```
#creating 2d array
```

```
import numpy as np
```

```
m1 = np.array([[1,2,3],[3,4,5]])
m1
```

```
array([[1, 2, 3],
       [3, 4, 5]])
```

```
m1.shape
```

```
(2, 3)
```

```
m1.ndim
```

```
2
```

```
len(m1)
```

```
2
```

```
#creating 1d array to convert into 2d
```

```
m2 = np.arange(1,13)
m2
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
m2.reshape(4,3)
```

```
array([[ 1,  2,  3],
       [ 4,  5,  6],
       [ 7,  8,  9],
       [10, 11, 12]])
```

```
m2.reshape(12,1)
```

```
array([[ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10],
       [11],
       [12]])
```

```
#transpose
# converting rows into colunms and columns into rows
```

```
m3 = m2.reshape(3,4)
m3
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
m3.T
```

```
array([[ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11],
       [ 4,  8, 12]])
```

```
np.transpose(m3)
```

```
array([[ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11],
       [ 4,  8, 12]])
```

```
#Flattening
# converting 2d array or n dimensional array to 1D array.
```

```
m3
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
m3.flatten()
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
#INDEXING
```

```
m3
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
m3[1][2]
```

```
7
```

```
m3[2,0]
```

```
9
```

```
m2
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
m2[[0,3,2,1]]
```

```
array([1, 4, 3, 2])
```

```
m3[[0,1,2],[0,1,2]]
```

```
array([ 1,  6, 11])
```

```
#slicing in 2D arrays
```

```
m3
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
m3[:2]
```

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

```
m3[:,:2]
```

```
array([[ 1,  2],
       [ 5,  6],
       [ 9, 10]])
```

```
m3[1:3 , 1:4]
```

```
array([[ 6,  7,  8],
       [10, 11, 12]])
```

```
m3[1:,1:3]
```

```
array([[ 6,  7],
       [10, 11]])
```

```
m3[:,1::2]
```

```
array([[ 2,  4],
       [ 6,  8],
       [10, 12]])
```

```
#quiz 2
a = np.array([0,1,2,3,4,5])
mask = (a%2 == 0)
a[mask] = -1
print(a)
```

```
[-1  1 -1  3 -1  5]
```

```
# quiz 1
a = [1,2,3,4,5]
b = [8,7,6]
a[3:] = b[::-2]
print(a)
```

```
[1, 2, 3, 6, 8]
```

```
 # quiz 3
 arr = np.array([-3,4,27,34,-2, 0, -45,-11,4, 0])
print(np.where(arr))
```

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

```
m3[[0,2],0:2]
```

```
array([[ 1,  2],
       [ 9, 10]])
```

```
#fancy indexing[masking]
# indexing through boolean values
```

```
m3
```

```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
m3 > 0
```

```
array([[ True,  True,  True,  True],
       [ True,  True,  True,  True],
       [ True,  True,  True,  True]])
```

```
m3[m3 < 6]
```

```
array([1, 2, 3, 4, 5])
```

```
m3[m3 % 2 == 0]
```

```
array([ 2,  4,  6,  8, 10, 12])
```

```
# aggregate functions
m2
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
m2.sum()
```

```
78
```

```
np.sum(m2)
```

```
78
```

```
m2.mean()
```

```
6.5
```

```
np.mean(m2)
```

```
6.5
```

```
m2.min()
```

```
1
```

```
m2.max()
```

```
12
```

```
np.max(m2)
```

```
12
```

```
np.min(m2)
```

```
1
```

```
np.sum(m3, axis=1)
```

```
array([10, 26, 42])
```

```
np.sum(m3, axis=0)
```

```
array([15, 18, 21, 24])
```

```
#logical operations
# 1.ANY()
# 2. all()
# 3. where()
m2
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
a = np.array([1,2,3,4])
b = np.array([4,3,2,1])
np.any(a<b)
```

```
True
```

```
np.all(a<b)
```

```
False
```

```
#prices of items on your shopping list
prices = np.array([50, 45, 25, 20, 35])
budget = 30
can_afford = np.any(prices<=budget)
can_afford
```

```
True
```

```
prices = np.array([50, 45, 25, 20, 35])
budget = 30
can_afford = np.all(prices<=budget)
can_afford
```

```
False
```

```
#where()
#give 10% discount on prices above 50
prices = np.array([45,55,60,75,40,90])
np.where(prices>50,prices*0.9,prices)
```

```
array([45. , 49.5, 54. , 67.5, 40. , 81. ])
```

```
np.where(m3<6)
```

```
(array([0, 0, 0, 0, 1]), array([0, 1, 2, 3, 0]))
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.