1.a)Write python programs to implement indexing, slicing and splitting, iterating a list.

**import** numpy **as** np

array1d **=** np**.**array([1, 2, 3, 4, 5, 6,7,8])

array2d **=** np**.**array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

array3d **=** np**.**array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])

print(array1d)

print("---------Indexing operation ----------")

print("---------Indexing on 1D----------")

print(array1d[0]) *# Get first value*

print(array1d[**-**1]) *# Get last value*

print(array1d[3]) *# Get 4th value from first*

print(array1d[**-**5]) *# Get 5th value from last*

print(array1d[[0,2,1]]) *#Get multiple values from array*

print("---------Indexing on 2D----------")

print(array2d)

print(array2d[0, 0]) *# Get first row first col*

print(array2d[0, 1]) *# Get first row second col*

print(array2d[0, 2]) *# Get first row third col*

print(array2d[0, 1]) *# Get first row second col*

print(array2d[**-**1, 1]) *# Get second row second col*

print(array2d[2, **-**2]) *# Get third row second col*

print(array2d[[0,1,2],[1,0,2]])*#Get multiple values from array*

print(array2d[[0,1,2],[**-**1,0,**-**3]])*#Get multiple values from array*

print(array2d[0,2]**+**array2d[1,1])*#adding the elements on array*

print("---------Indexing on 3D----------")

print(array3d)

print(array3d[0, 1, 2])

print(array3d[0, **-**2, 1])

print(array3d[0, 0, 2])

print(array3d[[0,1,0],[1,0,1],[1,0,2]])

print(array3d[[0,1,0,1],[1,0,1,**-**1],[1,0,2,**-**3]])

print(array3d[1,**-**1,2]**+**array3d[0, **-**1, **-**3])

print("---------Slicing operation----------")

print("---------Slicing on 1D----------")

print(array1d)

print(array1d[4:]) *# From index 4 to last index*

print(array1d[:4]) *# From index 0 to 4 index*

print(array1d[4:7]) *# From index 4(included) up to index 7(excluded)*

print(array1d[:**-**1]) *# Excluded last element*

print(array1d[:**-**2]) *# Up to second last index(negative index)*

print(array1d[::**-**1]) *# From last to first in reverse order(negative step)*

print(array1d[::**-**2]) *# All odd numbers in reversed order*

print(array1d[**-**2::**-**2]) *# All even numbers in reversed order*

print(array1d[::]) *# All elements*

array1d[0]**=**10 *#modifying the element in array*

print(array1d)

print("---------Slicing on 2D----------")

array2d **=** np**.**array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print(array2d)

print(array2d[:, 0:2]) *# 2nd and 3rd col*

print(array2d[1:3, 0:3]) *# 2nd and 3rd row*

print(array2d[**-**1::**-**1, **-**1::**-**1]) *# Reverse an array*

print(array2d[0:2, 0:2:2]) *#start,stop,step*

print(array2d[1,0:2:2])

print("---------Slicing on 3D----------")

array3d **=** np**.**array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])

print(array3d)

print(array3d[0,1,2])

print(array3d[1,**-**1,2])

print(array3d[1,0:1,0:2])

print(array3d[0:2,0:2,0:2])

print(array3d[0:2,1,**-**3])

print("---------Splitting operations----------")

print("---------Splitting on 1D----------")

print(array1d)

print("-------splitting on Horizontal---------")

print(np**.**hsplit(array1d,2))*#split divides into equal division*

*#or*

np**.**split(array1d,2)

*#np.split(array1d,3) error*

print(np**.**array\_split(array1d,3)

*#splitting on vertical on 1D not possible it requires 2D or More Dimensional*

print("---------Splitting on 2D----------")

print(array2d)

print("---------splitting on horizontal on 2D (column-wise)-----")

print(np**.**hsplit(array2d,3))*#split divides into equal division*

*#or*

print(np**.**split(array2d,3,axis**=**1))

*#np.split(array2d,2) error use array\_split()*

print(np**.**array\_split(array2d,2,axis**=**1))

print(np**.**array\_split(array2d,4,axis**=**1))

print("-"**\***10)

print("---------splitting on vertical on 2D (row-wise)---------")

print(np**.**vsplit(array2d,3))*#split divides into equal division*

*#or*

print(np**.**split(array2d,3,axis**=**0))

print(np**.**array\_split(array2d,2,axis**=**0))

print(np**.**array\_split(array2d,4,axis**=**0))

print("--------------splitting on 3D--------------")

print(array3d)

print("-------------split on horizontal----------------")

print(np**.**hsplit(array3d,2))*#split divides into equal division*

*#or*

print(np**.**split(array3d,2,axis**=**1))

*#print(np.split(array3d,3,axis=1)) error not in equal division so go for array\_split()*

print(np**.**array\_split(array3d,3,axis**=**1))

print(np**.**array\_split(array3d,4,axis**=**1))

print("---------split on vertical------------ ")

print(np**.**vsplit(array3d,2))*#split divides into equal division*

*#or*

print(np**.**split(array3d,2,axis**=**0))

*#print(np.split(array3d,3,axis=1)) error not in equal division so go for array\_split()*

print(np**.**array\_split(array3d,3,axis**=**0))

print(np**.**array\_split(array3d,4,axis**=**0))

print("-----Iterating operations------")

print("-----Iterating Arrays Using nditer()--------")

print("-----Iterating operations on 1D------")

print(array1d)

**for** x **in** np**.**nditer(array1d,flags**=**['buffered'],op\_dtypes**=**['S']):

print(x)

print("-----Iterating operations on 2D------")

print(array2d)

**for** x **in** np**.**nditer(array2d[:,::2]):

print(x)

print("-----Iterating operations on 3D------")

print(array3d)

**for** x **in** np**.**nditer(array3d[:,::2,2]):

print(x)

print("---------modifying array values----------")

**for** x **in** np**.**nditer(array3d):

x**=**5**\***x

print(x)

print("----------using ndenumerate()--------")

for idx,x in np.ndenumerate(array3d):

print(idx,x)

Output:

[1 2 3 4 5 6 7 8]

---------Indexing operation ----------

---------Indexing on 1D----------

1

8

4

4

[1 3 2]

---------Indexing on 2D----------

[[1 2 3]

[4 5 6]

[7 8 9]]

1

2

3

2

8

8

[2 4 9]

[3 4 7]

8

---------Indexing on 3D----------

[[[ 1 2 3]

[ 4 5 6]]

[[ 7 8 9]

[10 11 12]]]

6

2

3

[5 7 6]

[ 5 7 6 10]

16

---------Slicing operation----------

---------Slicing on 1D----------

[1 2 3 4 5 6 7 8]

[5 6 7 8]

[1 2 3 4]

[5 6 7]

[1 2 3 4 5 6 7]

[1 2 3 4 5 6]

[8 7 6 5 4 3 2 1]

[8 6 4 2]

[7 5 3 1]

[1 2 3 4 5 6 7 8]

[10 2 3 4 5 6 7 8]

---------Slicing on 2D----------

[[1 2 3]

[4 5 6]

[7 8 9]]

[[1 2]

[4 5]

[7 8]]

[[4 5 6]

[7 8 9]]

[[9 8 7]

[6 5 4]

[3 2 1]]

[[1]

[4]]

[4]

---------Slicing on 3D----------

[[[ 1 2 3]

[ 4 5 6]]

[[ 7 8 9]

[10 11 12]]]

6

12

[[7 8]]

[[[ 1 2]

[ 4 5]]

[[ 7 8]

[10 11]]]

[ 4 10]

---------Splitting operations----------

---------Splitting on 1D----------

[10 2 3 4 5 6 7 8]

-------splitting on Horizontal---------

[array([1, 2, 3, 4]), array([5, 6, 7, 8])]

[array([1, 2, 3]), array([4, 5, 6]), array([7, 8])]

---------Splitting on 2D----------

[[1 2 3]

[4 5 6]

[7 8 9]]

---------splitting on horizontal on 2D (column-wise)-----

[array([[1],

[4],

[7]]), array([[2],

[5],

[8]]), array([[3],

[6],

[9]])]

