In [2]:

```
import pandas as pd
import numpy as np
```

In [4]:

```
a = [1,2,3,4,5,6,"aman"]
print(a)
print(type(a))
```

```
[1, 2, 3, 4, 5, 6, 'aman']
<class 'list'>
```

In [5]:

```
a = np.array([1,2,3,4,5,6,"aman"])
print(a)
print(type(a))
```

```
['1' '2' '3' '4' '5' '6' 'aman']
<class 'numpy.ndarray'>
```

In [8]:

```
a = pd.Series([1,2,3,4,5,6,"aman"])
print(a)
print(type(a))
```

```
0       1
1       2
2       3
3       4
4       5
5       6
6    aman
dtype: object
<class 'pandas.core.series.Series'>
```

In [10]:

```
a = pd.Series([1,2,3,4,5,6,"aman"],index=['a','b','c','d','e','f','g'])
print(a)
print(type(a))
```

```
a       1
b       2
c       3
d       4
e       5
f       6
g    aman
dtype: object
<class 'pandas.core.series.Series'>
```

In [13]:

```
a = pd.Series([1,2,3,4,5,6],index=['a','b','c','d','e','f'],dtype=float)
print(a)
print(type(a))
```

```
a    1.0
b    2.0
c    3.0
d    4.0
e    5.0
f    6.0
dtype: float64
<class 'pandas.core.series.Series'>
```

In [16]:

```
a = pd.Series([1,2,3,4,5,6],index=['a','b','c','d','e','f'],dtype=float,name='xyz')
print(a)
print(type(a))
```

```
a    1.0
b    2.0
c    3.0
d    4.0
e    5.0
f    6.0
Name: xyz, dtype: float64
<class 'pandas.core.series.Series'>
```


In [18]:

```
a = pd.Series(5.2,index=['a','b','c','d','e','f'])
print(a)
print(type(a))
```

```
a    5.2
b    5.2
c    5.2
d    5.2
e    5.2
f    5.2
dtype: float64
<class 'pandas.core.series.Series'>
```

In [20]:

```
s1=pd.Series({'a':'ruchi','b':'cse','c':'aids'})
print(s1)
print(type(s1))
```

```
a    ruchi
b      cse
c     aids
dtype: object
<class 'pandas.core.series.Series'>
```

In [22]:

```
a = pd.Series([1,2,3,4,5,6,7,8,9],index=['a','b','c','d','e','f','g','h'],dtype=float,name='xyz')
rint(a)
print(type(a))p
```

```
a    1.0
b    2.0
c    3.0
d    4.0
e    5.0
f    6.0
Name: xyz, dtype: float64
<class 'pandas.core.series.Series'>
```

In [23]:

```
print(a['d'])
```

```
4.0
```

In [26]:

```
print(a['c':'g'])
```

```
c    3.0
d    4.0
e    5.0
f    6.0
Name: xyz, dtype: float64
```

In [27]:

```
max(a)
```

Out[27]:

```
6.0
```

In [28]:

```
min(a)
```

Out[28]:

```
1.0
```

In [30]:

```
s1=pd.Series([11,2,3,4,5,6,7,8,9,10])
print(s1)
print(type(s1))
```

```
0    11
1     2
2     3
3     4
4     5
5     6
6     7
7     8
8     9
9    10
dtype: int64
<class 'pandas.core.series.Series'>
```

In [31]:

```
s2=pd.Series([11.12,13,14,15,16,17,18,19,20])
print(s2)
print(type(s2))
```

```
0    11.12
1    13.00
2    14.00
3    15.00
4    16.00
5    17.00
6    18.00
7    19.00
8    20.00
dtype: float64
<class 'pandas.core.series.Series'>
```

In [32]:

```
print(s1+s2)
```

```
0    22.12
1    15.00
2    17.00
3    19.00
4    21.00
5    23.00
6    25.00
7    27.00
8    29.00
9      NaN
dtype: float64
```

Data Frame

In [34]:

```
data1 = pd.DataFrame([1,2,3,4,5,6])
data2 = pd.Series([1,2,3,4,5,6])
print("This is dataframe")
print(data1)
print("This is series")
print(data2)
```

```
This is dataframe
   0
0  1
1  2
2  3
3  4
4  5
5  6
This is series
0    1
1    2
2    3
3    4
4    5
5    6
dtype: int64
```


In [38]:

```
data3 = pd.DataFrame([[1,2,3],[1,2,3],[1,2,3]])
data3
```

Out[38]:

	0	1	2
0	1	2	3
1	1	2	3
2	1	2	3

In [40]:

```
data3 = pd.DataFrame([{'id':[1,2,3],'id2':[1,2,3],'id3':[1,2,3]}])
data3
```

Out[40]:

	id	id2	id3
0	[1, 2, 3]	[1, 2, 3]	[1, 2, 3]

In [42]:

```
data4= pd.DataFrame({'id':[1,2,3,4],"sn":[11,12,13,14]})
data4
```

Out[42]:

	id	sn
0	1	11
1	2	12
2	3	13
3	4	14

In [45]:

```
b = np.linspace(1,26,25).reshape(5,5)
print(b)
print(type(b))
```

```
[[ 1.          2.04166667  3.08333333  4.125       5.16666667]
 [ 6.20833333  7.25        8.29166667  9.33333333 10.375     ]
 [11.41666667 12.45833333 13.5        14.54166667 15.58333333]
 [16.625      17.66666667 18.70833333 19.75       20.79166667]
 [21.83333333 22.875      23.91666667 24.95833333 26.        ]]
<class 'numpy.ndarray'>
```

In [49]:

```
c = pd.DataFrame(b,dtype=int)
c
```

```
C:\Users\PC\AppData\Local\Temp\ipykernel_11000\3193869853.py:1: FutureWarning: In a future version, passing float-d
type values and an integer dtype to DataFrame will retain floating dtype if they cannot be cast losslessly (matchin
g Series behavior). To retain the old behavior, use DataFrame(data).astype(dtype)
  c = pd.DataFrame(b,dtype=int)
```

Out[49]:

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	26

In [51]:

```
a1 = pd.DataFrame({'id':[101,102,103,104,105,106,107],'name':['aman','shyam','ram','monu','pankaj','nikhil','sam'],'per':[95,85,7
a1
```

Out[51]:

	id	name	per
0	101	aman	95
1	102	shyam	85
2	103	ram	76
3	104	monu	64
4	105	pankaj	55
5	106	nikhil	44
6	107	sam	32

In [53]:

```
a2 = pd.DataFrame({'grade':['A+','A','B+','B','C+','C','D']})
a2
```

Out[53]:

	grade
0	A+
1	A
2	B+
3	B
4	C+
5	C
6	D

In [55]:

```
a1['grade']=a2
a1
```

Out[55]:

	grade
0	A+
1	A
2	B+
3	B
4	C+
5	C
6	D