**Numpy**

1.Write a NumPy program to test whether none of the elements of a given array is zero.

```
import numpy as np
x = np.array([1, 2, 3, 4])
```

```
np.all(x)
```

2.Write a NumPy program to test whether any of the elements of a given array is non-zero.

```
np.any(x)
```

3.Given array is finiteness (not infinity or not a number)

```
np.isfinite(a)
```

4.to test element-wise for positive or negative infinity.

```
np.isinf(a)
```

```
np.isnan(a) to test for NaN in any given number
```

```
np.iscomplex(a)
```

```
np.isreal(a)
```

```
np.isscalar(3.1)
```

5.Write a NumPy program to create an array with the values 1, 7, 13, 105 and determine the size of the memory occupied by the array.

```
import numpy as np
X = np.array([1, 7, 13, 105])
print("Original array:")
```

```
print(X)
print("Size of the memory occupied by the said array:")
print("%d bytes" % (X.size * X.itemsize))
```

6.Write a NumPy program to create an array of 10 zeros, 10 ones, 10 fives

```
import numpy as np
array=np.zeros(10)
print("An array of 10 zeros:")
print(array)
array=np.ones(10)
print("An array of 10 ones:")
print(array)
array=np.ones(10)*5
print("An array of 10 fives:")
print(array)
```

7. Write a NumPy program to create an array of the integers from 30 to 70.

```
import numpy as np
array=np.arange(30,71)
```

8.Write a NumPy program to create an array of all the even integers from 30 to 70.

```
array=np.arange(30,71,2)
```

9. Write a NumPy program to create a 3x3 identity matrix

```
import numpy as np
array_2D=np.identity(3)
```

10.Write a NumPy program to generate a random number between 0 and 1.

```
rand_num = np.random.normal(0,1,1)
```

11.Write a NumPy program to generate an array of 15 random numbers from a standard normal distribution.

```
rand_num = np.random.normal(0,1,15)
```

12.Write a NumPy program to create a vector with values ranging from 15 to 55 and print all values except the first and last.

```
import numpy as np
v = np.arange(15,55)

print(v[1:-1])
```

13.Write a NumPy program to create a 3X4 array using and iterate over it.

```
import numpy as np
a = np.arange(10,22).reshape((3, 4))
print("Original array:")
print(a)
print("Each element of the array is:")
for x in np.nditer(a):
  print(x,end=" ")
```

14.Write a NumPy program to create a vector of length 10 with values evenly distributed between 5 and 50.

```
import numpy as np
v = np.linspace(10, 49, 5)
```

15.Write a NumPy program to create a vector with values from 0 to 20 and change the sign of the numbers in the range from 9 to 15.

```
import numpy as np
```

```
x = np.arange(21)

x[(x >= 9) & (x <= 15)] *= -1
```

16.Write a NumPy program to create a 3x4 matrix filled with values from 10 to 21.

```
import numpy as np
m= np.arange(10,22).reshape((3, 4))
print(m)
```

17.Write a NumPy program to create a 3x3 identity matrix, i.e. diagonal elements are 1,the rest are 0.

```
import numpy as np
x = np.eye(3)
print(x)
```

18.Write a NumPy program to create a 10x10 matrix, in which the elements on the borders will be equal to 1, and inside 0.

```
import numpy as np
x = np.ones((10, 10))
x[1:-1, 1:-1] = 0
```

19.Write a NumPy program to convert a given array into a list and then convert it into a list again.

```
import numpy as np
a = [[1, 2], [3, 4]]
x = np.array(a)
a2 = x.tolist()
print(a == a2)
```

## NUMPY SORTING AND SEARCHING

1.Write a NumPy program to sort a given array of shape 2 along the first axis, last axis and on flattened array.

```
import numpy as np
a = np.array([[10,40],[30,20]])
print("Original array:")
print(a)
print("Sort the array along the first axis:")
print(np.sort(a, axis=0))
print("Sort the array along the last axis:")
print(np.sort(a))
print("Sort the flattened array:")
print(np.sort(a, axis=None))
```

Sample Output:

```
Original array:
[[10 40]
 [30 20]]
Sort the array along the first axis:
[[10 20]
 [30 40]]
Sort the array along the last axis:
[[10 40]
 [20 30]]
Sort the flattened array:
[10 20 30 40]
```

2.Write a NumPy program to create a structured array from given student name, height, class and their data types. Now sort the array on height.

```
import numpy as np
```

```
data_type = [('name', 'S15'), ('class', int), ('height', float)]
students_details = [('James', 5, 48.5), ('Nail', 6, 52.5),('Paul', 5, 42.10), ('Pit', 5, 40.11)]
# create a structured array
students = np.array(students_details, dtype=data_type)
print("Original array:")
print(students)
print("Sort by height")
print(np.sort(students, order='height'))
```

3.Write a NumPy program to create a structured array from given student name, height, class and their data types. Now sort by class, then height if class are equal.

```
import numpy as np
data_type = [('name', 'S15'), ('class', int), ('height', float)]
students_details = [('James', 5, 48.5), ('Nail', 6, 52.5),('Paul', 5, 42.10), ('Pit', 5, 40.11)]
# create a structured array
students = np.array(students_details, dtype=data_type)
print("Original array:")
print(students)
print("Sort by class, then height if class are equal:")
print(np.sort(students, order=['class', 'height']))
```

