Practical – 2

Aim :

1. **Create a 10x10 array with random values and find the minimum and maximum values**

Program :

```
import numpy as np

import random

arr = np.random.rand(10,10)

print(f"Maximum of given array is {np.max(arr)}.")

print(f"Minimum of given array is {np.min(arr)}.")
```

Output :

```
Maximum of given array is 0.9970711767056688.
Minimum of given array is 0.018700461153001946.
```

2. **Write a NumPy program to create a 8x8 matrix and fill it with a checkerboard pattern.**

Checkerboard pattern:
[[0 1 0 1 0 1 0 1]
[1 0 1 0 1 0 1 0]
[0 1 0 1 0 1 0 1]
[1 0 1 0 1 0 1 0]
[0 1 0 1 0 1 0 1]
[1 0 1 0 1 0 1 0]
[0 1 0 1 0 1 0 1]
[1 0 1 0 1 0 1 0]]

Program :

```
checker_board = np.zeros((8,8),dtype=int)

checker_board[1::2,::2] = 1

checker_board[::2,1::2] = 1

print(checker_board)
```

Output :

```
[[0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]
 [0 1 0 1 0 1 0 1]
 [1 0 1 0 1 0 1 0]]
```

3. **Write a NumPy program to append values to the end of an array.**

Expected Output: Original array: [10, 20, 30]

Program :

```
arr1 = np.array([int(i) for i in input("Enter values for array - 1 (Space-Separated) : ").split(' ')])

print(f"Original array : {arr1}")

arr2 = np.array([int(i) for i in input("Enter values for array - 2 (Space-Separated) : ").split(' ')])

arr = np.append(arr1,arr2)

print(f"Array after append : {arr}")
```

Output :

```
Enter values for array - 1 (Space-Separated) :  10 20 30
Original array : [10 20 30]
Enter values for array - 2 (Space-Separated) :  40 50 60 70 80 90
Array after append : [10 20 30 40 50 60 70 80 90]
```

4. **Write a NumPy program to calculate the arithmetic means of corresponding elements of two given arrays of same size.**

Program :

```
array1 = np.array([float(i) for i in input("Enter values for array - 1 (Space-Separated) : ").split(' ')])

array2 = np.array([float(i) for i in input("Enter values for array - 2 (Space-Separated) : ").split(' ')])

mean_array = (array1 + array2) / 2

print("Array 1:", array1)

print("Array 2:", array2)
```

```
print("Arithmetic Means:", mean_array)
```

Output :

```
Enter values for array - 1 (Space-Separated) : 10 15 20
Enter values for array - 2 (Space-Separated) : 20 19 32
Array 1: [10. 15. 20.]
Array 2: [20. 19. 32.]
Arithmetic Means: [15. 17. 26.]
```

5. **Write a Numpy program to test whether a numpy array is faster than a Python list or not.**

Program :

```
import time

my_list = [np.random.randint(1000) for i in range(10**5)]

my_array = np.arange(10**5)

start_time = time.time()

np.sort(my_array)

end_time = time.time()

print(f"Time taken for sorting array is {end_time - start_time}.")

start_time = time.time()

my_list.sort()

end_time = time.time()

print(f"Time taken for sorting list is {end_time - start_time}.")
```

Output :

```
Time taken for sorting array is 0.0.
Time taken for sorting list is 0.018715858459472656.
```

6. **Write a Python function to create an array of 10 values from 1 to 5 evenly spaced.**

Program :

```
array = np.linspace(1, 5, 10)

print("Evenly spaced array:", array)
```

Output :

```
Evenly spaced array: [1.         1.44444444 1.88888889 2.33333333 2.77777778 3.22222222
 3.66666667 4.11111111 4.55555556 5.        ]
```

7. Given a 1D array, negate all elements which are between 3 and 8, in place.

Program :

```
arr = np.array([float(i) for i in input("Enter values for Array (Space-Separated) : ").split(' ')])

print(f"Original array : {arr}")

indices = np.where((arr > 3) & (arr < 8))

arr[indices] = -arr[indices]

print(f"Array after negation : {arr}")
```

Output :

```
Enter values for Array (Space-Separated) : 1 2 8 5 4 6 7
Original array : [1. 2. 8. 5. 4. 6. 7.]
Array after negation : [ 1.  2.  8. -5. -4. -6. -7.]
```

8. How to compute ((A+B)*(-A/2)) in place (without copy)? where A = [1. 1. 1.] and B=[2. 2. 2.]

Program :

```
a = np.ones(3)*1

b = np.ones(3)*2

np.add(a,b,out=b)

np.divide(a,2,out=a)

np.negative(a,out=a)

np.multiply(a,b,out=a)
```

Output :

```
array([-1.5, -1.5, -1.5])
```

9. Create random vector of size 10 and replace the maximum value by 0.

Program :

```
random_vector = np.random.rand(10)

print(f"Random vector before replacing maximum value : {random_vector}")

random_vector[random_vector == np.max(random_vector)] = 0

print(f"Random vector after replacing maximum value : {random_vector}")
```

Output :

```
Random vector before replacing maximum value : [0.10142025 0.20092468 0.47020336 0.55690621 0.7940462  0.09472048
 0.86424212 0.08432712 0.34702337 0.48902755]
Random vector after replacing maximum value : [0.10142025 0.20092468 0.47020336 0.55690621 0.7940462  0.09472048
 0.         0.08432712 0.34702337 0.48902755]
```

10. Write a NumPy program to subtract the mean of each row of a given matrix.

Program :

```
size = int(input("Enter the size of the array : "))
arr = np.array([[float(j) for j in input(f"Enter row - {i + 1} for Array (Space-Separated) : ").split(' ')] for i in range(size)])

print(f"Original array : \n{arr}")

arr = arr - np.mean(arr,axis=1,keepdims=True)
print(f"Updated Array : {arr}")
```

Output :

```
Enter the size of the array : 3
Enter row - 1 for Array (Space-Separated) : 10 20 30
Enter row - 2 for Array (Space-Separated) : 1 2 9
Enter row - 3 for Array (Space-Separated) : 5 6 8
Original array :
[[10. 20. 30.]
 [ 1.  2.  9.]
 [ 5.  6.  8.]]
Updated Array : [[-10.          0.         10.        ]
 [ -3.         -2.          5.        ]
 [ -1.33333333  -0.33333333   1.66666667]]
```