

In [2]:

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

1. Create any Series and print the output

In [5]:

```
a=pd.Series([1,2,3,4,5])
a
```

Out[5]:

```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

2. Create any dataframe of 10x5 with few nan values and print the output

In [13]:

```
x=pd.DataFrame(  
{  
    "A": [1,2,3,4,5,6,7,8,9,10],  
    "B": [6,7,8,9,0,np.nan,np.nan,1,2,3],  
    "C": [1,4,6,8,4,2,np.nan,np.nan,np.nan,2],  
    "D": [3,5,6,np.nan,4,1,7,8,np.nan,np.nan],  
    "E": [5,np.nan,3,4,5,np.nan,np.nan,np.nan,2,1]  
}  
)  
x
```

Out[13]:

	A	B	C	D	E
0	1	6.0	1.0	3.0	5.0
1	2	7.0	4.0	5.0	NaN
2	3	8.0	6.0	6.0	3.0
3	4	9.0	8.0	NaN	4.0
4	5	0.0	4.0	4.0	5.0
5	6	NaN	2.0	1.0	NaN
6	7	NaN	NaN	7.0	NaN
7	8	1.0	NaN	8.0	NaN
8	9	2.0	NaN	NaN	2.0
9	10	3.0	2.0	NaN	1.0

3.Display top 7 and last 6 rows and print the output

In [16]:

```
x.head(7)
```

Out[16]:

	A	B	C	D	E
0	1	6.0	1.0	3.0	5.0
1	2	7.0	4.0	5.0	NaN
2	3	8.0	6.0	6.0	3.0
3	4	9.0	8.0	NaN	4.0
4	5	0.0	4.0	4.0	5.0
5	6	NaN	2.0	1.0	NaN
6	7	NaN	NaN	7.0	NaN

4. Fill with a constant value and print the output

In [22]:

```
d=pd.DataFrame(np.random.randn(3,4))  
d
```

Out[22]:

	0	1	2	3
0	-2.027046	0.835038	0.222293	-0.347093
1	-0.392422	-1.174279	0.524512	0.378444
2	1.950716	1.198610	-0.351603	-0.124885

5. Drop the column with missing values and print the output

In [20]:

```
x.dropna(axis=1,how='any')
```

Out[20]:

	A
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	10

6. Drop the row with missing values and print the output

In [19]:

```
x.dropna()
```

Out[19]:

	A	B	C	D	E
0	1	6.0	1.0	3.0	5.0
2	3	8.0	6.0	6.0	3.0
4	5	0.0	4.0	4.0	5.0

7. To check the presence of missing values in your dataframe

In [26]:

```
x.isna()
```

Out[26]:

	A	B	C	D	E
0	False	False	False	False	False
1	False	False	False	False	True
2	False	False	False	False	False
3	False	False	False	True	False
4	False	False	False	False	False
5	False	True	False	False	True
6	False	True	True	False	True
7	False	False	True	False	True
8	False	False	True	True	False
9	False	False	False	True	False

8. Use operators and check the condition and print the output

In [31]:

```
x[x["B"]>7]
```

Out[31]:

	A	B	C	D	E
2	3	8.0	6.0	6.0	3.0
3	4	9.0	8.0	NaN	4.0

9. Display your output using loc and iloc, row and column heading

In [32]:

```
x.loc[2:5]
```

Out[32]:

	A	B	C	D	E
2	3	8.0	6.0	6.0	3.0
3	4	9.0	8.0	NaN	4.0
4	5	0.0	4.0	4.0	5.0
5	6	NaN	2.0	1.0	NaN

10. Display the statistical summary of data

In [34]:

```
x.describe()
```

Out[34]:

	A	B	C	D	E
count	10.000000	8.000000	7.000000	7.000000	6.000000
mean	5.500000	4.500000	3.857143	4.857143	3.333333
std	3.02765	3.422614	2.478479	2.410295	1.632993
min	1.000000	0.000000	1.000000	1.000000	1.000000
25%	3.250000	1.750000	2.000000	3.500000	2.250000
50%	5.500000	4.500000	4.000000	5.000000	3.500000
75%	7.750000	7.250000	5.000000	6.500000	4.750000
max	10.000000	9.000000	8.000000	8.000000	5.000000

In []: