, 31, 13, 51, 16, 32, 32, 12, 46,  6, 13, 40,  8, 26, 47,  8,
        11, 20, 36, 27,  5, 55, 16, 35, 38, 54, 19, 31, 27, 51, 42, 11]])
```

Q.2

In [8]:
```
# (a) import the file “mpg.csv”

data = pd.read_csv("..\dataFiles\mpg.csv")
data
```

Out[8]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
0	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
3	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
4	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact
...	...	...	...	...	...	...	...	...	...	...	...
229	volkswagen	passat	2.0	2008	4	auto(s6)	f	19	28	p	midsize
230	volkswagen	passat	2.0	2008	4	manual(m6)	f	21	29	p	midsize
231	volkswagen	passat	2.8	1999	6	auto(l5)	f	16	26	p	midsize
232	volkswagen	passat	2.8	1999	6	manual(m5)	f	18	26	p	midsize
233	volkswagen	passat	3.6	2008	6	auto(s6)	f	17	26	p	midsize

234 rows × 11 columns

In [9]:
```
# (b) extract the column [“hwy”] and covert it into a numpy array.

hwy = np.array(data["hwy"])
hwy
```

Out[9]:
```
array([29, 29, 31, 30, 26, 26, 27, 26, 25, 28, 27, 25, 25, 25, 25, 24, 25,
       23, 20, 15, 20, 17, 17, 26, 23, 26, 25, 24, 19, 14, 15, 17, 27, 30,
       26, 29, 26, 24, 24, 22, 22, 24, 24, 17, 22, 21, 23, 23, 19, 18, 17,
       17, 19, 19, 12, 17, 15, 17, 17, 12, 17, 16, 18, 15, 16, 12, 17, 17,
       16, 12, 15, 16, 17, 15, 17, 17, 18, 17, 19, 17, 19, 19, 17, 17, 17,
       16, 16, 17, 15, 17, 26, 25, 26, 24, 21, 22, 23, 22, 20, 33, 32, 32,
       29, 32, 34, 36, 36, 29, 26, 27, 30, 31, 26, 26, 28, 26, 29, 28, 27,
       24, 24, 24, 22, 19, 20, 17, 12, 19, 18, 14, 15, 18, 18, 15, 17, 16,
       18, 17, 19, 19, 17, 29, 27, 31, 32, 27, 26, 26, 25, 25, 17, 17, 20,
       18, 26, 26, 27, 28, 25, 25, 24, 27, 25, 26, 23, 26, 26, 26, 26, 25,
       27, 25, 27, 20, 20, 19, 17, 20, 17, 29, 27, 31, 31, 26, 26, 28, 27,
       29, 31, 31, 26, 26, 27, 30, 33, 35, 37, 35, 15, 18, 20, 20, 22, 17,
       19, 18, 20, 29, 26, 29, 29, 24, 44, 29, 26, 29, 29, 29, 29, 23, 24,
       44, 41, 29, 26, 28, 29, 29, 29, 28, 29, 26, 26, 26], dtype=int64)
```

In [10]:

```
# (c) find the sample size of “hwy”

def sampleSize (population):
    error = 0.05
    confidence = 0.95
    sigma = 1/2
    alpha = 1 - (confidence)
    zdict = {
        .90: 1.645,
        .91: 1.695,
        .99: 2.576,
        .97: 2.17,
        .94: 1.881,
        .93: 1.812,
        .95: 1.96,
        .98: 2.326,
        .96: 2.054,
        .92: 1.751
    }
    if confidence in zdict:
        z = zdict[confidence]
    else:
        z = norm.ppf(1 - (alpha/2))
    N = population
    M = error
    numerator = z**2 * sigma**2 * (N / (N-1))
    denominator = M**2 + ((z**2 * sigma**2)/(N-1))
    return numerator/denominator

pop_size = np.array(data)
pop_size = pop_size.size
print("Sample Size from Population: " + str(sampleSize(pop_size)))
print("Sample Size from HWY of DataFile: " + str(hwy.size))
```

```
Sample Size from Population: 334.3842876273181
Sample Size from HWY of DataFile: 234
```

In [11]:

```
# (d) find the min and 75th percentile of “hwy”.

np.percentile(hwy,75)
```

Out[11]:

```
27.0
```

In [12]:

```
# (e) find how many elements of “hwy” greater than the mean of “hwy”.

np.count_nonzero(hwy.mean()> 5)
```

Out[12]:

```
1
```

```
In [13]: # (f) sort “hwy” in descending order.

         arrg = (-np.sort(-hwy))
         arrg
```

```
Out[13]: array([44, 44, 41, 37, 36, 36, 35, 35, 34, 33, 33, 32, 32, 32, 32, 31, 31,
                31, 31, 31, 31, 31, 30, 30, 30, 30, 29, 29, 29, 29, 29, 29, 29, 29,
                29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 28, 28, 28,
                28, 28, 28, 28, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27,
                27, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26,
                26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 26, 25,
                25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 24, 24, 24,
                24, 24, 24, 24, 24, 24, 24, 24, 24, 24, 23, 23, 23, 23, 23, 23, 23,
                22, 22, 22, 22, 22, 22, 22, 21, 21, 20, 20, 20, 20, 20, 20, 20, 20,
                20, 20, 20, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 19, 18,
                18, 18, 18, 18, 18, 18, 18, 18, 18, 17, 17, 17, 17, 17, 17, 17, 17,
                17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17, 17,
                17, 17, 17, 17, 17, 17, 16, 16, 16, 16, 16, 16, 16, 15, 15, 15, 15,
                15, 15, 15, 15, 15, 15, 14, 14, 12, 12, 12, 12, 12], dtype=int64)
```