In [2]:

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

1. Create any Series and print the output

In [6]:

```
a=pd.Series([6,7,8,9,10])
a
```

Out[6]:

```
0     6
1     7
2     8
3     9
4    10
dtype: int64
```

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

In [15]:

```
a=pd.DataFrame(
{
    "A":45,
    "B":21,
    "C":pd.Series(np.nan,index=list(range(10))),
    "D":90
})
a
```

Out[15]:

	A	B	C	D
0	45	21	NaN	90
1	45	21	NaN	90
2	45	21	NaN	90
3	45	21	NaN	90
4	45	21	NaN	90
5	45	21	NaN	90
6	45	21	NaN	90
7	45	21	NaN	90
8	45	21	NaN	90
9	45	21	NaN	90

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

In [17]:

```
print(a.head(7))
print(a.tail(6))
```

```
    A   B   C   D
0  45  21 NaN  90
1  45  21 NaN  90
2  45  21 NaN  90
3  45  21 NaN  90
4  45  21 NaN  90
5  45  21 NaN  90
6  45  21 NaN  90
    A   B   C   D
4  45  21 NaN  90
5  45  21 NaN  90
6  45  21 NaN  90
7  45  21 NaN  90
8  45  21 NaN  90
9  45  21 NaN  90
```

4. Fill with a constant value and print the output

In [21]:

```
a.fillna(value=30)
```

Out[21]:

	A	B	C	D
0	45	21	30.0	90
1	45	21	30.0	90
2	45	21	30.0	90
3	45	21	30.0	90
4	45	21	30.0	90
5	45	21	30.0	90
6	45	21	30.0	90
7	45	21	30.0	90
8	45	21	30.0	90
9	45	21	30.0	90

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

In [23]:

```
a.dropna(axis=1)
```

Out[23]:

	A	B	D
0	45	21	90
1	45	21	90
2	45	21	90
3	45	21	90
4	45	21	90
5	45	21	90
6	45	21	90
7	45	21	90
8	45	21	90
9	45	21	90

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

In [25]:

```
a.dropna()
```

Out[25]:

	A	B	C	D

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

In [26]:

```
a.isna()
```

Out[26]:

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

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

In [39]:

```
c=np.arange(1,20)
e=c[(c>5)&(c<10)]
print(e)
```

```
[6 7 8 9]
```

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

In [34]:

```
print(a.loc[0:2])
print(a.iloc[0:2])
```

```
    A   B   C   D
0  45  21 NaN  90
1  45  21 NaN  90
2  45  21 NaN  90
    A   B   C   D
0  45  21 NaN  90
1  45  21 NaN  90
```

10. Display the statistical summary of data

In [31]:

```
a.describe()
```

Out[31]:

	A	B	C	D
count	10.0	10.0	0.0	10.0
mean	45.0	21.0	NaN	90.0
std	0.0	0.0	NaN	0.0
min	45.0	21.0	NaN	90.0
25%	45.0	21.0	NaN	90.0
50%	45.0	21.0	NaN	90.0
75%	45.0	21.0	NaN	90.0
max	45.0	21.0	NaN	90.0

In [ ]:

```

```