John Hua - Assignment 1

Problem 1

Import the numpy package under the name np

```
import numpy as np
```

Create a vector or 1D array with 10 zeros and print it

```
a = np.zeros(10)
print(a)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

Find the memory size of this array

```
print("%d bytes" % (a.nbytes))
```

```
80 bytes
```

Problem 2:

Create another vector or 1D array with values ranging from 10 to 20

```
b = np.arange(10, 21)
print(b)
```

```
[10 11 12 13 14 15 16 17 18 19 20]
```

Reverse the created vector (first element becomes last) -- Is there any NumPy method that you can use?

```
print(np.flip(b))
```

```
[20 19 18 17 16 15 14 13 12 11 10]
```

Problem 3:

Create a 3x4 array with random values (standard normal distribution) and find the minimum and maximum values

```
c = np.random.standard_normal(size=(3,4))
print(c)
print("Max: ", np.amax(c))
print("Min: ", np.amin(c))
```

```
[[-0.82913529  0.08771022  1.00036589 -0.38109252]
 [-0.37566942 -0.07447076  0.43349633  1.27837923]
 [-0.63467931  0.50839624  0.21611601 -1.85861239]]
Max:  1.2783792302718682
Min:  -1.8586123861234976
```

Problem 4:

Given the following 1D array, negate all elements which are between 3 and 8, in place. (include both 3 and 8 in conditional statements)

```
# note this will not run without completing the first step of problem 1
Z = np.arange(11)
print(Z)
index = (Z >= 3) & (Z <= 8)
Z[index] = np.negative(Z[index])
print(Z)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10]
[ 0  1  2 -3 -4 -5 -6 -7 -8  9 10]
```

Given the 1D array Z, find the closest value to the given scalar v?

```
Z = np.arange(5,100)
v = np.array(33.2)
idx = (np.abs(Z - v)).argmin()
print(Z)
print("Closest value:", Z[idx])
```

```
[ 5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76
```

```
 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99]
Closest value: 33
```

Subtract the mean of each row of the following matrix

```
np.random.seed(2)
X = np.random.rand(3, 4)
print(X)
```

```
[[0.4359949  0.02592623 0.54966248 0.43532239]
 [0.4203678  0.33033482 0.20464863 0.61927097]
 [0.29965467 0.26682728 0.62113383 0.52914209]]
```

```
# complete the following:
Y = X - X.mean(axis=1, keepdims=True)
print(Y)
```

```
[[ 0.0742684  -0.33580027  0.18793598  0.07359589]
 [ 0.02671225 -0.06332073 -0.18900692  0.22561541]
 [-0.1295348  -0.16236219  0.19194436  0.09995263]]
```

Timing comparison for multiplication of 4 arrays. Find the fastest way to compute the multiplication ABCD. Make sure you report the elapsed time. (hint: you can find relevant information at https://youtu.be/SeBRHg9ZrSs)

Complete the following:

```
A = np.random.random((10000,1000))
B = np.random.random((1000,10000))
C = np.random.random((10000,5))
D = np.random.random((5,1000))
```

```
import time

start = time.time()
np.linalg.multi_dot([A, B, C, D])
end = time.time()

elapsed_time = end - start
print(elapsed_time)
```

```
0.18535566329956055
```

Problem 5

Import and print the file 'parks.csv' (Park Code should be the index column)

```
import pandas as pd
pk = pd.read_csv("parks.csv")
pk.head(10)
```

	Park Code	Park Name	State	Acres	Latitude	Longitude
0	ACAD	Acadia National Park	ME	47390	44.35	-68.21
1	ARCH	Arches National Park	UT	76519	38.68	-109.57
2	BADL	Badlands National Park	SD	242756	43.75	-102.50
3	BIBE	Big Bend National Park	TX	801163	29.25	-103.25
4	BISC	Biscayne National Park	FL	172924	25.65	-80.08
5	BLCA	Black Canyon of the Gunnison National Park	CO	32950	38.57	-107.72
6	BRCA	Bryce Canyon National Park	UT	35835	37.57	-112.18
7	CANY	Canyonlands National Park	UT	337598	38.20	-109.93
8	CARE	Capitol Reef National Park	UT	241904	38.20	-111.17

Print all column names

```
print(pk.columns)
```

```
Index(['park_code', 'park_name', 'state', 'acres', 'latitude', 'longtitude'], dtype='obj
```

Make sure tha all letters are lower case and replace space with _

```
pk = pk.rename(columns={"Park Code": "park_code", "Park Name ": "park_name", "State ": "state
```

Which state has the smallest national park?

```
pk["acres"].min()
pk.sort_values(by="acres", ascending=True).head(1)
#pk[pk["acres"].apply(lambda state: state <= 30000)] # Test example.
```

	park_code	park_name	state	acres	latitude	longtitude
29	HOSP	Hot Springs National Park	AR	5550	34.51	-93.05

Produce a histogram plot that shows the distribution of 'acres'.

```
import matplotlib.pyplot as plt
plot = pk.hist("acres")
plt.xlabel("Acres (per million)")
plt.ylabel("Number of Parks")
```

```
Text(0, 0.5, 'Number of Parks')
```