**Python Libraries**

**(Numpy)**

**Practice Questions:-**

Question no :-

**a.** Write the code using np.arange() to get all even numbers between 21 and 70, (70 inclusive)

**b.** Write the code using np.arange() to get all odd numbers between 20 and 71. (71 inclusive)

Which options correctly answer questions a and b?

Choose the correct answer from below**, please note that this question may have multiple correct answers**

For question 'a', the answer is np.arange(22,71,2)

For question 'a', the answer is np.arange(21,70,2)

For question 'b', the answer is np.arange(20, 72, 2)

For question 'b', the answer is np.arange(21, 72, 2)

Question no :-

Raghu has created a numpy array **arr** using the following code:

**import numpy as np**

**arr = np.array([10, 20, 30, 40, 50])**

He wants to change the data type from integer to **float**.Which of the following is the correct approach to do so?

Choose the correct answer from below:

**float(arr)**

**arr.to\_float()**

**arr.astype('float64')**

**type(arr, dtype='float64')**

Question no :-

What woud be the output of the following code :

**import numpy as np**

**a = np.arange(10)**

**print(a[2:5])**

Choose the correct answer from below:

**[2, 4, 6]**

**[0, 1, 2]**

**[5, 6, 7]**

**[2, 3, 4]**

Question no :-

Which option is correct about the output of the following code snippet?

**import numpy as np  
x = np.array([-5, 9 , 20 , 25, -3, 5, 16, 10,-8])  
x[(x >= -5) & (x <= 15)] \*= -1  
print(x)**

Choose the correct answer from below:

**Given code will change the sign of all the numbers in x.**

**Given code will change the sign of all the numbers in range [-5, 15] in x.**

**Given code will change the sign of all the numbers greater than -5 in x.**

**Given code will change the sign of all the numbers smaller than 15 in x.**

Question no :-

What will be the output of the following code?

**a = np.array([100, 200, 300, 400])  
b = np.array([300, 200, 100, 400])  
print(a == b)**

Choose the correct answer from below:

**True**

**False**

**[True, False, True, False]**

**[False, True, False, True]**

Question no :-

Given the start, end, and the stepsize return a numpy array sequence in given range with the specified stepsize.

**Input Format:**

One line of input will have 3 space-separated integers consisting of start, end of sequence and step.

**Output Format:**

A numpy array with elements rounded off to 2 decimal places.

**Sample Input:**

5 7 0.5

**Sample Output:**

[5 5.5 6 6.5]

**Note: To round off the numpy array, use np.round()**

P.s: Recall that we can have float step size in numpy array

import numpy as np

def create\_seq(start, end, step):

    arr = np.round(np.arange(start, end, step),2)

    return arr

Question no :-

Given a 1D array, return the first and last elements from the array.

**Input Format:**

A 1D numpy array

**Output Format:**

A tuple (first\_element, last\_element)

**Sample Input:**

[0, 1, 2, 3, 4, 5]

**Sample Output:**

(0, 5)

import numpy as np

def get\_elements(arr):

    '''

    INPUT: arr -> 1D numpy array

    OUTPUT elements -> tuple of first and last element.

    '''

    first\_element = arr[0]

    last\_element = arr[-1]

    return (first\_element, last\_element)

Question no :-

What would be the output for the following code :

**import numpy as np**

**arr = np.array([1, 2, 3, 4])**

**print(arr[2] + arr[-2])**

Choose the correct answer from below:

**3**

**5**

**4**

**6**

Question no :-

Create a sequence of a given **length** from a given **start** point, where the difference between 2 consecutive elements of the expected sequence is also given as **step**.

**Input Format:**

One line of input will have 3 space-separated integers consisting of start, length of sequence and step between two continuous elements of the sequence.

**Output Format:**

A numpy array of integers

**Sample Input:**

5 10 3

**Sample Output:**

[5 8 11 14 17 20 23 26 29 32]

**Sample explanation:**

Start of the sequence = 5, length of the sequence = 10 and step = 3.

First point would be the start point only, second point = start + 1\*step = 5 + 1\*3 =8, ....... 10th point = start + (10-1)\*step = 5 + 9\*3 = 32

**Note**: You can use either .arange() or .linspace() to solve this question.

import numpy as np

def seq(start, length, step):

    ''' start, length and step are in form of integers all representing the attributes as their names suggest

        output -> A numpy array is expected to be returned'''

    # YOUR CODE GOES HERE

    sequence = np.arange(start, start+length\*step, step)

    return sequence

Question no :-

Given an array, return the shape and dimension of the array.