[array([[1],

[4],

[7]]), array([[2],

[5],

[8]]), array([[3],

[6],

[9]])]

----------

---------splitting on vertical on 2D (row-wise)---------

[array([[1, 2, 3]]), array([[4, 5, 6]]), array([[7, 8, 9]])]

[array([[1, 2, 3]]), array([[4, 5, 6]]), array([[7, 8, 9]])]

[array([[1, 2, 3],

[4, 5, 6]]), array([[7, 8, 9]])]

[array([[1, 2, 3]]), array([[4, 5, 6]]), array([[7, 8, 9]]), array([], shape=(0, 3), dtype=int32)]

--------------splitting on 3D--------------

[[[ 1 2 3]

[ 4 5 6]]

[[ 7 8 9]

[10 11 12]]]

-------------split on horizontal----------------

[array([[[1, 2, 3]],

[[7, 8, 9]]]), array([[[ 4, 5, 6]],

[[10, 11, 12]]])]

[array([[[1, 2, 3]],

[[7, 8, 9]]]), array([[[ 4, 5, 6]],

[[10, 11, 12]]])]

[array([[[1, 2, 3]],

[[7, 8, 9]]]), array([[[ 4, 5, 6]],

[[10, 11, 12]]]), array([], shape=(2, 0, 3), dtype=int32)]

[array([[[1, 2, 3]],

[[7, 8, 9]]]), array([[[ 4, 5, 6]],

[[10, 11, 12]]]), array([], shape=(2, 0, 3), dtype=int32), array([], shape=(2, 0, 3), dtype=int32)]

---------split on vertical------------

[array([[[1, 2, 3],

[4, 5, 6]]]), array([[[ 7, 8, 9],

[10, 11, 12]]])]

[array([[[1, 2, 3],

[4, 5, 6]]]), array([[[ 7, 8, 9],

[10, 11, 12]]])]

[array([[[1, 2, 3],

[4, 5, 6]]]), array([[[ 7, 8, 9],

[10, 11, 12]]]), array([], shape=(0, 2, 3), dtype=int32)]

[array([[[1, 2, 3],

[4, 5, 6]]]), array([[[ 7, 8, 9],

[10, 11, 12]]]), array([], shape=(0, 2, 3), dtype=int32), array([], shape=(0, 2, 3), dtype=int32)]

-----Iterating operations------

-----Iterating Arrays Using nditer()--------

-----Iterating operations on 1D------

[10 2 3 4 5 6 7 8]

b'10'

b'2'

b'3'

b'4'

b'5'

b'6'

b'7'

b'8'

-----Iterating operations on 2D------

[[1 2 3]

[4 5 6]

[7 8 9]]

1

3

4

6

7

9

-----Iterating operations on 3D------

[[[ 1 2 3]

[ 4 5 6]]

[[ 7 8 9]

[10 11 12]]]

3

9

---------modifying array values----------

5

10

15

20

25

30

35

40

45

50

55

60

----------using ndenumerate()--------

(0, 0, 0) 1

(0, 0, 1) 2

(0, 0, 2) 3

(0, 1, 0) 4

(0, 1, 1) 5

(0, 1, 2) 6

(1, 0, 0) 7

(1, 0, 1) 8

(1, 0, 2) 9

(1, 1, 0) 10

(1, 1, 1) 11

(1, 1, 2) 12

1.b) Write a python program on Indexing, Slicing, Splitting & Iterating on Dataset.

DATASET.csv file

65.78331 ,112.9925 ,345.89 ,234.23

71.51521 ,136.4873 ,346.89, 235.23

69.39874 ,153.0269 ,347.89 ,236.23

68.2166 ,142.3354 ,348.89, 237.23

67.78781 ,144.2971 ,349.89 ,238.23

68.69784 ,123.3024, 350.89, 239.23

69.80204 ,141.4947 ,351.89 ,240.23

70.01472, 136.4623 ,352.89, 241.23

67.90265 ,112.3723 ,353.89, 242.23

66.78236, 120.6672, 354.89 ,243.23

import numpy as np

dataset =np.genfromtxt('C:/Users/JOSHITH/Desktop/DATASET1.csv',delimiter=',')

print(dataset)

print("The rows & columns on dataset:", dataset.shape)

print("---------Perform Indexing operation on Dataset------ ")

print(dataset[0]) # index single element in outermost dimension

print(dataset[-1]) # index in reversed order in outermost dimension

print(dataset[1, 1]) # index single element in two-dimensional data

print(dataset[-1, -1]) # index in reversed order in two-dimensional data

print(dataset[8,3])

print(dataset[[0,7,3,9],[1,-1,2,3]])

print(dataset[0,3]+dataset[2,-1])

print(dataset[dataset>70])#booloean operation

print("---------Perform Slicing operation on Dataset------ ")

print(dataset[1:3]) # rows 1 and 2

print(dataset[:2, :2]) # 2x2 subset of the data

print(dataset[-1, ::-1]) # last row with elements reversed

print(dataset[-5:-1, :6:2])# last 4 rows,every other element up to index 6

print("---------Perform Splitting operation on Dataset------ ")

print("-----------Horizontal splitting-----------")

print(np.hsplit(dataset,2)) # split horizontally in 3 equal lists

print("-----------vertical splitting-----------")

print(np.vsplit(dataset, 2)) # split vertically in 2 equal lists

print("---------Perform Iterating operation on Dataset------ ")

print("----------using nditer()--------")

for x in np.nditer(dataset):

print(x)

print("----------using ndenumerate()--------")

for idx,x in np.ndenumerate(dataset):

print(idx,x)

Output:

[[ 65.78331 112.9925 345.89 234.23 ]

[ 71.51521 136.4873 346.89 235.23 ]

[ 69.39874 153.0269 347.89 236.23 ]

[ 68.2166 142.3354 348.89 237.23 ]

[ 67.78781 144.2971 349.89 238.23 ]

[ 68.69784 123.3024 350.89 239.23 ]

[ 69.80204 141.4947 351.89 240.23 ]

[ 70.01472 136.4623 352.89 241.23 ]

[ 67.90265 112.3723 353.89 242.23 ]

[ 66.78236 120.6672 354.89 243.23 ]]

The rows & columns on dataset: (10, 4)

---------Perform Indexing operation on Dataset------

[ 65.78331 112.9925 345.89 234.23 ]

[ 66.78236 120.6672 354.89 243.23 ]

136.4873

243.23

242.23

[112.9925 241.23 348.89 243.23 ]

470.46

[112.9925 345.89 234.23 71.51521 136.4873 346.89 235.23

153.0269 347.89 236.23 142.3354 348.89 237.23 144.2971

349.89 238.23 123.3024 350.89 239.23 141.4947 351.89

240.23 70.01472 136.4623 352.89 241.23 112.3723 353.89

242.23 120.6672 354.89 243.23 ]

---------Perform Slicing operation on Dataset------

[[ 71.51521 136.4873 346.89 235.23 ]

[ 69.39874 153.0269 347.89 236.23 ]]

[[ 65.78331 112.9925 ]

[ 71.51521 136.4873 ]]

[243.23 354.89 120.6672 66.78236]

[[ 68.69784 350.89 ]

[ 69.80204 351.89 ]

[ 70.01472 352.89 ]

[ 67.90265 353.89 ]]

---------Perform Splitting operation on Dataset------

-----------Horizontal splitting-----------

[array([[ 65.78331, 112.9925 ],

[ 71.51521, 136.4873 ],

[ 69.39874, 153.0269 ],

[ 68.2166 , 142.3354 ],

[ 67.78781, 144.2971 ],

[ 68.69784, 123.3024 ],

[ 69.80204, 141.4947 ],

[ 70.01472, 136.4623 ],

[ 67.90265, 112.3723 ],

[ 66.78236, 120.6672 ]]), array([[345.89, 234.23],

[346.89, 235.23],

[347.89, 236.23],

[348.89, 237.23],

[349.89, 238.23],

[350.89, 239.23],

[351.89, 240.23],

[352.89, 241.23],

[353.89, 242.23],

[354.89, 243.23]])]

-----------vertical splitting-----------

[array([[ 65.78331, 112.9925 , 345.89 , 234.23 ],

[ 71.51521, 136.4873 , 346.89 , 235.23 ],

[ 69.39874, 153.0269 , 347.89 , 236.23 ],

[ 68.2166 , 142.3354 , 348.89 , 237.23 ],

[ 67.78781, 144.2971 , 349.89 , 238.23 ]]), array([[ 68.69784, 123.3024 , 350.89 , 239.23 ],

[ 69.80204, 141.4947 , 351.89 , 240.23 ],

[ 70.01472, 136.4623 , 352.89 , 241.23 ],

[ 67.90265, 112.3723 , 353.89 , 242.23 ],

[ 66.78236, 120.6672 , 354.89 , 243.23 ]])]

---------Perform Iterating operation on Dataset------

----------using nditer()--------

65.78331

112.9925

345.89

234.23

71.51521

136.4873

346.89

235.23

69.39874

153.0269

347.89

236.23

68.2166

142.3354

348.89

237.23

67.78781

144.2971

349.89

238.23

68.69784

123.3024

350.89

239.23

69.80204

141.4947

351.89

240.23

70.01472

136.4623

352.89

241.23

67.90265

112.3723

353.89

242.23

66.78236

120.6672

354.89

243.23

----------using ndenumerate()--------

(0, 0) 65.78331

(0, 1) 112.9925

(0, 2) 345.89

(0, 3) 234.23

(1, 0) 71.51521

(1, 1) 136.4873

(1, 2) 346.89

(1, 3) 235.23

(2, 0) 69.39874

(2, 1) 153.0269

(2, 2) 347.89

(2, 3) 236.23

(3, 0) 68.2166

(8, 3) 242.23

(9, 0) 66.78236

(9, 1) 120.6672

(9, 2) 354.89

(9, 3) 243.23

2.a) Write python programs to implement statistical functions like Mean, Median, Variance, and Standard Deviation using numpy.

Program:

import numpy as np

a = np.array([[2,10,20],[80,43,31],[22,43,10]])

print("The original array:\n")

print(a)

print("\nThe minimum element among the array:",np.amin(a))

print("The maximum element among the array:",np.amax(a))

print("\nThe minimum elements with axis=0 ",np.amin(a,0))

print("The maximum elements with axis=0 ",np.amax(a,0))

print("\nThe minimum elements with axis=1 ",np.amin(a,1))

print("The maximum element with axis=1 ",np.amax(a,1))

print("Array:\n",a)

print("Mean of array :",np.mean(a))

print("Mean of array along axis 0:",np.mean(a,0))

print("Mean of array along axis 1:",np.mean(a,1))

print("\nMedian of array:",np.median(a))

print("\nMedian of array along axis 0:",np.median(a,0))

print("\nMedian of array along axis 1:",np.median(a,1))

print("Average of array along axis 1:",np.average(a,1))

print("Variance of array :",np.var(a))

print("Variance of array with axis=0 :",np.var(a,0))

print("Variance of array with axis=1 :",np.var(a,1))

print("Standard Deviation of array :",np.std(a))

print("Standard Deviation of array with axis=0 :",np.std(a,0))

print("Standard Deviation of array with axis=1 :",np.std(a,1))

Output:

The original array:

[[ 2 10 20]

[80 43 31]

[22 43 10]]

The minimum element among the array: 2

The maximum element among the array: 80

The minimum elements with axis=0 [ 2 10 10]

The maximum elements with axis=0 [80 43 31]

The minimum elements with axis=1 [ 2 31 10]

The maximum element with axis=1 [20 80 43]

Array:

[[ 2 10 20]

[80 43 31]

[22 43 10]]

Mean of array : 29.0

Mean of array along axis 0: [34.66666667 32. 20.33333333]

Mean of array along axis 1: [10.66666667 51.33333333 25. ]

Median of array: 22.0

Median of array along axis 0: [22. 43. 20.]

Median of array along axis 1: [10. 43. 22.]

