Programming Assignment #2

Question I (30 pts):

Q1.1. (8 pts) Write a NumPy program to get the following array by using loops
Output:

```
array([[ 1,  2,  3,  4,  5,  6,  7,  8,  9],
       [ 2,  4,  6,  8, 10, 12, 14, 16, 18],
       [ 3,  6,  9, 12, 15, 18, 21, 24, 27],
       [ 4,  8, 12, 16, 20, 24, 28, 32, 36],
       [ 5, 10, 15, 20, 25, 30, 35, 40, 45],
       [ 6, 12, 18, 24, 30, 36, 42, 48, 54],
       [ 7, 14, 21, 28, 35, 42, 49, 56, 63],
       [ 8, 16, 24, 32, 40, 48, 56, 64, 72],
       [ 9, 18, 27, 36, 45, 54, 63, 72, 81]])
```

Q1.2. Replace all odd numbers in array with -1.

Q1.3. By using numpy functions and prog array, get the result.

prog = np.array([1,2,3,4,6]).

Result= array([1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3,
3, 3, 4, 4, 4, 4, 4, 4, 4, 4, 6, 6, 6, 6, 6, 6, 6, 6, 1, 2, 3, 4,
6, 1, 2, 3, 4, 6, 1, 2, 3, 4, 6, 1, 2, 3, 4, 6])

Q1.4. Write a NumPy program to create random 2 array and size of arrays should be 1x15. In your array there should be integer numbers from 0 to 9 and by using numpy function, find common numbers in your array.

Q1.5. Write a NumPy program to generate 50 random number between 0 and 150. If your number bigger than 100, make this number 100 and if your number smaller than 50 make this number 50 by using numpy functions.

Q1.6. Write a NumPy program to generate 16 random integer numbers. Then convert this array to 4x4 array. Repeat this operation and create 3 different 4x4 arrays. By getting summation of these 3 arrays, find the average array, and calculate the maximum, minimum, mean, and standard deviation of each line of matrix values.

Question II (30 pts):
In this question, you are going to write a program that computes matrix operations using NumPy. The program reads values of matrices A, B, and C stored in a file called inputs.txt. This file should be placed under current directory where you have the program. The first line before each matrix contains the number of rows and the number of columns as shown below.

<Input>
4 4
55 55 55 56
66 66 66 67
77 77 77 78
88 88 88 89
4 4
1 2 3 4
2 2 2 2
3 3 3 3
4 4 4 4
4 4
11 12 13 14
22 24 24 25
33 34 35 36
44 45 46 47
<End Input>

As a first step, the program reads data for matrices A, B and C from inputs.txt file and write them into console. A fourth matrix D is generated randomly.

The program will calculate S = (A+B) * Transpose(C) + D) – A and find the maximum element in S. Complete the code given above so that it will produce an output as follows:

<Output>
Reading data from inputs.txt file in current directory

**** Matrix A ****
55 55 55 56
66 20 12 67
77 15 25 78
88 12 13 89

**** Matrix B ****
1 2 3 4
2 2 2 2
3 3 3 3
4 4 4 4

**** Matrix C ****
50 12 75 14
55 24 24 25
33 34 35 36
44 45 46 47

**** Matrix D ****

```
 19   46   14   61
 54   81   91   34
 46   60   95   40
 52   74   95   31

*** Computing S = (A+B) * Transpose(C) + D) - A ***

**** MatriX T1 = (A+B) ****
 56   57   58   60
 68   22   14   69
 80   18   28   81
 92   16   17   93

**** Matrix T2 = Transpose(C) ****
 50   55   33   44
 12   24   34   45
 75   24   35   46
 14   25   36   47

**** Matrix T3 =(A+B) * transpose(C) ****
8674  7340  7976 10517
5680  6329  5966  7869
7450  7529  7148  9425
7369  8177  7523  9921

**** Matrix T4 =(A+B) * transpose(C)+ D ****
8693  7386  7990 10578
5734  6410  6057  7903
7496  7589  7243  9465
7421  8251  7618  9952

**** Matrix S =(A+B) * transpose(C) + D - A ****
8638  7331  7935 10522
5668  6390  6045  7836
7419  7574  7218  9387
7333  8239  7605  9863

Maximum Element in S = 10522

<End Output>
```

