In [1]:

```
import pandas as pd
```

In [3]:

```
import numpy as np
```

Create a Series and print the output

In [4]:

```
series_data = pd.Series([10, 20, 30, 40, 50])
print(series_data)
```

```
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

Create a DataFrame of 10x5 with some NaN values and print the output

In [10]:

```
data = np.random.randint(1, 100, size=(10, 5)).astype(float)
data[1, 2] = np.nan
data[3, 4] = np.nan
df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D', 'E'])
print(df)
```

```
      A     B     C     D     E
0  85.0  87.0  82.0  12.0  89.0
1  53.0  60.0   NaN  54.0  10.0
2   7.0  85.0  17.0  50.0  28.0
3  48.0  29.0   2.0  16.0   NaN
4  29.0  10.0  43.0  84.0  64.0
5  92.0  94.0  21.0  59.0  13.0
6   2.0  54.0  90.0  20.0  70.0
7  54.0  23.0  89.0  64.0  33.0
8  83.0  90.0   1.0  83.0  34.0
9  42.0  60.0  66.0   6.0  46.0
```

Display top 7 and last 6 rows of the DataFrame and print the output

In [12]:

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

```
      A     B     C     D     E
0  85.0  87.0  82.0  12.0  89.0
1  53.0  60.0   NaN  54.0  10.0
2   7.0  85.0  17.0  50.0  28.0
3  48.0  29.0   2.0  16.0   NaN
4  29.0  10.0  43.0  84.0  64.0
5  92.0  94.0  21.0  59.0  13.0
6   2.0  54.0  90.0  20.0  70.0
      A     B     C     D     E
4  29.0  10.0  43.0  84.0  64.0
5  92.0  94.0  21.0  59.0  13.0
6   2.0  54.0  90.0  20.0  70.0
7  54.0  23.0  89.0  64.0  33.0
8  83.0  90.0   1.0  83.0  34.0
9  42.0  60.0  66.0   6.0  46.0
```

Fill NaN values with a constant value and print the output

In [13]:

```
filled_df = df.fillna(0)
print(filled_df)
```

```
      A     B     C     D     E
0  85.0  87.0  82.0  12.0  89.0
1  53.0  60.0   0.0  54.0  10.0
2   7.0  85.0  17.0  50.0  28.0
3  48.0  29.0   2.0  16.0   0.0
4  29.0  10.0  43.0  84.0  64.0
5  92.0  94.0  21.0  59.0  13.0
6   2.0  54.0  90.0  20.0  70.0
7  54.0  23.0  89.0  64.0  33.0
8  83.0  90.0   1.0  83.0  34.0
9  42.0  60.0  66.0   6.0  46.0
```

Drop the column with missing values and print the output

In [14]:

```
dropped_row_df = df.dropna()
print(dropped_row_df)
```

```
      A     B     C     D     E
0  85.0  87.0  82.0  12.0  89.0
2   7.0  85.0  17.0  50.0  28.0
4  29.0  10.0  43.0  84.0  64.0
5  92.0  94.0  21.0  59.0  13.0
6   2.0  54.0  90.0  20.0  70.0
7  54.0  23.0  89.0  64.0  33.0
8  83.0  90.0   1.0  83.0  34.0
9  42.0  60.0  66.0   6.0  46.0
```

Check the presence of missing values in the DataFrame

In [15]:

```
print(df.isnull().any())
```

```
A    False
B    False
C     True
D    False
E     True
dtype: bool
```

Use operators and check the condition and print the output

In [16]:

```
print(df[df > 50])
```

```
      A     B     C     D     E
0  85.0  87.0  82.0   NaN  89.0
1  53.0  60.0   NaN  54.0   NaN
2   NaN  85.0   NaN   NaN   NaN
3   NaN   NaN   NaN   NaN   NaN
4   NaN   NaN   NaN  84.0  64.0
5  92.0  94.0   NaN  59.0   NaN
6   NaN  54.0  90.0   NaN  70.0
7  54.0   NaN  89.0  64.0   NaN
8  83.0  90.0   NaN  83.0   NaN
9   NaN  60.0  66.0   NaN   NaN
```

Display output using loc and iloc, row, and column headings

```
In [17]: print(df.loc[2:4, 'B':'D'])
```

```
      B     C     D
2  85.0  17.0  50.0
3  29.0   2.0  16.0
4  10.0  43.0  84.0
```

```
In [18]: print(df.iloc[2:5, 1:4])
```

```
      B     C     D
2  85.0  17.0  50.0
3  29.0   2.0  16.0
4  10.0  43.0  84.0
```

Display the statistical summary of data

```
In [19]: print(df.describe())
```

```
               A          B          C          D          E
count  10.000000  10.000000   9.000000  10.000000   9.000000
mean   49.500000  59.200000  45.666667  44.800000  43.000000
std    31.202742  30.312447  36.939139  29.256718  26.669271
min     2.000000  10.000000   1.000000   6.000000  10.000000
25%    32.250000  35.250000  17.000000  17.000000  28.000000
50%    50.500000  60.000000  43.000000  52.000000  34.000000
75%    75.750000  86.500000  82.000000  62.750000  64.000000
max    92.000000  94.000000  90.000000  84.000000  89.000000
```

```
In [ ]:
```