In [3]:
```
import numpy as np
```

In [8]:
```
n=np.array([[4,6,8],[8,9,10]])
```

In [9]:
```
n
```

Out[9]:
```
array([[ 4,  6,  8],
       [ 8,  9, 10]])
```

In [10]:
```
m=np.array([[6,9,4],[3,5,6]])
```

In [11]:
```
m
```

Out[11]:
```
array([[6, 9, 4],
       [3, 5, 6]])
```

In [12]:
```
d=n+m
print(d)
```

```
[[10 15 12]
 [11 14 16]]
```

In [ ]:

In [ ]:

Que-2 Given a numpy array (matrix), how to get a numpy array output which is equal to the original matrix multiplied by a scalar?

In [11]:
```
import numpy as np
```

In [12]:
```
a=np.array([[4,5,6],[2,3,6]])
```

In [13]:
```
a
```

Out[13]:
```
array([[4, 5, 6],
       [2, 3, 6]])
```

In [16]:
```
z=np.array((a*2))
```

In [17]:
```
z
```

Out[17]:
```
array([[ 8, 10, 12],
       [ 4,  6, 12]])
```

In [18]:
```
a.shape
```

Out[18]:
```
(2, 3)
```

Que-3 Create an identity matrix of dimension 4-by-4.

In [20]:
```
import numpy as np
```

In [21]:
```
i=np.eye(4)
```

In [22]:
```
i
```

Out[22]:
```
array([[1., 0., 0., 0.],
       [0., 1., 0., 0.],
       [0., 0., 1., 0.],
       [0., 0., 0., 1.]])
```

In [23]:
```
i.shape
```

Out[23]:
```
(4, 4)
```

In [24]:
```
i.ndim
```

Out[24]:
```
2
```

que-4 Convert a 1-D array to a 3-D array

In [4]:
```
import numpy as np
```

In [34]:
```
np.array([6,7,8,9,10,11,12,13]).reshape(2,2,2)
```

Out[34]:
```
array([[[ 6,  7],
        [ 8,  9]],

       [[10, 11],
        [12, 13]]])
```

Que-5 Convert a binary numpy array (containing only 0s and 1s) to a boolean numpy array

In [35]:
```
import numpy as np
```

In [36]:
```
b=np.array([[1,0,1,0],[0,1,0,1]])
```

In [37]:
```
b
```

Out[37]:
```
array([[1, 0, 1, 0],
       [0, 1, 0, 1]])
```

In [40]:
```
z=b.astype('bool')
```

In [41]:
```
print(z)
```

```
[[ True False  True False]
 [False  True False  True]]
```

Que-6 Stack 2 numpy arrays horizontally i.e., 2 arrays having the same 1st dimension (number of rows in 2D arrays)

In [42]:
```
a=np.array([2,4,6,8])
```

In [43]:
```
b=np.array([10,12,14,16])
```

In [44]:
```
n=np.array((a,b))
```

In [45]:
```
print(n)
```

```
[[ 2  4  6  8]
 [10 12 14 16]]
```

In [46]:
```
n.shape
```

Out[46]:
```
(2, 4)
```

In [47]:
```
n.ndim
```

Out[47]:
```
2
```

Que-7 Convert all the elements of a numpy array from float to integer datatype

In [86]:
```
import numpy as np
```

In [87]:
```
n=np.array([[1.1, 2.5, 3.5],[4.5, 5.5, 6.5]])
```

In [89]:
```
o = n.astype('int')
```

In [90]:
```
print(o)
```

```
[[1 2 3]
 [4 5 6]]
```

Que-8 Output a sequence of equally gapped 5 numbers in the range 0 to 100 (both inclusive

In [93]:
```
import numpy as np
```

In [94]:
```
n=np.arange(0,101,5)
```

In [95]:
```
print(n)
```

```
[  0   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85
  90  95 100]
```

Que-9 Output a matrix (numpy array) of dimension 2-by-3 with each and every value equal to 5

In [96]:
```
import numpy as np
```

In [102…
```
x=np.ones((2,3))
```

In [103…
```
y=5*x
```

In [104…
```
print(y)
```

```
[[5. 5. 5.]
 [5. 5. 5.]]
```

Que-10 Given 2 numpy arrays as matrices, output the result of multiplying the 2 matrices (as a numpy array)

In [105…
```
import numpy as np
```

In [107…
```
p=np.array([[1,2,3], [4,5,6],[7,8,9]])
```

In [108…
```
q=np.array([[3,4,5],[6,7,8],[9,6,4]])
```

In [109…
```
r=p*q
```

In [110…
```
r
```

Out[110]:
```
array([[ 3,  8, 15],
       [24, 35, 48],
       [63, 48, 36]])
```

Que-11 Output the array element indexes such that the array elements appear in the ascending order

In [111…

In [ ]: