1. ****What are the various features of NumPy?****

As a powerful open-source package used for array-processing, NumPy has various useful features. They are:

* Contains a N-dimensional array object
* It is interolerable; compatible with many hardware and computing platforms
* Works extremely well with array libraries; sparse, distributed or GPU
* Ability to perform complicated (broadcasting) functions
* Tools that enable integration with C or C++ and Fortran code
* Ability to perform high-level mathematical functions like statistics, Fourier transform, sorting, searching, linear algebra, etc
* It can also behave as a multi-dimensional container for generic data
* Supports scientific and financial calculations

1. ****What are the steps to use shape for a 1D array, 2D array and 3D/ND array respectively?****

1D Array:

num=[1,2,3] if not added

print(‘\nshpae of 1d ‘,num.shape)

2D Array:

num2=[[1,2,3],[4,5,6]] if not added

print(‘\nshpae of 2d ‘,num2.shape)

3D or ND Array:

num3=[[[1,2,3],[4,5,6],[7,8,9]]] if not added

print(‘\nshpae of 3d ‘,num3.shape)

1. ****What is the procedure to count the number of times a given value appears in an array of integers?****

Note : It should be noted that the bincount() function accepts positive integers or boolean expressions as its argument. Negative integers cannot be used.

arr = NumPy.array([0, 5, 4, 0, 4, 4, 3, 0, 0, 5, 2, 1, 1, 9])

NumPy.bincount(arr)

1. ****What is the procedure to find the indices of an array on NumPy where some condition is true?****

>>> import numpy as np

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

>>> a > 3

array([[False, False, False],

       [ True,  True,  True],

       [ True,  True,  True]], dtype=bool)

>>> np.nonzero(a > 3)

(array([1, 1, 1, 2, 2, 2]), array([0, 1, 2, 0, 1, 2]))

You can also call the nonzero() method of the boolean array.

>>> (a > 3).nonzero()

(array([1, 1, 1, 2, 2, 2]), array([0, 1, 2, 0, 1, 2]))

1. ****Shown below is the input NumPy array. Delete column two and replace it with the new column given below.****

import NumPy

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

newColumn = NumPy.array([[10,10,10]])

Expected Output:

Printing Original array

[[34 43 73]

 [82 22 12]

 [53 94 66]]

Array after deleting column 2 on axis 1

[[34 73]

 [82 12]

 [53 66]]

Array after inserting column 2 on axis 1

[[34 10 73]

 [82 10 12]

 [53 10 66]]

**Solution:**

import NumPy

print(“Printing Original array”)

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

print (sampleArray)

print(“Array after deleting column 2 on axis 1”)

sampleArray = NumPy.delete(sampleArray , 1, axis = 1)

print (sampleArray)

arr = NumPy.array([[10,10,10]])

print(“Array after inserting column 2 on axis 1”)

sampleArray = NumPy.insert(sampleArray , 1, arr, axis = 1)

print (sampleArray)

#### ****Use numpy to generate array of 25 random numbers sampled from a standard normal**distribution**

print(‘\n random number 25\n ‘,np.random.rand(25))

1. **Dot product: product of two arrays**  
   f = np.array([1,2])  
   g = np.array([4,5])  
   ### 1\*4+2\*5  
   np.dot(f, g)
2. **Extract all odd numbers from **arr****

# Input arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

**# Solution** arr[arr % 2 == 1]****Output:**** array([1, 3, 5, 7, 9])

#### ****Replace items that satisfy a condition without affecting the original array?****

arr = np.arange(10)

out = np.where(arr % 2 == 1, -1, arr)

print(arr)

out

Output: [0 1 2 3 4 5 6 7 8 9]

array([ 0, -1, 2, -1, 4, -1, 6, -1, 8, -1])

#### ****How to stack two arrays vertically?****

a = np.arange(10).reshape(2,-1)

b = np.repeat(1, 10).reshape(2,-1)

# Answers# Method 1:

np.concatenate([a, b], axis=0)

# Method 2:

np.vstack([a, b])

# Method 3:

np.r\_[a, b]

****Output:****

#> array([[0, 1, 2, 3, 4],

#> [5, 6, 7, 8, 9],

#> [1, 1, 1, 1, 1],

#> [1, 1, 1, 1, 1]])

#### ****How to get the positions where elements of two arrays match?****

a = np.array([1,2,3,2,3,4,3,4,5,6])

b = np.array([7,2,10,2,7,4,9,4,9,8])

np.where(a == b)

****Output:****#> (array([1, 3, 5, 7]),)

#### ****How to reverse the rows of a 2D array?****

Input

arr = np.arange(9).reshape(3,3)

# Solution

arr[::-1]

****Output:****

array([[6, 7, 8],

[3, 4, 5],

[0, 1, 2]])

1. **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.**

Expected Output:

Creating 8X3 array using numpy.arange[[10 11 12]

[13 14 15]

[16 17 18]

[19 20 21]

[22 23 24]

[25 26 27]

