Deena 20104016

In [1]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
"Github Link
PROBLEM STATEMENT
1. Create any Series and print the output
2. Create any dataframe of 10x5 with few nan values and print the output
3.Display top 7 and last 6 rows and print the output
4. Fill with a constant value and print the output
5. Drop the column with missing values and print the output
6. Drop the row with missing values and print the output
7. To check the presence of missing values in your dataframe
8. Use operators and check the condition and print the output
9. Display your output using loc and iloc, row and column heading
10. Display the statistical summary of data
```

1. Create any Series and print the output

In [3]:
```
df=pd.Series([1,2,3,4,5,6,7,8,9])
df
```

Out[3]:
```
0    1
1    2
2    3
3    4
4    5
5    6
6    7
7    8
8    9
dtype: int64
```

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

In [5]:
```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df
```

Out[5]:

	a	b	c	d	e
0	1.0	3.0	2.0	56.0	12

	a	b	c	d	e
1	2.0	4.0	3.0	21.0	13
2	3.0	5.0	5.0	24.0	14
3	4.0	7.0	7.0	22.0	15
4	5.0	86.0	8.0	24.0	16
5	6.0	2.0	6.0	27.0	14
6	7.0	4.0	4.0	52.0	17
7	NaN	NaN	5.0	NaN	18
8	12.0	16.0	NaN	29.0	19
9	13.0	14.0	32.0	30.0	21

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

In [7]:

```
d=pd.DataFrame(
{
     "a":np.empty(20,dtype='int64'),
     "b":np.empty(20,dtype='int64'),
     "c":np.empty(20,dtype='int64'),
     "d":np.empty(20,dtype='int64')



})
d.head(7)
```

Out[7]:

	a	b	c	d
0	4622945017495814144	1758654366840	6653607755208884992	8030485320392769652
1	0	127	8935572783028534530	5940810858463130231
2	4623507967449235456	0	8962288613799127824	7237964107322753024
3	0	0	7711398413547440144	7021804519348502528
4	4624070917402656768	18296268623183872	7639086198161575940	143646523392
5	0	7310593858020254331	7205786898161237001	9695493533737330
6	4624633867356078080	3689064036650590820	4330628957000011017	7598266771532546048

In [8]:

```
d.tail(6)
```

Out[8]:

	a	b	c	d
14	4625759767262920704	7076342913183136290	-9007124481455920640	146368138958471438
15	0	4050254925114062947	-8069188292882358015	-68678751755108582
16	4626041242239631360	3835146281331012653	8940665703703409154	433751852451496450
17	0	7305183173089769265	8935168154143386112	80502977228178184

	a	b	c	d
18	4626604192193052672	2463524188219324469	7205784806007794176	146931080321827076
19	0	8319683848551211564	7638241307479154949	148056980229193988

4. Fill with a constant value and print the output

In [9]:
```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.fillna(value=0)
```

Out[9]:

	a	b	c	d	e
0	1.0	3.0	2.0	56.0	12
1	2.0	4.0	3.0	21.0	13
2	3.0	5.0	5.0	24.0	14
3	4.0	7.0	7.0	22.0	15
4	5.0	86.0	8.0	24.0	16
5	6.0	2.0	6.0	27.0	14
6	7.0	4.0	4.0	52.0	17
7	0.0	0.0	5.0	0.0	18
8	12.0	16.0	0.0	29.0	19
9	13.0	14.0	32.0	30.0	21

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

In [10]:
```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.isna()
```

Out[10]:

	a	b	c	d	e
0	False	False	False	False	False

	a	b	c	d	e
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
5	False	False	False	False	False
6	False	False	False	False	False
7	True	True	False	True	False
8	False	False	True	False	False
9	False	False	False	False	False

In [11]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.dropna(axis=0)
```

Out[11]:

	a	b	c	d	e
0	1.0	3.0	2.0	56.0	12
1	2.0	4.0	3.0	21.0	13
2	3.0	5.0	5.0	24.0	14
3	4.0	7.0	7.0	22.0	15
4	5.0	86.0	8.0	24.0	16
5	6.0	2.0	6.0	27.0	14
6	7.0	4.0	4.0	52.0	17
9	13.0	14.0	32.0	30.0	21

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

In [13]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.dropna()
```

Out[13]:

	a	b	c	d	e
0	1.0	3.0	2.0	56.0	12
1	2.0	4.0	3.0	21.0	13
2	3.0	5.0	5.0	24.0	14
3	4.0	7.0	7.0	22.0	15
4	5.0	86.0	8.0	24.0	16
5	6.0	2.0	6.0	27.0	14
6	7.0	4.0	4.0	52.0	17
9	13.0	14.0	32.0	30.0	21

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

In [14]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,np.nan,12,13],
    "b":[3,4,5,7,86,2,4,np.nan,16,14],
    "c":[2,3,5,7,8,6,4,5,np.nan,32],
    "d":[56,21,24,22,24,27,52,np.nan,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.isna()
```

Out[14]:

	a	b	c	d	e
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
5	False	False	False	False	False
6	False	False	False	False	False
7	True	True	False	True	False
8	False	False	True	False	False
9	False	False	False	False	False

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

In [15]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,5,12,13],
    "b":[3,4,5,7,86,2,4,9,16,14],
    "c":[2,3,5,7,8,6,4,5,6,32],
    "d":[56,21,24,22,24,27,52,4,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df[df["a"]>2]
```

Out[15]:

	a	b	c	d	e
2	3	5	5	24	14
3	4	7	7	22	15
4	5	86	8	24	16
5	6	2	6	27	14
6	7	4	4	52	17
7	5	9	5	4	18
8	12	16	6	29	19
9	13	14	32	30	21

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

In [16]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,5,12,13],
    "b":[3,4,5,7,86,2,4,9,16,14],
    "c":[2,3,5,7,8,6,4,5,6,32],
    "d":[56,21,24,22,24,27,52,4,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.loc[0:2]
```

Out[16]:

	a	b	c	d	e
0	1	3	2	56	12
1	2	4	3	21	13
2	3	5	5	24	14

In [17]:

```
df.iloc[0:5]
```

Out[17]:

	a	b	c	d	e
0	1	3	2	56	12

	a	b	c	d	e
1	2	4	3	21	13
2	3	5	5	24	14
3	4	7	7	22	15
4	5	86	8	24	16

10. Display the statistical summary of data

In [18]:

```
df=pd.DataFrame(
{
    "a":[1,2,3,4,5,6,7,5,12,13],
    "b":[3,4,5,7,86,2,4,9,16,14],
    "c":[2,3,5,7,8,6,4,5,6,32],
    "d":[56,21,24,22,24,27,52,4,29,30],
    "e":[12,13,14,15,16,14,17,18,19,21]
})
df.describe()
```

Out[18]:

	a	b	c	d	e
count	10.000000	10.000000	10.000000	10.000000	10.00000
mean	5.800000	15.000000	7.800000	28.900000	15.90000
std	3.966527	25.381533	8.689713	15.095621	2.84605
min	1.000000	2.000000	2.000000	4.000000	12.00000
25%	3.250000	4.000000	4.250000	22.500000	14.00000
50%	5.000000	6.000000	5.500000	25.500000	15.50000
75%	6.750000	12.750000	6.750000	29.750000	17.75000
max	13.000000	86.000000	32.000000	56.000000	21.00000