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

```
In [16]: x=np.array([11,2,23,34,34])
```

```
In [17]: np.min(x)

Out[17]: 2
```

```
In [18]: np.max(x)

Out[18]: 34
```

```
In [19]: np.mean(x)

Out[19]: 20.8
```

```
In [20]: np.median(x)

Out[20]: 23.0
```

```
In [21]: np.var(x)

Out[21]: 160.56
```

```
In [22]: np.std(x)

Out[22]: 12.671227249165726
```

```
In [25]: x1=np.array([[1,2,3,4],[15,12,14,15]])
         x1

Out[25]: array([[ 1,  2,  3,  4],
                [15, 12, 14, 15]])
```

```
In [26]: np.min(x1,axis=0)

Out[26]: array([1, 2, 3, 4])
```

```
In [27]: np.max(x1,axis=1)

Out[27]: array([ 4, 15])
```

```
In [28]: np.mean(x1,axis=0)

Out[28]: array([8. , 7. , 8.5, 9.5])
```

```
In [29]: np.mean(x1,axis=1)

Out[29]: array([ 2.5, 14. ])
```

```
In [30]: np.median(x1,axis=0)

Out[30]: array([8. , 7. , 8.5, 9.5])
```

```
In [31]: np.median(x1,axis=1)

Out[31]: array([ 2.5, 14.5])
```

```
In [32]: np.var(x1,axis=0)

Out[32]: array([49.  , 25.  , 30.25, 30.25])
```

```
In [33]: np.var(x1,axis=1)

Out[33]: array([1.25, 1.5 ])
```

```
In [34]: #two more imp for mean and meadian
         #argmin,argmax
```

```
In [35]: x=np.array([2,3,4,5,6,7])
```

```
In [36]: x[1]

Out[36]: 3
```

```
In [37]: np.argmin(x)

Out[37]: 0

In [39]: s=np.array([[3,4,5,6,7],[4,5,6,6,1]])

In [40]: np.argmin(s)

Out[40]: 9

In [43]: np.argmax(s)

Out[43]: 4

In [42]: s

Out[42]: array([[3, 4, 5, 6, 7],
                [4, 5, 6, 6, 1]])

In [44]: np.argmax(s,axis=0)

Out[44]: array([1, 1, 1, 0, 0], dtype=int64)

In [47]: #numpy(L)--random(m)---random numbers

In [101]: np.random.rand()

Out[101]: 0.2049914252438948

In [102]: np.random.randint(1,3,(2,6))#randomly select row,column,integer

Out[102]: array([[2, 2, 1, 2, 1, 2],
                 [1, 2, 1, 2, 2, 1]])

In [103]: np.random.rand(2,3)#0 and 1 not including 1

Out[103]: array([[0.4731568 , 0.07106096, 0.22246034],
                 [0.40084203, 0.79773003, 0.31112617]])
```

```
In [104]: np.random.randn()

Out[104]: 1.584885851045635

In [105]: np.random.sample()

Out[105]: 0.5987371766882608

In [14]: x=np.random.randint(1,100,(5,15))
         x

Out[14]: array([[51,  7, 68, 44, 34, 13, 83, 79, 26, 12, 35, 77, 67, 15, 27],
                [76, 94, 11, 70, 74,  4, 62, 41, 97, 25, 57, 93, 73, 29, 67],
                [75, 19, 99, 33, 77, 50, 22, 44, 15, 23, 90, 28, 85, 81, 92],
                [35, 25, 68,  4, 51, 45, 90, 56, 89,  7, 51,  2,  7, 79, 11],
                [32, 63, 91, 55,  6,  4, 61, 81, 77, 91, 10, 46, 30, 15, 20]])

In [15]: np.max(x)

Out[15]: 99

In [16]: np.argmax(x)

Out[16]: 32

In [17]: np.min(x,axis=0)

Out[17]: array([32,  7, 11,  4,  6,  4, 22, 41, 15,  7, 10,  2,  7, 15, 11])

In [18]: np.max(x,axis=0)

Out[18]: array([76, 94, 99, 70, 77, 50, 90, 81, 97, 91, 90, 93, 85, 81, 92])

In [19]: np.mean(x,axis=0)

Out[19]: array([53.8, 41.6, 67.4, 41.2, 48.4, 23.2, 63.6, 60.2, 60.8, 31.6, 48.6,
                49.2, 52.4, 43.8, 43.4])
```

```
In [20]: np.median(x,axis=0)

Out[20]: array([51., 25., 68., 44., 51., 13., 62., 56., 77., 23., 51., 46., 67.,
                29., 27.])
```

```
In [21]: np.var(x,axis=0)

Out[21]: array([ 355.76, 1037.44,  947.44,  495.76,  697.04,  406.96,  562.64,
                 286.96, 1135.36,  927.04,  693.04, 1075.76,  852.64,  900.16,
                 957.04])
```

```
In [22]: np.var(x,axis=1)

Out[22]: array([663.04888889, 828.82666667, 890.24888889, 914.08888889,
                905.04888889])
```

```
In [23]: np.std(x,axis=0)

Out[23]: array([18.8616012 , 32.20931542, 30.78051332, 22.26566864, 26.40151511,
                20.17324961, 23.72003373, 16.93989374, 33.6951035 , 30.44733157,
                26.32565289, 32.79878047, 29.2       , 30.00266655, 30.9360631 ])
```

```
In [43]: x=np.array([11,2,23,34,34,1,2,3,4,5,23])
```

```
In [44]: x1=np.unique(x,)
         x1

Out[44]: array([ 1,  2,  3,  4,  5, 11, 23, 34])
```

```
In [26]: x1=np.unique(x,return_counts=True)
         x1

Out[26]: (array([ 1,  2,  3,  4,  5, 11, 23, 34]),
          array([1, 2, 1, 1, 1, 1, 2, 2], dtype=int64))
```