[28 29 30]

[31 32 33]]

Dividing 8X3 array into 4 sub array

[array([[10, 11, 12],[13, 14, 15]]),

array([[16, 17, 18],[19, 20, 21]]),

array([[22, 23, 24],[25, 26, 27]]),

array([[28, 29, 30],[31, 32, 33]])]

**Solution:**

import numpy

print("Creating 8X3 array using numpy.arange")

sampleArray = numpy.arange(10,34,1)

sampleArray = sampleArray.reshape(8,3)print(sampleArray)

print("\nDividing 8X3 array into 4 sub array\n")

subArrays = numpy.split(sampleArray,4)print(subArrays)

1. ****Create a two 2-D array. Plot it using matplotlib?****

data = [[1, 2, 3],

[4, 5, 6],

[7, 8, 9]]

plt.imshow(data)

plt.show()

1. **Days(x-axis) represents 8 days and Speed represents a car’s speed. Plot a Basic line plot between days and car speed, put x axis label as days and y axis label as car speed and put title Car Speed Measurement.**

**Days=[1,2,3,4,5,6,7,8]**

**Speed=[60,62,61,58,56,57,46,63]**​

​​​​

import matplotlib.pyplot as plt # Import Required Package

Days = [1,2,3,4,5,6,7,8]

Speed = [60,62,61,58,56,57,46,63]

plt.plot(Days, Speed, color = 'blue') # For Create a Line Plot

plt.xlabel('Days')

plt.ylabel('Car Speed')

plt.title('Car Speed Measurement')

plt.show()

1. **Plot Simple bar chart showing popularity of Programming Languages.**

**Plot Multiple Bars showing Popularity and Security of major Programming Languages. Also Create Horizontal bar chart using barh function.**

**Languages =['Python', 'SQL', 'Java', 'C++', 'JavaScript']**

**Popularity = [56, 39, 34, 34, 29]**

**Security = [44 ,36 ,55, 50, 42]**

Languages =['Python', 'SQL', 'Java', 'C++', 'JavaScript']

Popularity = [56, 39, 34, 34, 29]

Security = [44 ,36 ,55, 50, 42]

plt.subplot(2,1,1)

plt.bar(Languages, Popularity, width=0.5, color = 'orange', align='center')

plt.xlabel('Languages')

plt.ylabel('Popularity')

plt.legend(['Popularity'])

plt.show()

plt.subplot(2,1,2)

plt.bar(Languages, Security, width=0.5, color = 'red', align='center')

plt.xlabel('Languages')

plt.ylabel('Security')

plt.legend(['Security'])

plt.show()

Horizontal Bar

Languages =['Python', 'SQL', 'Java', 'C++', 'JavaScript']

Popularity = [56, 39, 34, 34, 29]

Security = [44 ,36 ,55, 50, 42]

plt.subplot(2,1,1)

plt.barh(Languages, Popularity, color = 'g', align='center')

plt.xlabel('Languages')

plt.ylabel('Popularity')

plt.legend(['Popularity'])

plt.show()

plt.subplot(2,1,2)

plt.barh(Languages, Security, color = 'red', align='center')

plt.xlabel('Languages')

plt.ylabel('Security')

plt.legend(['Security'])

plt.show()

1. **Plot Histogram, We have a sample data of Students marks of various Students, we will try to plot number of Students by marks range and try to figure out how many Students are average, below-average and Excellent.**

**Marks = [ 61,86,42,46,73,95,65,78,53,92,55,69,70,49,72,86,64]**

**Histogram showing Below Average, Average and Execellent distribution**

**40-60: Below Average**

**60-80: Average**

**80-100: Excellent**

Marks = np.array([61,86,42,46,73,95,65,78,53,92,55,69,70,49,72,86,64])

below\_avg = Marks[np.logical\_and(Marks >= 40, Marks < 60)]

avg\_marks = Marks[np.logical\_and(Marks >= 60, Marks < 80)]

exe\_marks = Marks[np.logical\_and(Marks >= 80, Marks <= 100)]

plt.hist(Marks)

plt.show()

print('Below Average Students Are : ',below\_avg.size)

print('Average Students Are : ',avg\_marks.size)

print('Execelent Students Are : ',exe\_marks.size)

1. **Write a Python program to plot two or more lines with legends, different widths and colors.**

import matplotlib.pyplot as plt # line 1 points

x1 = [10,20,30]

y1 = [20,40,10] # line 2 points

x2 = [10,20,30]

y2 = [40,10,30]

# Set the x axis label of the current axis.

plt.xlabel('x - axis')

# Set the y axis label of the current axis.

plt.ylabel('y - axis')

# Set a title

plt.title('Two or more lines with different widths and colors with suitable legends ')

# Display the figure.

plt.plot(x1,y1, color='blue', linewidth = 3, label = 'line1-width-3')

plt.plot(x2,y2, color='red', linewidth = 5, label = 'line2-width-5')

# show a legend on the plot

plt.legend()

plt.show()