**Input Format:**

A numpy array

**Output Format:**

A Tuple (shape, dim)

**Sample Input:**

[0 1 2 3 4 5]

**Sample Output:**

((6,), 1)

import numpy as np

def get\_array\_properties(arr):

    ## STEP 1. Get the shape of array

    shape = np.shape(arr)

    ## STEP 2. Get the dimension of array

    dim = np.ndim(arr)

    return shape,dim

END OF NUMPY 1

Question no :-

**import numpy as np  
arr = 2 \* np.arange(0,2,0.5)  
if arr <= 0.6:  
 print("condition satisfies")  
else:  
 print("condition doesn't satisfy")**

In the above code, '**condition**' implies the situation that **arr** has **at least one value smaller than or equal to 0.6**. Which option is true with respect to the code and condition?

Choose the correct answer from below:

**The Code will give the required output.**

**The Code will throw ValueError, and np.any() should be used to get the required output.**

**The Code will throw ValueError and np.all() should be used to get the required output.**

**None of the given option is correct**

Question no :-

Mark the options which are true about the outputs for code snippets **a** and **b**.

Code Snippet a:

**import numpy as np  
x = np.array([[200,200,200],[300,300,300],[400,400,400]])  
v = np.array([200,300,400])  
print((x / v[:,None])[1][1])**

Code Snippet b:

**import numpy as np  
p = np.array([[0], [10], [20]])  
q = np.array([10, 11, 12])   
print((p + q)[1][1])**

Choose the correct answer from below**, please note that this question may have multiple correct answers**

For 'a', the answer is 1.0

For 'a', the answer is 2.0

For 'b', the answer is 21

The code in 'b' will throw Value Error.

Question no :-

Your batchmate writes the following code :

import numpy as np

a = np.array([[16, 5], [81, 6], [33, 1]])

x=np.transpose(a).reshape(2,3)

print(x.flatten())

Which of the following is the correct output for the above code?  
  
**Note:**

* flatten() function is used to transform a multi-dimensional array into a one-dimensional array.
* In other words, it "flattens" a multi-dimensional array structure into a simple linear sequence.

Choose the correct answer from below:

**[16 81 33 5 6 1]**

**[16 33 81 5 1 6]**

**[16 5 81 6 33 1]**

**[[16 5 81 6 33 1]]**

Question no :-

Given an array of marks, return the array only containing elements with marks > 40

**Input Format:**

A 1D numpy array

**Output Format:**

A 1D numpy array

**Sample Input:**

[85, 18, 2, 57, 65, 44]

**Sample Output:**

[85, 57, 65, 44]

import numpy as np

def filter\_marks(marks):

    ### Step 1 Get the mask for marks > 40

    mask = marks > 40

    ### STEP 2 use the mask to filter the array

    filtered\_array = marks[mask]

    return filtered\_array

Question no :-

Given the following code, what will be the output?

import numpy as np

a = np.array([[34, 28,55], [8, 56, 3], [77, 87, 19]])

print(a.transpose()[-2,-2])

Choose the correct answer from below:

**55**

**28**

**56**

**3**

Question no :-

Given a numpy array and target value k,

Return True if all elements of array satisfy all below given conditions

* Multiple of 2
* Greater than k

**Input Format:**

The input has two lines

First line is the array

Second line is the integer value k

**Output Format:**

Boolean value i.e. True or False

**Sample Input:**

[0, 1, 2, 3, 4, 5, 6, 7, 8]

3

**Sample Output:**

False

**Sample Input:**

[8, 12, 16, 20]

4

**Sample Output:**

True

Note: Recall logical functions in numpy

import numpy as np

def check\_conditions(arr, k):

    result = None

    ## STEP 1 : Create mask for the given condition

    mask = ((arr > k) & (arr%2==0)).any()

    ## STEP 2: Use logical function on mask

    return mask

Question no :-

Given a 2d array, write a program to return a subarray such that the subarray consists of the elements from:

**1.** the second to the fourth row of the original array,

**2.** the elements in these rows should be from the last three columns of the corresponding rows of the original array,

**3.** the rows should be in reversed order.

**Sample Input:**

[[ 0, 1, 2, 3],

[ 4, 5, 6, 7],

[ 8, 9, 10, 11],

[12, 13, 14, 15],

[16, 17, 18, 19]]