Average of array along axis 1: [10.66666667 51.33333333 25. ]

Variance of array : 508.6666666666667

Variance of array with axis=0 : [1094.22222222 242. 73.55555556]

Variance of array with axis=1 : [ 54.22222222 434.88888889 186. ]

Standard Deviation of array : 22.55363976538303

Standard Deviation of array with axis=0 : [33.07902995 15.55634919 8.57645355]

Standard Deviation of array with axis=1 : [ 7.36357401 20.85398976 13.6381817 ]

2.b) Write python programs to implement statistical functions like Mean, Median, Variance, and Standard Deviation using numpy on dataset.

Program:

import numpy as np

dataset =np.genfromtxt('C:/Users/pc/Desktop/dataset.csv',delimiter=',')

print("The original array:\n")

print(dataset)

print("The rows & columns on dataset:", dataset.shape)

print("\nThe minimum element among the array:",np.amin(dataset))

print("The maximum element among the array:",np.amax(dataset))

print("\nThe minimum elements with axis=0 ",np.amin(dataset,0))

print("The maximum elements with axis=0 ",np.amax(dataset,0))

print("\nThe minimum elements with axis=1 ",np.amin(dataset,1))

print("The maximum element with axis=1 ",np.amax(dataset,1))

print("Mean of array :",np.mean(dataset))

print("Mean of array along axis 0:",np.mean(dataset,0))

print("Mean of array along axis 1:",np.mean(dataset,1))

print("\nMedian of array:",np.median(dataset))

print("\nMedian of array along axis 0:",np.median(dataset,0))

print("\nMedian of array along axis 1:",np.median(dataset,1))

print("Average of array along axis 1:",np.average(dataset,1))

print("Variance of array :",np.var(dataset))

print("Variance of array with axis=0 :",np.var(dataset,0))

print("Variance of array with axis=1 :",np.var(dataset,1))

print("Standard Deviation of array :",np.std(dataset))

print("Standard Deviation of array with axis=0 :",np.std(dataset,0))

print("Standard Deviation of array with axis=1 :",np.std(dataset,1))

Output:

The original array:

[[ 65.78331 112.9925 345.89 234.23 ]

[ 71.51521 136.4873 346.89 235.23 ]

[ 69.39874 153.0269 347.89 236.23 ]

[ 68.2166 142.3354 348.89 237.23 ]

[ 67.78781 144.2971 349.89 238.23 ]

[ 68.69784 123.3024 350.89 239.23 ]

[ 69.80204 141.4947 351.89 240.23 ]

[ 70.01472 136.4623 352.89 241.23 ]

[ 67.90265 112.3723 353.89 242.23 ]

[ 66.78236 120.6672 354.89 243.23 ]]

The rows & columns on dataset: (10, 4)

The minimum element among the array: 65.78331

The maximum element among the array: 354.89

The minimum elements with axis=0 [ 65.78331 112.3723 345.89 234.23 ]

The maximum elements with axis=0 [ 71.51521 153.0269 354.89 243.23 ]

The minimum elements with axis=1 [65.78331 71.51521 69.39874 68.2166 67.78781 68.69784 69.80204 70.01472

67.90265 66.78236]

The maximum element with axis=1 [345.89 346.89 347.89 348.89 349.89 350.89 351.89 352.89 353.89 354.89]

Mean of array : 197.51348449999995

Mean of array along axis 0: [ 68.590128 132.34381 350.39 238.73 ]

Mean of array along axis 1: [189.7239525 197.5306275 201.63641 199.168 200.0512275 195.53006

200.854185 200.149255 194.0987375 196.39239 ]

Median of array: 193.62845

Median of array along axis 0: [ 68.45722 136.4748 350.39 238.73 ]

Median of array along axis 1: [173.61125 185.85865 194.62845 189.7827 191.26355 181.2662 190.86235

188.84615 177.30115 181.9486 ]

Average of array along axis 1: [189.7239525 197.5306275 201.63641 199.168 200.0512275 195.53006

200.854185 200.149255 194.0987375 196.39239 ]

Variance of array : 11533.83509367655

Variance of array with axis=0 : [ 2.51218688 177.97972078 8.25 8.25 ]

