2. Assignment on Practice of Pandas Library

Create Dataframe:

In [1]:

```
import pandas as pd
df = pd.DataFrame({'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]});
print(df)
```

```
    X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

Create DataSeries:

In [2]:

```
import pandas as pd
s = pd.Series([2, 4, 6, 8, 10])
print(s)
```

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

Creating 2D dataframe

In [3]:

```
import pandas as pd
import numpy as np
data = {
 'w': np.random.randn(5),
 'x': np.random.randn(5),
 'y': np.random.randn(5),
 'Z': np.random.randn(5)
}
index = ['a', 'b', 'c', 'd', 'e']
df = pd.DataFrame(data, index=index)
print(df)
```

```
          w         x         y         Z
a  0.788508  0.359622 -0.357006  0.065455
b  1.372744 -0.704107 -1.374672  0.064281
c -2.175178 -1.512153 -0.448932  0.629077
d  0.330367 -1.090455 -0.932629 -0.294877
e -0.875626 -0.426688  0.759883 -0.263554
```

Printing specific columns

In [5]:

```
selected_columns = df[['y', 'Z']]
print(selected_columns)
```

```
          y         Z
a -0.357006  0.065455
b -1.374672  0.064281
c -0.448932  0.629077
d -0.932629 -0.294877
e  0.759883 -0.263554
```

Creating new column by the help of existing columns

In [7]:

```
df['new'] = df['w'] + df['y']
print(df)
```

```
          w         x         y         Z       new
a  0.788508  0.359622 -0.357006  0.065455  0.431502
b  1.372744 -0.704107 -1.374672  0.064281 -0.001929
c -2.175178 -1.512153 -0.448932  0.629077 -2.624110
d  0.330367 -1.090455 -0.932629 -0.294877 -0.602262
e -0.875626 -0.426688  0.759883 -0.263554 -0.115743
```

Deleting column from the datasets

In [8]:

```
df = df.drop('new', axis=1)
print(df)
```

```
          w         x         y         Z
a  0.788508  0.359622 -0.357006  0.065455
b  1.372744 -0.704107 -1.374672  0.064281
c -2.175178 -1.512153 -0.448932  0.629077
d  0.330367 -1.090455 -0.932629 -0.294877
e -0.875626 -0.426688  0.759883 -0.263554
```

Displaying specific rows

In [10]:

```
row_b = df.loc['b']
print(row_b)
```

```
w    1.372744
x   -0.704107
y   -1.374672
Z    0.064281
Name: b, dtype: float64
```

performing slicing operations

In [11]:

```
slice1 = df.loc[['a', 'b'], ['y', 'Z']]
print(slice1)
slice2 = df.loc[['c', 'd'], ['x', 'y']]
print(slice2)
```

```
          y         Z
a -0.357006  0.065455
b -1.374672  0.064281
          x         y
c -1.512153 -0.448932
d -1.090455 -0.932629
```

Converting the values into binary form

In [12]:

```
threshold = 0
boolean_df = df > threshold
print(boolean_df)
```

```
       w      x      y      Z
a   True   True  False   True
b   True  False  False   True
c  False  False  False   True
d   True  False  False  False
e  False  False   True  False
```

performing some filtering operations

In [13]:

```
filtered_df = df[(df['w'] > 0) & (df['y'] > 1)]
print(filtered_df)
```

```
Empty DataFrame
Columns: [w, x, y, Z]
Index: []
```

Performing concatination operation

In [14]:

```
import pandas as pd
data1 = {'A': [1, 2, 3], 'B': [4, 5, 6]}
df1 = pd.DataFrame(data1)
print(df1)
data2 = {'A': [7, 8, 9], 'B': [10, 11, 12]}
df2 = pd.DataFrame(data2)
print(df2)
result = pd.concat([df1, df2])
print(result)
```

```
   A  B
0  1  4
1  2  5
2  3  6
   A   B
0  7  10
1  8  11
2  9  12
   A   B
0  1   4
1  2   5
2  3   6
0  7  10
1  8  11
2  9  12
```

Merging two data sets

In [15]:

```
import pandas as pd
data1 = {'key': ['A', 'B', 'C'], 'value1': [1, 2, 3]}
df1 = pd.DataFrame(data1)
print(df1)
data2 = {'key': ['B', 'C', 'D'], 'value2': [4, 5, 6]}
df2 = pd.DataFrame(data2)
print(df2)
merged_df = pd.merge(df1, df2, on='key')
print(merged_df)
```

```
  key  value1
0   A       1
1   B       2
2   C       3
  key  value2
0   B       4
1   C       5
2   D       6
  key  value1  value2
0   B       2       4
1   C       3       5
```

Performing join operation

In [16]:

```
import pandas as pd
data1 = {'A': [1, 2, 3], 'B': [4, 5, 6]}
df1 = pd.DataFrame(data1, index=['X', 'Y', 'Z'])
print(df1)
data2 = {'C': [7, 8, 9], 'D': [10, 11, 12]}
df2 = pd.DataFrame(data2, index=['Y', 'Z', 'W'])
print(df2)
joined_df = df1.join(df2, how='inner') # 'inner' will keep only common index values
print(joined_df)
```

```
   A  B
X  1  4
Y  2  5
Z  3  6
   C   D
Y  7  10
Z  8  11
W  9  12
```

```
   A  B  C   D
Y  2  5  7  10
Z  3  6  8  11
```