**Sample Output:**

array([[13, 14, 15],

[ 9, 10, 11],

[ 5, 6, 7]])

**Input Format:**

A 2D list

**Output Format:**

A 2D numpy array

**Note:**

This question can be solved using negative indexing and slicing

import numpy as np

def extract\_subarray(arr):

    '''

    INPUT: arr -> 2D array

    OUPUT: result -> 2D array

    '''

    ### STEP1: Get the rows (2nd  to 4th row)

    row\_array = arr[1:4]

    #### STEP 2: Get the last 3 cols from the row array

    cols\_array = row\_array[:,-3::]

    #### STEP3: Reverse the rows in cols array

    result = cols\_array[::-1,:]

    return result

Question no :-

What would the following code print?

**import numpy as np**

**a = np.array([[6, 28], [8, 56], [7, 19]])**

**x = np.transpose(a).reshape(1,6)**

**print(x)**

Choose the correct answer from below:

**[ 6 8 7 28 56 19]**

**[[ 6 8 7 28 56 19]]**

**[ 6 28 8 56 7 19]**

**[[ 6 28 8 56 7 19]]**

Question no :-

**Problem Statement:**

Given a 2D numpy array, return array with its values in the column reversed.

**Input Format:**

A 2D numpy array

**Output Format:**

A 2D numpy array

**Sample Input:**

[[0, 1, 2],

[3, 4, 5],

[6, 7, 8]]

**Sample Output:**

[[2, 1, 0],

[5, 4, 3],

[8, 7, 6]]

P.S : Think about how we reverse list in python.

import numpy as np

def reverse\_column(arr):

    '''

    INPUT: arr -> 2D array

    OUTPUT rev\_arr -> 2D array

    '''

    rev\_arr = arr[::,::-1]

    return rev\_arr

Question no :-

Given an array in form of a matrix of size (n, n), rotate the matrix clockwise by 90º.

**Input Format:**

A 2d numpy array

**Output Format:**

A 2d numpy array

**Sample Input:**

[[1 2 3]   
 [4 5 6]  
 [7 8 9]]

**Sample Output:**

[[7 4 1]  
 [8 5 2]  
 [9 6 3]]

Note: Try Transpose / reversing.

import numpy as np

def rotate\_img(mat):

    '''mat -> A 2d numpy array

       output -> A 2d numpy array is expected to be returned'''

    # YOUR CODE GOES HERE

    result = mat.T

    return result[::,::-1]

END OF NUMPY 2

Question no :-

What will be the **outcome** of the following code snippet ?

x = np.ones((5,5))

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

Choose the correct answer from below:

**All the elements except those at the border are equated to zero.**

**All the elements at the border are equated to zero.**

**All the elements in the first and last columns are equated to zero.**

**All the elements in the first and last rows are equated to zero.**

Question no :-

Given the NumPy array **arr**, which of the following line of code will return the expected output?

**import numpy as np  
arr= np.array([[2,3,4,5],[1,7,3,5],[2,8,6,9],[11,23,12,19]])**

**Expected output:**

**array ([[4,6,8,10],  
 [2,14,6,10],  
 [4,16,12,18],  
 [22,46,24,38]])  
a.**

**arr1 = np.array([[2,2,2,2]])  
def func(x, y):  
 return x \* y  
vec = np.vectorize(func)  
vec(arr,arr1)  
b.**

**arr1 = np.array([[2],[2],[2],[2]])  
def func(x, y):  
 return x \* y  
vec = np.vectorize(func)  
vec(arr, arr1)  
c.**

**arr1 = 2  
def func(x, y):  
 return x \* y  
vec = np.vectorize(func)  
vec(arr, arr1)**

Choose the correct answer from below**, please note that this question may have multiple correct answers**

b

c

a

None of the options are correct

Question no :-

Given a list of birds and their corresponding age, return the name of birds sorted according to age (ascending)

**Input Format:**

Two 1D array list i.e. bird array and age array

**Output Format:**

A 1D array

**Sample Input:**

birds = ['spoonbills', 'plovers', 'plovers', 'plovers', 'plovers', 'Cranes', 'plovers', 'plovers', 'Cranes', 'spoonbills']

age = [5.5, 6.0, 3.5, 1.5, 3.0, 4.0, 3.5, 2.0, 5.5, 6.0]

