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

import numpy as np

arr = np.array(['Vaibhav',23,45,67,'hello'])

lis=list(arr)

print(lis) #Converting given array to a list

newarr=np.array(lis) #converting list to array again

print(newarr)

output:

['Vaibhav', '23', '45', '67', 'hello']

['Vaibhav' '23' '45' '67' 'hello']

# Create a 5X2 integer array from a range between 100 to 200 such that the difference between each element is 10.

arr = np.arange(100,200,10).reshape(5,2)

arr

output:

array([[100, 110],

[120, 130],

[140, 150],

[160, 170],

[180, 190]])

#Add the two 2D NumPy arrays and modify the  resulting array by calculating the square root of each element

arr1 = np.array([[2,3],[4,5],[6,7]])

arr2 = np.array([[12,34],[56,78],[32,54]])

res= arr1+arr2

print(res)

res=np.sqrt(res)

res

output:

[[14 37]

[60 83]

[38 61]]

array([[3.74165739, 6.08276253],

[7.74596669, 9.11043358],

[6.164414 , 7.81024968]])

# Create an 8X3 integer array from a range between 10 to 34 such that the difference between each element is 1 and then Split the array into four equal-sized sub-arrays.

arr = np.arange(10,34,1).reshape(8,3)

arr=arr.reshape(4,2,3) #Dividing the array into 4 different equal sized arrays

arr

output:

array([[[10, 11, 12],

[13, 14, 15]],

[[16, 17, 18],

[19, 20, 21]],

[[22, 23, 24],

[25, 26, 27]],

[[28, 29, 30],

[31, 32, 33]]])

#Consider the following array:

# sampleArray = numpy.array([[34,43,73],[82,22,12],[53,94,66]])

#     i. Sort above array by second row

#     ii. Sort above array by second column

#     iii. Print max from axis 0 and min from axis 1

#      iv. Delete col 2 and insert new Column numpy.array([[10,10,10]]) in its place

sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66]])

print("Initial array:\n",sampleArray)

ind = np.argsort(sampleArray[1,]) #Sorting by 2nd row

sampleArray=sampleArray[:,ind]

print("Sorting by 2nd row:\n",sampleArray)

ind = np.argsort(sampleArray[0:,1]) #Sorting 2nd column

sampleArray = sampleArray[ind,]

print("Soring by 3rd row:\n",sampleArray)

print("Max from axis=0:",np.max(sampleArray,axis=0)) #Max from axis 0

print("Min from axis=1:",np.min(sampleArray,axis=1)) #Min from axis 1

sampleArray=np.delete(sampleArray,1,1) #Deleting 2nd column

print("Deleting 2nd column:\n",sampleArray)

sampleArray=np.insert(sampleArray,1,np.array([[10,10,10]]),1) #Adding an new column

print(sampleArray)

output:

Initial array:

[[34 43 73]

[82 22 12]

[53 94 66]]

Sorting by 2nd row:

[[73 43 34]

[12 22 82]

[66 94 53]]

Soring by 3rd row:

[[12 22 82]

[73 43 34]

[66 94 53]]

Max from axis=0: [73 94 82]

Min from axis=1: [12 34 53]

Deleting 2nd column:

[[12 82]

[73 34]

[66 53]]

[[12 10 82]

[73 10 34]

[66 10 53]]

# Remove all the elemnets from an array that exist in another array?

print(sampleArray)

arr = np.array([34,73,82,22,12,53,94,66,77,22]).reshape(5,2)

print(arr)

sampleArray = np.setdiff1d(sampleArray,arr)

print("Elements in sampleArray that are not in arr:",sampleArray)

output:

[[12 10 82]

[73 10 34]

[66 10 53]]

[[34 73]

[82 22]

[12 53]

[94 66]

[77 22]]

Elements in sampleArray that are not in arr: [10]

#Swap two columns in a 2d NumPy array.

sampleArray = np.array([[34,43,73],[82,22,12],[53,94,66],[77,23,45]])

print("Before swapping:"sampleArray)

sampleArray[:,[2,0]] = sampleArray[:,[0,2]] #swapping first and last columns

print(sampleArray)

output:

[[34 43 73]

[82 22 12]

[53 94 66]

[77 23 45]]

[[73 43 34]

[12 22 82]

[66 94 53]

[45 23 77]]