Question III (20 pts):

-Write a Pandas program to find the index of the first occurrence of the smallest and largest value of a given series.

nums = pd.Series([8, 30, 39, 12, 29, 8, 25, 16, 22, 32, 15, 37, 35, 39, 26, 6])

-Write a program that will check the series equal or not one by one.
nums1 = pd.Series([1, 8, 7, 5, 6, 5, 3, 4, 7, 1])
nums2 = pd.Series([1, 8, 7, 5, 6, 5, 3, 4, 7, 1])

-Write a Pandas program to select the 'name' and 'score' columns from the following DataFrame.

-Write a Pandas program to select the rows where the number of attempts in the examination is greater than 2.
-Write a Pandas program to select the rows where the score is missing, i.e. is NaN.

exam_data = {'name': ['Ahmet', 'Mehmet', 'Zehra', 'Osman', 'Fatma', 'Mert', 'Cem', 'Kerem', 'Sevgi', 'Pınar'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

Question IV(20 pts):
The objective of this question is to learn hash functions and collision handling. Analyze the HashTable class code given in https://runestone.academy/runestone/books/published/pythonds/SortSearch/Hashing.html#implementing-the-map-abstract-data-type.

Make the following changes in the HashTable class code:

- Change the hash function so that it can accept string-type keys by using the method given in Listing 1.

- In the hash table map implementation, the hash table size was chosen to be 11. If the table gets full, this needs to be increased. Re-implement the put method so that the table will automatically resize itself when the loading factor reaches a predetermined value (you can decide the value based on your assessment of load versus performance).

- Implement chaining as a rehash technique.

Test the program with the data.

Key	**Value**
Abandon	TERK ETMEK
Ability	YETENEK
Able	MUKTEDİR
Aboard	(bir taşıtın)İÇİNDE OLMAK
About	1.HAKKINDA 2.YAKLAŞIK OLARAK
Above	YUKARIDAKİ
Abroad	YURT DIŞI
Absence	YOKLUK
Absent	1.YOK 2.EKSİK
Absolute	MUTLAK, KESİN
Absurd	SAÇMA
Accept	KABUL ETMEK
Accident	KAZA
Accommodate	YERLEŞTİRMEK
Accommodation	KONAKLAMA YERİ
Accompany	EŞLİK ETMEK
According To	GÖRE
Account	HESAP

Accurate	DOĞRU, HATASIZ
Accuse	SUÇLAMAK
Ache	AĞRI
Acquaint	TANIMAK,BİLMEK
Across	1.BİR UÇTAN DİĞERİNE 2.DİĞER TARAFTA
Act	1.DAVRANIŞ 2.DAVRANMAK
Active	ETKİN, FAAL
Actor	ERKEK OYUNCU
Actress	KADIN OYUNCU
Actual	GERÇEK
Add	TOPLAMAK,EKLEMEK
Address	ADRES
Administration	İDARE
Admire	BEĞENMEK,HAYRAN OLMAK
Admit	1.KABUL ETMEK 2.İZİN VERMEK
Adult	YETİŞKİN
Advance	1.İLERİ 2.AVANS
Advanced	GELİŞMİŞ
Advantage	AVANTAJ
Adventure	MACERA
Advertise	REKLAM YAPMAK, İLAN VERMEK
Advice	TAVSİYE
Advise	TAVSİYE ETMEK
Aerial	ANTEN
Aeroplane	UÇAK
Affair	1.OLAY 2.İŞ 3.İLİŞKİ
Affect	ETKİLEMEK
Afford	SATIN ALMA GÜCÜ OLMAK
Afraid	KORKMAK
After	SONRA

Example:
H=HashTable()
H[“Abandon”] = “TERK ETMEK”
H[“Ability”] =”YETENEK”
H[“Able”] =”MUKTEDİR”
.
.

H.slots
H.data

H[“Abandon”]
“TERK ETMEK”
H[“Able”]
”MUKTEDİR”