**Sample Output:**

['plovers', 'plovers', 'plovers', 'plovers', 'plovers', 'Cranes', 'spoonbills', 'Cranes', 'plovers', 'spoonbills']

Side Note: Recall the functionality of .sort() and .argsort()

import numpy as np

def sort\_birds(birds, age):

    '''

    INPUT: birds, age -> 1D numpy array

    OUTPUT: result -> sorted bird 1D array

    '''

    ## STEP 1 : Get the sorted index of age.

    sorted\_age\_index = np.argsort(age)

    ## STEP 2: Use the index from previous step to get sorted birds

    result = birds[sorted\_age\_index]

    return result

Question no :-

**a.**What is the last element of the output?

**import numpy as np  
print(np.sort(np.array(['Ram','Astha','Raghavendra'])))**

**b.**What is the output of the code snippet given below?

**arr1 = np.array(['Ram','Astha','Brahat'])  
arr2 = np.array(['Shyam','Kalyan','Naveen'])  
arr1 > arr2**

Choose the options which are answers to questions **a** and **b**.

Choose the correct answer from below**, please note that this question may have multiple correct answers**

For block 'a', answer is 'Ram'.

For block 'a', answer is 'Raghavendra'.

For block 'b', answer is array([False, False, False]).

For block 'b', answer is False.

Question no :-

Given two 2D arrays, row and column ranges,

Perform the following operations:

* Find the matrix multiplication of the given two matrices and
* Extract the elements from the output of above step using the given ranges
* If matrix multiplication is not possible, return -1

Note: The end points (upper range) of both rows and columns are excluded.

**Input Format:**

There will be four lines of input as follows:

First line will have mat1. Second line will have mat2.

Third line will have two space-separated integers representing start and end point of rows.

Fourth line will have two space-separated integers representing start and end point of columns.

**Output Format:**

A Numpy array

**Sample Input:**

[[6 6 4 7 9]

[0 2 2 9 3]

[6 0 2 5 2]

[2 4 3 5 5]]

[[8 6 6 8 3]

[2 7 0 3 1]

[3 2 1 5 2]

[7 0 7 6 8]

[1 5 6 4 5]]

1 3

2 4

**Sample Output:**

[[83 82]

[85 96]]

import numpy as np

def specific\_elements(mat1,mat2,r1,r2,c1,c2):

    # STEP1 CHECK whether matrix multiplication is possible

    if mat1.shape[1] != mat2.shape[0]:

        return -1

    ## STEP 2 Perform matrix multiplication

    matmul\_array = np.dot(mat1,mat2)

    ## STEP 3 slice the array based on range value

    result = matmul\_array[r1:r2,c1:c2]

    return result

Question no :-

What is the below code printing?

import numpy as np

a = np.arange(10,22).reshape((3, 4))

print(a)

Choose the correct answer from below:

**A 1D numpy array filled with values from 10 to 21**

**A 3X4 matrix filled with values from 10 to 22**

**A 3X4 matrix filled with values from 10 to 21**

**A 1D numpy array filled with values from 10 to 22**

Question no :-

Given 3 arrays,

arr1 = np.array([1,2,3,6,3,2])   
arr2 = np.array([4,2,1,3,3,2])   
arr3 = np.zeros(len(arr1))

Which of the following are vectorized code(s) for array operations?

A.

for i in range(len(arr1)):   
 arr3[i] = arr1[i] \* arr2[i]

B.

arr3 = arr1\*arr2

C.

for i in range(len(arr1)):   
 if(arr1[i] < 0 ):   
 arr1[i] = -1   
 else:   
 arr1[i] = 1

D.

np.where(arr1 > 0, 1, -1)

Choose the correct answer from below**, please note that this question may have multiple correct answers**

A

B

C

D

Question no :-

Which of the following code will **NOT throw** an error?

**A.**

arr1 = np.array([1,2,3])  
arr2 = np.array([9,8,7])  
np.dot(arr1, arr2)

**B.**

arr1 = np.array([[1,2], [3,4]])  
arr2 = np.array([[1], [2]])  
np.dot(arr1, arr2)

**C.**

arr1 = np.array([1,2,3])  
k = 3  
np.dot(arr1, k)

**D.**

arr1 = np.array([[1,2], [3,4]])  
arr2 = np.array([1,1])  
np.dot(arr1, arr2)

Choose the correct answer from below**, please note that this question may have multiple correct answers**

A

B

C

D

Question no :-

What would the following code do?

import numpy as np

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

print(A[:, ::-1])

Choose the correct answer from below:

**Reverse the rows of a 2D array A**

**Reverse the columns of a 2D array A**

**Reverse both rows and columns of a 2D Array A**

**None of the above**

Question no :-

Given a list of birds and their corresponding age, calculate the mean age of the Crane bird (rounded off to 2 decimal points)

**Input Format:**

Two 1D array list i.e. bird array and age array

**Output Format:**

Float value representing mean age of crane birds

**Sample Input:**

birds = ['spoonbills', 'plovers', 'plovers', 'plovers', 'plovers', 'Cranes', 'plovers', 'plovers', 'Cranes', 'spoonbills']

age = [5.5, 6.0, 3.5, 1.5, 3.0, 4.0, 3.5, 2.0, 5.5, 6.0]

**Sample Output:**

4.75

**Note:**

To round off the result, use np.round()  
Documentation link: [np.round()](https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.round_.html)

import numpy as np

def calculate\_mean\_age(birds, age):

    mean\_age = None

    ## STEP1. Create mask to get Crane birds from birds array

    mask = birds[birds == 'Cranes']

    ## STEP2. Get the age of crane birds

    crane\_ages = age[birds == 'Cranes']

    # ## STEP 3. Calculate mean age of crane birds

    mean\_age = np.mean(age[birds == 'Cranes'])

    # ## STEP 4. Round off the mean age to 2 decimal points

    mean\_age = np.round(np.mean(age[birds == 'Cranes']),2)

    return mean\_age

Question no :-

Given a array of bird names and another array with corresponding ages of the birds, find the name of the oldest bird in the list.

**Input Format:**

Input has two lines.

First line is the array of bird names (String).

Second line is the array of ages of the birds (Int).

**Output Format:**

string

**Sample Input:**

['sparrow', 'peacock', 'parrot', 'owl', 'peacock', 'macaw', 'macaw', 'parrot', 'macaw', 'peacock']

[6, 1, 6, 5, 7, 6, 0, 9, 0, 7]

**Sample Output:**

parrot

**Explanation:**

parrot has age 9 which is the max of all the ages in the array; therefore parrot is returned.

**Note:**

Recall how to get index of maximum element

import numpy as np

def oldest\_bird(birds, age):

    ''' birds[i] consist of the names of the type of ith bird

        age[i] consist of the age of ith bird'''

    ## STEP 1: Get the index of maximum age element

    max\_age\_index = np.argmax(age)

    ## STEP 2: Get the bird with maxium age using the above index

    old\_bird = birds[max\_age\_index]

    return old\_bird

END OF NUMPY 3

Question no :-

Given a NumPy array of shape (n,m). Add padding of a layer of 0’s on all 4 boundaries of the matrix.

**Input Format:**

First line will be consisting of two space-separated integers representing n and m.

There will be n lines of input consisting of m space-separated integers representing the elements of rows of the array.

**Output Format:**

A 2d numpy array.

**Sample Input:**

3 2

1 2

3 4

5 6

**Sample Output:**

[[0 0 0 0]

[0 1 2 0]

[0 3 4 0]

[0 5 6 0]

[0 0 0 0]]

import numpy as np

def add\_padding(mat):

    '''mat-> NumPy array

       output-> NumPy array is expected to be returned'''

    # YOUR CODE GOES HERE

    res = None

    n = len(mat)

    m = len(mat[0])

    intmdt\_mat1 = np.vstack((mat, [0]\*m))

    intmdt\_mat2 = np.vstack(([0]\*m, intmdt\_mat1))

    a = np.array([0]\*(n+2)).reshape(n+2, 1)

    intmdt\_mat3 = np.hstack((intmdt\_mat2, a))

    res = np.hstack((a, intmdt\_mat3))

    return res

Question no :-

Given an MxN 2D array (M >= 4),

Split the array **column wise** such that,

1. 1st sub array contains the first 2 columns
2. 2nd sub array contains the 3rd column
3. 3rd sub array contains the rest of the columns

**Input Format:**

A 2D array

**Output Format:**

List of arrays

**Sample Input:**

[[0, 1, 2, 3],

[4, 5, 6, 7],

[8, 9, 10, 11],

[12, 13, 14, 15],

[16, 17, 18, 19],

[20, 21, 22, 23]]