In [91]:

```
x1=np.unique(x,return_counts=True,return_index=True)
x1
```

Out[91]:

```
(array([1, 2, 3, 4, 5]),
 array([0, 1, 2, 3, 4], dtype=int64),
 array([1, 1, 1, 1, 1], dtype=int64))
```

In [92]:

```
unique_x1#doubt
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_9172\3939876425.py in <module>
----> 1 unique_x1#doubt

NameError: name 'unique_x1' is not defined
```

In [93]:

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

In [94]:

```
np.repeat(x,3)#it is repeating the elements inside the array
```

Out[94]:

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

In [95]:

```
np.tile(x,(2,3))#repeat the entir array
```

Out[95]:

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

In [96]:

```
np.repeat(x[2],4)
```

Out[96]:

```
array([3, 3, 3, 3])
```

In [99]:

```
x
```

Out[99]:

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

In [100]:

```
np.random.choice(x,size=(2,3))
```

Out[100]:

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

```
In [101]: np.random.choice(x,size=10)

Out[101]: array([2, 4, 1, 1, 4, 3, 2, 2, 3, 1])
```

```
In [35]: unique_e,index_v,count_v=np.unique(x,return_counts=True,return_index=True)
```

```
In [36]: unique_e

Out[36]: array([1, 2, 3, 4])
```

```
In [37]: index_v

Out[37]: array([0, 1, 2, 3], dtype=int64)
```

```
In [38]: count_v

Out[38]: array([1, 1, 1, 1], dtype=int64)
```

```
In [39]: #practice
```

```
In [40]: np.random.randint(2,5,(5,10))#2 to 5 not including 5

Out[40]: array([[4, 4, 3, 4, 3, 3, 2, 2, 2, 3],
                [3, 3, 4, 3, 3, 4, 2, 3, 2, 3],
                [2, 3, 3, 4, 2, 2, 2, 4, 3, 2],
                [4, 2, 3, 3, 2, 3, 4, 2, 2, 3],
                [3, 2, 3, 4, 4, 3, 2, 2, 3, 2]])
```

```
In [47]: np.random.randint(1,3,(2,6))#randomly select row,column,integer

Out[47]: array([[1, 1, 1, 2, 2, 1],
                [1, 1, 2, 2, 2, 1]])
```

In [48]:

```
x=np.random.randint(2,5,(4,5))
x
```

Out[48]:

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

In [49]:

```
np.unique(x)
```

Out[49]:

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

In [50]:

```
np.unique(x,return_counts=True,return_index=True)
```

Out[50]:

```
(array([2, 3, 4]),
 array([0, 6, 1], dtype=int64),
 array([9, 5, 6], dtype=int64))
```

In [51]:

```
x1=np.unique(x,return_counts=True,return_index=True)
x1
```

Out[51]:

```
(array([2, 3, 4]),
 array([0, 6, 1], dtype=int64),
 array([9, 5, 6], dtype=int64))
```

In [69]:

```
#1
w=np.array([2,3,4])
```

In [70]:

```
np.repeat(w,3)
```

Out[70]:

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

In [71]:

```
np.tile(w,(3,4))
```

Out[71]:

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

In [57]:
```
#2
w=np.arange(1,10).reshape(3,3)
w
```

Out[57]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [58]:
```
np.repeat(w,2)
```

Out[58]:
```
array([1, 1, 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 7, 7, 8, 8, 9, 9])
```

In [59]:
```
np.tile(w,(2,3))#repeat the entir array
```

Out[59]:
```
array([[1, 2, 3, 1, 2, 3, 1, 2, 3],
       [4, 5, 6, 4, 5, 6, 4, 5, 6],
       [7, 8, 9, 7, 8, 9, 7, 8, 9],
       [1, 2, 3, 1, 2, 3, 1, 2, 3],
       [4, 5, 6, 4, 5, 6, 4, 5, 6],
       [7, 8, 9, 7, 8, 9, 7, 8, 9]])
```

In [66]:
```
#3
x=np.array([1,2,3,4])
```

In [67]:
```
np.repeat(x,3)
```

Out[67]:
```
array([1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4])
```

In [68]:
```

np.tile(x,(2,3))#repeat the entir array
```

Out[68]:
```
array([[1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4],
       [1, 2, 3, 4, 1, 2, 3, 4, 1, 2, 3, 4]])
```

In [ ]:
```
np.random.choice(x1,size=(2,3))
```

In [ ]:
```
#choice not get error if matrics equal to number of elements
```

```
In [86]: x=np.array([1,2,3,4,5])
         x
```

```
Out[86]: array([1, 2, 3, 4, 5])
```

```
In [87]: np.random.choice(x,size=(2,3))
```

```
Out[87]: array([[3, 4, 1],
                [5, 2, 2]])
```

```
In [88]: #choice get error if matrics not equal to number of elements
```

```
In [89]: y=np.array([2,3,4,5,6])
         y
```

```
Out[89]: array([2, 3, 4, 5, 6])
```

```
In [90]: np.random.choice(y,size=(2,3))
```

```
Out[90]: array([[6, 5, 6],
                [2, 5, 3]])
```

```
In [ ]:
```