#Swap two rows in a 2d NumPy array

sA = np.array([[34,43,73],[82,22,12],[53,94,66],[77,23,45]])

print("Before swapping\n",sA)

sA[[2,1]] = sA[[1,2]] #swapping 2nd and 3rd row

print("After swapping\n",sA)

output:

Before swapping

[[34 43 73]

[82 22 12]

[53 94 66]

[77 23 45]]

After swapping

[[34 43 73]

[53 94 66]

[82 22 12]

[77 23 45]]

#Reverse the order of rows of a 2D array.

sA = np.array([[34,43,73],[82,22,12],[53,94,66],[77,23,45]])

print("Before reversing the rows:\n",sA)

nrow,ncol = sA.shape

sA = sA[nrow::-1]

print("After reversing rows:\n",sA)

output:

Before reversing the rows:

[[34 43 73]

[82 22 12]

[53 94 66]

[77 23 45]]

After reversing rows:

[[77 23 45]

[53 94 66]

[82 22 12]

[34 43 73]]

#Reverse the order of columns of a 2D array

sA = np.array([[34,43,73],[82,22,12],[53,94,66],[77,23,45]])

print("Before reversing the columns:\n",sA)

nrow,ncol = sA.shape

sA = sA[:,ncol::-1]

print("After reversing columns:\n",sA)

output:

Before reversing the columns:

[[34 43 73]

[82 22 12]

[53 94 66]

[77 23 45]]

After reversing columns:

[[73 43 34]

[12 22 82]

[66 94 53]

[45 23 77]]

#Retrive indices where elements of two arrays match

sA = np.array([34,43,73,82,22,12,53,94,66,77,23,45])

sA2 = np.array([12,34,56,78,32,54])

for i in sA:

  if i in sA2:

    print(i)

    print(np.where(sA==i)[0])

    print(np.where(sA2==i)[0])

output:

34

[0]

[1]

12

[5]

[0]

#Get all items between 5 and 10 from an array

eg = np.array([5,6,17,8,2,32,4,53,6,7,86,9])

print(eg[np.logical\_and(eg>=5, eg<=10)])

output:

[5 6 8 6 7 9]

#For a 1D array with numeric values, find minimum, maximum, mean, median, standard deviation, 5th and 95th percentile, unique values, count of unique values, and the most frequent value.

eg = np.array([5,6,17,8,2,32,4,53,6,7,86,9])

print("Minimum:",np.min(eg))

print("Maximum:",np.max(eg))

print("Median:",np.median(eg))

print("Mean:",np.mean(eg))

print("Standard deviation",np.std(eg))

print("5th and 95th percentile respectively:",np.percentile(eg,5),"  ",np.percentile(eg,95))

print("Unique values:",np.unique(eg))

print("Count of unique values:",np.unique(eg).shape)

no\_of\_occ = np.bincount(eg)

print("Most frequent value:",no\_of\_occ.argmax())

Minimum: 2

Maximum: 86

Median: 7.5

Mean: 19.583333333333332

Standard deviation 24.540640895371

5th and 95th percentile respectively: 3.1 67.84999999999998

Unique values: [ 2 4 5 6 7 8 9 17 32 53 86]

Count of unique values: (11,)

Most frequent value: 6

#Write a NumPy program to create a 10x10 matrix, in which all the elements on the borders should be equal to 1, and rest others should be 0.

arr = np.ones([10,10])

arr[1:-1,1:-1]=0

print(arr)

output:

[[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]]

#Write a NumPy program to create a 5x5 zero matrix with elements on the main diagonal equal to 1, 2, 3, 4, 5

arr = np.zeros([5,5])

for i in range(5):

  arr[i,i]=i+1

print(arr)

arr = np.diag([1,2,3,4,5])

print(arr)

output:

[[1. 0. 0. 0. 0.]

[0. 2. 0. 0. 0.]

[0. 0. 3. 0. 0.]

[0. 0. 0. 4. 0.]

[0. 0. 0. 0. 5.]]

[[1 0 0 0 0]

[0 2 0 0 0]

[0 0 3 0 0]

[0 0 0 4 0]

[0 0 0 0 5]]

#Count the number of elements in a numpy array which are greater than 10. Further, multiply all such elements with value 10.

eg = np.array([5,6,17,8,2,32,4,53,6,7,86,9])

print("No of elements greater than 10:",len(eg[eg>10]))

for i in range(len(eg)):

  if eg[i]>10:

    eg[i]\*=10

print(eg)

output:

No of elements greater than 10: 4

[ 5 6 170 8 2 320 4 530 6 7 860 9]