**Sample Output:**

[

array([[ 0, 1],

[ 4, 5],

[ 8, 9],

[12, 13],

[16, 17],

[20, 21]]),

array([[ 2],

[ 6],

[10],

[14],

[18],

[22]]),

array([[ 3],

[ 7],

[11],

[15],

[19],

[23]])]

**Output explanation:**

* Here, the first sub-array contains the first two columns of the input array.
* Second sub-array contains the third column of the input array.
* Third sub-array contains the rest of the columns, i.e the fourth column of the input array.
* subarrays = None
* ### CODE starts here
* first\_subarray = arr[:,0:2]
* second\_subarray = arr[:,2:3]
* third\_subarray = arr[:,3:]
* subarrays = [first\_subarray,second\_subarray,third\_subarray]
* return subarrays

Question no :-

Given an 1D array and an integer k that specifies the number of equal parts to split the array into,

Perform the following operations:

1. Split the array into k number of equal parts.
2. Return the list of split arrays.

**Assumption**: The array can be split into k equal parts

**Note**: Recall how to split an array into equal parts.

**Input Format:**

Line separated numpy array and split count

**Output Format:**

List of numpy arrays

**Sample Input:**

arr = [0,1,2,4,5,6,7,8]

k = 3

**Sample Output:**

[array([0, 1, 2]), array([3, 4, 5]), array([6, 7, 8])]

import numpy as np

def split(arr, k):

    split\_arr = None

    ## CODE starts here

    split\_arr = np.split(arr,k)

    return split\_arr

Question no :-

What will be the output of following code?

import numpy as np  
arr = np.arange(16).reshape(4,4)

print(np.split(arr,4))

**A.**

Error

**B.**

[array([[ 0, 4, 8, 12]]),  
 array([[ 1, 5, 9, 13]]),  
 array([[ 2, 6, 10, 14]]),  
 array([[ 3, 7, 11, 15]])]

**C.**

[array([[0, 1, 2, 3]]),  
 array([[4, 5, 6, 7]]),  
 array([[ 8, 9, 10, 11]]),  
 array([[12, 13, 14, 15]])]

**D.**

[array([[1, 2, 3, 4]]),  
 array([[5, 6, 7, 8]]),  
 array([[ 9, 10, 11, 12]]),  
 array([[13, 14, 15, 16]])]

Choose the correct answer from below:

**A**

**B**

**C**

**D**

Question no :-

Which of the following will return a deep copy of array?

arr = np.array([1,2,3,0,-2,4])

**A.**arr1 = arr\*1

**B.**arr1 = arr[:]

**C.**arr1 = arr[arr > 0]

**D.**arr1 = arr.reshape(2,3)

Choose the correct answer from below**, please note that this question may have multiple correct answers**

A

B

C

D

Question no :-

Given a 3D array of shape (2, 3, 3)

array([[[ 0, 1, 2],  
 [ 3, 4, 5],   
 [ 6, 7, 8]],   
  
 [[ 9, 10, 11],   
 [12, 13, 14],   
 [15, 16, 17]]])

What will be the output of **arr[1, :, :]** ?

**A.**

array([[ 3, 4, 5],   
 [12, 13, 14]])

**B.**

array([[ 9, 10, 11],   
 [12, 13, 14],   
 [15, 16, 17]])

**C.**

array([[ 1, 4, 7],   
 [10, 13, 16]])

**D.**

array([[0, 1, 2],   
 [3, 4, 5],   
 [6, 7, 8]])

Choose the correct answer from below**, please note that this question may have multiple correct answers**

A

B

C

D

Question no :-

Given the following array:

import numpy as np  
arr = np.array([[1,2,3],  
 [4,5,6],  
 [7,8,9]])

Which options are **correct**?

**Note:** np is the alias for NumPy in the options.

Choose the correct answer from below**, please note that this question may have multiple correct answers**

np.hstack((arr, arr[:, 0])).shape = (3, 4)

np.hstack((arr, arr[:, [0]])).shape= (3, 4)

np.hstack((arr, arr[:, [0]])).shape= (4, 3)

np.hstack((arr,arr[:, 0])) => Throws Error

Question no :-

Which of the following is used to stack arrays column wise?

Choose the correct answer from below:

**hsplit()**

**vsplit()**

**hstack()**

**vstack()**

Question no :-

Which of the following code snippet will raise an error?