Variance of array with axis=1 : [11904.40533048 10833.90953011 10609.12872161 11060.90167818

11127.85268899 11837.410653 11265.12134705 11502.09705744

12613.63227198 12462.07959084]

Standard Deviation of array : 107.3956940183197

Standard Deviation of array with axis=0 : [ 1.58498797 13.34090405 2.87228132 2.87228132]

Standard Deviation of array with axis=1 : [109.10731108 104.0860679 103.00062486 105.17082142 105.48863772

108.79986513 106.13727595 107.24783008 112.31042815 111.63368484]

3. Write python programs to implement Filtering, Sorting, Combining (vstack), Reshaping operations using numpy.

**import** numpy **as** np

arr**=**np**.**array([[1,2,3],[5,2,3],[6,4,9],[9,11,5]])

print('Our array is:')

print(arr)

print('\n')

print("-----1.Performing Filtering operations in Numpy-------")

print("----a)Based on Boolean index list-----------")

x**=**[**False**,**True**,**True**,**False**]

print("The boolean index list:",arr[x])

print('\n')

print("-----b)Based on Conditions,filtering the array-------")

print("arr[arr>5]:",arr[arr**>**5])

print("arr[arr==2]:",arr[arr**==**2])

print("arr[(arr>2)&(arr<10)]:",arr[(arr**>**2)**&**(arr**<**10)])

print("arr[(arr>3|arr<10)]:",arr[(arr**>**3)**|**(arr**<**10)])

print('\n')

print("----c)Using where(),filtering the array-------")

b**=**np**.**where(arr**>**2)

print("The array is:",b)

print('\n')

c**=**np**.**where((arr**<**4)**|**(arr**>**3))

print("where((arr<4)|(arr>3)):",c)

print('\n')

d**=**np**.**where((arr**>**2)**&**(arr**<**5))

print("where((arr>2)&(arr<5)):",d)

print('\n')

print("----d)Using extract(),filtering the array-------")

a1**=**np**.**extract((arr**>**5),arr)

print("extract((arr>5),arr)):",a1)

print('\n')

a2**=**np**.**extract((arr**<**4)**|**(arr**>**3),arr)

print("extract((arr<4)|(arr>3),arr):",a2)

print('\n')

a3**=**np**.**extract((arr**<**4)**&**(arr**>**3),arr)

print("extract((arr<4)&(arr>3),arr):",a2)

print('\n')

print("-----2.Performing Sorting operations in Numpy-------")

b1**=**np**.**sort(arr)

print("The sorted array is:\n",b1)

b2**=**np**.**sort(arr,axis**=**0)

print("Sort along axis=0:\n",b2)

b3**=**np**.**sort(arr,axis**=**1)

print("Sort along axis=1:\n",b3)

b4**=**np**.**sort(arr,axis**=None**)

print("Sort along axis=None:\n",b4)

print('\n')

print("-----Applying argmax() function--------")

print(" The index value of maximum element:",np**.**argmax(arr))

print("Indices of maximum along axis=0:",np**.**argmax(arr,axis**=**0))

print("Indices of maximum along axis=1:",np**.**argmax(arr,axis**=**1))

print('\n')

print("-----Applying argmin() function--------")

print(" The index value of minimum element:",np**.**argmin(arr))

print("Indices of minimum along axis=0:",np**.**argmin(arr,axis**=**0))

print("Indices of minimum along axis=1:",np**.**argmin(arr,axis**=**1))

print('\n')

print("-----Applying argsort() function--------")

print("Indices of sorted elements:\n",np**.**argsort(arr))

print("Indices of sorted elements along axis=0:\n",np**.**argsort(arr,axis**=**0))

print("Indices of sorted elements along axis=1:\n",np**.**argsort(arr,axis**=**1))

print('\n')

print("-----3.Applying vstack,hstack function--------")

arr1**=**np**.**array([[0,1],[2,3]])

print("First array:\n",arr1)

arr2**=**np**.**array([[4,5],[6,7]])

print("Second array:\n",arr2)

print("The Horizontal Stack:\n",np**.**hstack((arr1,arr2)))

print("The Vertical Stack:\n",np**.**vstack((arr1,arr2)))

print('\n')

print("----4.Performing Reshaping operation on the array------")

print("The array is :\n",arr)

res **=** np**.**reshape(arr, (2, 6))

print("Reshaping the original array with 2 rows, 6 columns:\n",res)

res1 **=** np**.**reshape(arr, (2,3,2))

print("Reshaping the original array with 2 matrix,3 rows, 2 columns:\n ",res1)