4.Write a NumPy program to sort the student id with increasing height of the students from given students id and height. Print the integer indices that describes the sort order by multiple columns and the sorted data.

```
import numpy as np
student_id = np.array([1023, 5202, 6230, 1671, 1682, 5241, 4532])
student_height = np.array([40., 42., 45., 41., 38., 40., 42.0])
#Sort by studen_id then by student_height
indices = np.lexsort((student_id, student_height))
```

```
print("Sorted indices:")
print(indices)
print("Sorted data:")
for n in indices:
  print(student_id[n], student_height[n])
```

5.Write a NumPy program to get the indices of the sorted elements of a given array.

```
import numpy as np
student_id = np.array([1023, 5202, 6230, 1671, 1682, 5241, 4532])
print("Original array:")
print(student_id)
i = np.argsort(student_id)
print("Indices of the sorted elements of a given array:")
print(i)
```

6.Write a NumPy program to partition a given array in a specified position and move all the smaller elements values to the left of the partition, and the remaining values to the right, in arbitrary order (based on random choice)

```
import numpy as np
nums = np.array([70, 50, 20, 30, -11, 60, 50, 40])
print("Original array:")
print(nums)
print("\nAfter partitioning on 4 the position:")
print(np.partition(nums, 4))
```

7.Write a NumPy program to sort the specified number of elements from beginning of a given array.

```
import numpy as np
```

```
nums =  np.random.rand(10)
print("Original array:")
print(nums)
print("\nSorted first 5 elements:")
print(nums[np.argpartition(nums,range(5))])
```

NUMPY RANDOM

1.Write a NumPy program to generate five random numbers from the normal distribution

```
import numpy as np
x = np.random.normal(size=5)
print(x)
```

2.Write a NumPy program to generate six random integers between 10 and 30.

```
import numpy as np
x = np.random.randint(low=10, high=30, size=6)
print(x)
```

3.Write a NumPy program to create a 3x3x3 array with random values.

```
import numpy as np
x = np.random.random((3,3,3))
print(x)
```

4.Write a NumPy program to create a 5x5 array with random values and find the minimum and maximum values.

```
import numpy as np
x = np.random.random((5,5))
print("Original Array:")
print(x)
```

```
xmin, xmax = x.min(), x.max()
print("Minimum and Maximum Values:")
print(xmin, xmax)
```

5.Write a NumPy program to create a random 10x4 array and extract the first five rows of the array and store them into a variable

```
import numpy as np
x = np.random.rand(10, 4)
print("Original array: ")
print(x)
y= x[:5, :]
print("First 5 rows of the above array:")
print(y)
```

6.Write a NumPy program to shuffle numbers between 0 and 10 (inclusive).

```
import numpy as np
x = np.arange(10)
np.random.shuffle(x)
print(x)
print("Same result using permutation():")
print(np.random.permutation(10))
```

7.Write a NumPy program to normalize a 3x3 random matrix.

```
import numpy as np
x= np.random.random((3,3))
print("Original Array:")
print(x)
xmax, xmin = x.max(), x.min()
x = (x - xmin)/(xmax - xmin)
print("After normalization:")
print(x)
```

8.Write a NumPy program to create a random vector of size 10 and sort it.

```
import numpy as np
x = np.random.random(10)
print("Original array:")
print(x)
x.sort()
print("Sorted array:")
print(x)
```

9.Write a NumPy program to find the nearest value from a given value in an array.

```
import numpy as np
x = np.random.uniform(1, 12, 5)
v = 4
n = x.flat[np.abs(x - v).argmin()]
print(n)
```

10.Write a NumPy program to check two random arrays are equal or not.

```
import numpy as np
x = np.random.randint(0,2,6)
print("First array:")
print(x)
y = np.random.randint(0,2,6)
print("Second array:")
print(y)
print("Test above two arrays are equal or not!")
array_equal = np.allclose(x, y)
print(array_equal)
```

11.Write a NumPy program to create random vector of size 15 and replace the maximum value by -1.

```
import numpy as np
```

```
x = np.random.random(15)
print("Original array:")
print(x)
x[x.argmax()] = -1
print("Maximum value replaced by -1:")
print(x)
```

12.Write a NumPy program to find point by point distances of a random vector with shape (10,2) representing coordinates.

```
import numpy as np
a= np.random.random((10,2))
x,y = np.atleast_2d(a[:,0], a[:,1])
d = np.sqrt( (x-x.T)**2 + (y-y.T)**2)
print(d)
```

13.Write a NumPy program to find the closest value (to a given scalar) in an array.

```
import numpy as np
x = np.arange(100)
print("Original array:")
print(x)
a = np.random.uniform(0,100)
print("Value to compare:")
print(a)
index = (np.abs(x-a)).argmin()
print(x[index])
```

14.Write a NumPy program to get the n largest values of an array

```
import numpy as np
x = np.arange(10)
print("Original array:")
print(x)
np.random.shuffle(x)
```

```
n = 1
print (x[np.argsort(x)[-n:]])
```