Q1. Wise split

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]])]
```

Q2. Split second

Problem statement:

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])]

Write your code in the following format:

```
import numpy as np

def split(arr, k):
    '''
    INPUT: arr, k

    OUTPUT: split_arr -> list of arrays
    '''

    ## Your Code goes here


    return split_arr
```

Q3. Split the difference

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])
```

Q4. Column split

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.

Write your code in the following format:

```
import numpy as np

def split(arr):
    '''
    INPUT: arr -> 2D array

    OUTPUT: subarrays -> list of 2D arrays
    '''

    ### Your code goes here


    return subarrays
```

Q5. Row vs column

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

A. Hsplit()
B. Vsplit()
C. Hstack()
D. Vstack()

Q6. Stack up

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

A. Hsplit()
B. Vsplit()
C. Hstack()
D. Vstack()

Q7. Inter dimension

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]])
```

Q8. Dark dimension

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]]])
```

Q9. Copy types

```
1  import numpy as np
2  arr = np.array([2, 4, 6, 8, 10])
3  c0 = arr
4  c0[0]= 12
5  arr = np.array([2, 4, 6, 8, 10])
6  c1 = arr.view()
7  arr[0] = 12
8  arr = np.array([2, 4, 6, 8, 10])
9  c2 = arr.copy()
10 arr[0] = 12
```

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

A. c1 represents shallow copy and c2 represents deep copy.
B. c1 represents deep copy and c2 represents shallow copy.
C. If arr and c1 are printed after line 4 and 7 respectively, they would have printed 12 as first element.
D. If c1 and c2 are printed after line 7 and 10 respectively, they would print 12 as first element of array.

*There may be more than one correct answer to this question. Please select all which you feel correct.

Q10. Filter Copy

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)

*There may be more than one correct answer to this question. Please select all which you feel correct.

Q11. Hstack

Given the following array:

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

Which options are **correct**?

A. np.hstack((arr, arr[:, 0])).shape = (3, 4)
B. np.hstack((arr, arr[:, [0]])).shape= (3, 4)
C. np.hstack((arr, arr[:, [0]])).shape= (4, 3)
D. np.hstack((arr,arr[:, 0])) => Throws Error

*There may be more than one correct answer to this question. Please select all which you feel correct.

Q12. Add padding

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]]
```

```python
import numpy as np
def add_padding(mat):
    '''mat-> NumPy array
       output-> NumPy array is expected to be returned'''

    # YOUR CODE GOES HERE

    return res
```

Q13. Changing Dimension

Given two numpy arrays **A** and **B** such that `A.shape = (3,3,1)` and `B.shape =(3,3)`

```
a) A = A.squeeze(axis=2)

b) B = np.expand_dims(B, axis=2)
```

After executing the above lines of code, the **shape** of arrays A and B will be?

A. A.shape=(3,3), B.shape=(3,3,1)
B. A.shape=(3,1,3), B.shape=(3,1,3)
C. A.shape=(3,3,1), B.shape=(3,3)
D. None of these