**A.**

arr = np.arange(10)  
np.split(arr, 3)

**B.**

arr = np.arange(10)  
np.split(arr, 5)

**C.**

arr = np.arange(10)  
np.split(arr, [2,5])

**D.**

arr = np.arange(10)  
np.split(arr, [5, 12])

Choose the correct answer from below:

**A**

**B**

**C**

**D**

Question no :-

Given a 2D array, which of the following will split the array**row-wise**?

Choose the correct answer from below:

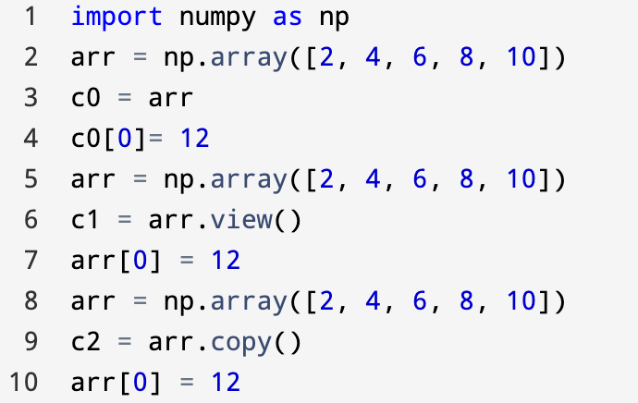
**hsplit()**

**vsplit()**

**hstack()**

**vstack()**

Question no :-



According to the above code snippet, which **two** options are **correct**?

Choose the correct answer from below**, please note that this question may have multiple correct answers**

c1 represents shallow copy and c2 represents deep copy

c1 represents deep copy and c2 represents shallow copy.

If arr and c1 are printed after line 4 and 7 respectively, they would have printed 12 as first element.

If c1 and c2 are printed after line 7 and 10 respectively, they would print 12 as first element of array.

Question no :-

Given a 3D array of shape (2,2,2)

arr = array([[[0, 1],   
 [2, 3]],   
   
 [[4, 5],   
 [6, 7]]])

What will be the output of **arr[:, 0:,:1]**?

**A.**

array([[2, 3],   
 [6, 7]])

**B.**

array([[0, 1],   
 [4, 5]])

**C.**

array([[[0],   
 [2]],   
  
 [[4],   
 [6]]])

**D.**

array([[[0, 1],   
 [2, 3]],   
  
 [[4, 5],   
 [6, 7]]])

Choose the correct answer from below:

**A**

**B**

**C**

**D**

Question no :-

Complete the Python function named sorted\_matrix\_multiplication that takes two matrices as input. The function should first sort each row of the first matrix in ascending order, and then perform a matrix multiplication with the second matrix.

**Input Format**:

matrix\_a: Numpy Array

matrix\_b: Numpy Array

**Output Format**

Numpy Array

**Sample Input**

[[3, 2], [1, 4], [5, 0]]

[[1, 2], [3, 4]]

**Sample Output**

[[11, 16], [13, 18], [15, 20]]

import numpy as np

def sorted\_matrix\_multiplication(matrix\_a, matrix\_b):

    # code here

    frstmat = np.sort(matrix\_a, axis=1)

    matmul = np.matmul(frstmat,matrix\_b)

    return matmul

Question no :-

Given the following Python code that uses NumPy for matrix multiplication:

import numpy as np

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

B = np.array([[1, 2]])

np.dot(A, B)

This code will produce a shape mismatch error. Which of the following modifications to the code will fix the shape so that the dot product happens successfully?

Choose the correct answer from below**, please note that this question may have multiple correct answers**

A = A.T

B = B.reshape(2,1)

B = B.T

A = A.reshape(2,2)

Question no :-

Complete the function slice\_and\_sum that slices a 2D NumPy array from start\_row to end\_row and start\_col to end\_col, and returns the sum of the sliced array.

**Input Format**:

array: Numpy Array

start\_row: int

end\_row: int

start\_col: int

end\_col: int

**Output Format**:

integer representing sum of corresponding array slice

**Sample Input**

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

**Sample Output**

12

**Explanation**

sliced array will be:

[[1 2]

[4 5]]

With total sum as 12

import numpy as np

def slice\_and\_sum(array, start\_row, end\_row, start\_col, end\_col):

    # write your code here

    slice1 = array[start\_row:end\_row, start\_col:end\_col]

    sum1 = np.sum(slice1)

    return sum1

END OF NUMPY 4