Output:

Our array is:

[[ 1 2 3]

[ 5 2 3]

[ 6 4 9]

[ 9 11 5]]

-----1.Performing Filtering operations in Numpy-------

----Based on Boolean index list-----------

The boolean index list: [[5 2 3]

[6 4 9]]

-----Based on Conditions,filtering the array-------

arr[arr>5]: [ 6 9 9 11]

arr[arr==2]: [2 2]

arr[(arr>2)&(arr<10)]: [3 5 3 6 4 9 9 5]

arr[(arr>3|arr<10)]: [ 1 2 3 5 2 3 6 4 9 9 11 5]

----Using where(),filtering the array-------

The array is: (array([0, 1, 1, 2, 2, 2, 3, 3, 3], dtype=int64), array([2, 0, 2, 0, 1, 2, 0, 1, 2], dtype=int64))

where((arr<4)|(arr>3)): (array([0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3], dtype=int64), array([0, 1, 2, 0, 1, 2, 0, 1, 2, 0, 1, 2], dtype=int64))

where((arr>2)&(arr<5)): (array([0, 1, 2], dtype=int64), array([2, 2, 1], dtype=int64))

----Using extract(),filtering the array-------

extract((arr>5),arr)): [ 6 9 9 11]

extract((arr<4)|(arr>3),arr): [ 1 2 3 5 2 3 6 4 9 9 11 5]

extract((arr<4)&(arr>3),arr): [ 1 2 3 5 2 3 6 4 9 9 11 5]

-----2.Performing Sorting operations in Numpy-------

The sorted array is:

[[ 1 2 3]

[ 2 3 5]

[ 4 6 9]

[ 5 9 11]]

Sort along axis=0:

[[ 1 2 3]

[ 5 2 3]

[ 6 4 5]

[ 9 11 9]]

Sort along axis=1:

[[ 1 2 3]

[ 2 3 5]

[ 4 6 9]

[ 5 9 11]]

Sort along axis=None:

[ 1 2 2 3 3 4 5 5 6 9 9 11]

-----Applying argmax() function--------

The index value of maximum element: 10

Indices of maximum along axis=0: [3 3 2]

Indices of maximum along axis=1: [2 0 2 1]

-----Applying argmin() function--------

The index value of minimum element: 0

Indices of minimum along axis=0: [0 0 0]

Indices of minimum along axis=1: [0 1 1 2]

-----Applying argsort() function--------

Indices of sorted elements:

[[0 1 2]

[1 2 0]

[1 0 2]

[2 0 1]]

Indices of sorted elements along axis=0:

[[0 0 0]

[1 1 1]

[2 2 3]

[3 3 2]]

Indices of sorted elements along axis=1:

[[0 1 2]

[1 2 0]

[1 0 2]

[2 0 1]]

-----3.Applying vstack,hstack function--------

First array:

[[0 1]

[2 3]]

Second array:

[[4 5]

[6 7]]

The Horizontal Stack:

[[0 1 4 5]

[2 3 6 7]]

The Vertical Stack:

[[0 1]

[2 3]

[4 5]

[6 7]]

----4.Performing Reshaping operation on the array------

The array is :

[[ 1 2 3]

[ 5 2 3]

[ 6 4 9]

[ 9 11 5]]

Reshaping the original array with 2 rows, 6 columns:

[[ 1 2 3 5 2 3]

[ 6 4 9 9 11 5]]

Reshaping the original array with 2 matrix,3 rows, 2 columns:

[[[ 1 2]

[ 3 5]

[ 2 3]]

[[ 6 4]

[ 9 9]

[11 5]]]

4. Write python programs to implement Indexing, Slicing, Iterating using Pandas.