In [1]:
```
!pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\dhruv\appdata\local\program
s\python\python38\lib\site-packages (1.4.3)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\dhruv\appda
ta\local\programs\python\python38\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\dhruv\appdata\local\p
rograms\python\python38\lib\site-packages (from pandas) (2023.2)
Requirement already satisfied: numpy>=1.18.5 in c:\users\dhruv\appdata\local
\programs\python\python38\lib\site-packages (from pandas) (1.24.2)
Requirement already satisfied: six>=1.5 in c:\users\dhruv\appdata\local\progr
ams\python\python38\lib\site-packages (from python-dateutil>=2.8.1->pandas)
(1.16.0)
```

In [2]:
```
import pandas as pd
```

In [3]:
```
pd.__version__
```

Out[3]:
```
'1.4.3'
```

In [ ]:
```
Pandas -: Pandas is the fast, flexiable and easy to use open source data analy
    manipulation tool, which is built on top of the Python programming lanagua
```

In [ ]:
```
# Series -- > Series is one dimension arra like homegeneous data, homegeneous

# Data Frame --> Dataframe is the two dimensional array with hetrogeneous data
```

In [5]:
```
# How to create an empty Series

s = pd.Series()
s
```

```
C:\Users\DHRUV\AppData\Local\Temp\ipykernel_2428\20424751.py:3: FutureWarnin
g: The default dtype for empty Series will be 'object' instead of 'float64' i
n a future version. Specify a dtype explicitly to silence this warning.
  s = pd.Series()
```

Out[5]:
```
Series([], dtype: float64)
```

In [6]:
```
type(s)
```

Out[6]:
```
pandas.core.series.Series
```

In [ ]:
```
## Array,List, dict
```

In [7]:
```
!pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\dhruv\appdata\local\programs
\python\python38\lib\site-packages (1.24.2)
```

In [8]:
```
import numpy as np
```

In [9]:
```
np.__version__
```

Out[9]:
```
'1.24.2'
```

In [14]:
```
data = np.array([1,2,3,4,5])
data
```

Out[14]:
```
array([1, 2, 3, 4, 5])
```

In [15]:
```
type(data)
```

Out[15]:
```
numpy.ndarray
```

In [17]:
```
s_array = pd.Series(data)
print(s_array)
```

```
0    1
1    2
2    3
3    4
4    5
dtype: int32
```

In [18]:
```
type(s_array)
```

Out[18]:
```
pandas.core.series.Series
```

In [19]:
```
## List

lst = [2,3,45,6,7]

s_list = pd.Series(lst)
s_list
```

Out[19]:
```
0     2
1     3
2    45
3     6
4     7
dtype: int64
```

In [21]:
```
## Dict

d = {1:"john",2:"Bob"}

s_dict = pd.Series(d)
s_dict
```

Out[21]:
```
1    john
2     Bob
dtype: object
```

In [22]:
```
s_dict.index
```

Out[22]:
```
Int64Index([1, 2], dtype='int64')
```

In [ ]:
```
## DataFrame


#list
#dict
#Series
## Array
## dataframe
```

In [26]:
```
df = pd.DataFrame()
print(df)
```

```
Empty DataFrame
Columns: []
Index: []
```

In [28]:
```
##list to dataframe

lst1 = [10,20,30,40]

df1 = pd.DataFrame(lst1)
df1
```

Out[28]:

	0
0	10
1	20
2	30
3	40

In [29]:
```
lst1 = [10,20,30,40]

df1 = pd.DataFrame(lst1,columns=["Values"])
df1
```

Out[29]:

	Values
0	10
1	20
2	30
3	40

In [30]:
```
df1.index
```

Out[30]:
```
RangeIndex(start=0, stop=4, step=1)
```

In [31]:
```
df1.columns
```

Out[31]:
```
Index(['Values'], dtype='object')
```

In [32]:
```
df1.dtypes
```

Out[32]:
```
Values    int64
dtype: object
```

In [33]:
```
## Convert dict to dataframe
```

In [34]:
```
d = {"name": ("john","Bob","Elon"), "Age":(20,30,40)}
```

In [35]:
```
d
```

Out[35]:
```
{'name': ('john', 'Bob', 'Elon'), 'Age': (20, 30, 40)}
```

In [36]:
```
df2 = pd.DataFrame(d)
```

In [37]:
```
df2
```

Out[37]:

	name	Age
0	john	20
1	Bob	30
2	Elon	40

In [38]:
```
df2.columns
```
Out[38]:
```
Index(['name', 'Age'], dtype='object')
```

In [39]:
```
df2.index
```
Out[39]:
```
RangeIndex(start=0, stop=3, step=1)
```

In [40]:
```
df2 = pd.DataFrame(d,index=["Name1","Name2","Name3"])
```

In [41]:
```
df2
```
Out[41]:

	name	Age
Name1	john	20
Name2	Bob	30
Name3	Elon	40

In [42]:
```
df2.index
```
Out[42]:
```
Index(['Name1', 'Name2', 'Name3'], dtype='object')
```

In [44]:
```
## Series to dataframe

lst10 = [20,40,60,80]

d_series = pd.Series(lst10)
df3 = pd.DataFrame(d_series)
df3
```
Out[44]:

	0
0	20
1	40
2	60
3	80

In [45]:
```
lst10 = [20,40,60,80]

d_series = pd.Series(lst10)
df3 = pd.DataFrame(d_series,columns=["Status"])
df3
```

Out[45]:

	Status
0	20
1	40
2	60
3	80

In [46]:
```
## Array to Dataframe

arr = np.array([20,30,40,60])
df4 = pd.DataFrame(arr)
df4
```

Out[46]:

	0
0	20
1	30
2	40
3	60

In [47]:
```
# Dataframe to ANother dataFrame

df5 = df4
```

In [48]:
```
df5
```

Out[48]:

	0
0	20
1	30
2	40
3	60

In [51]:
```
df4 + df5 + df5 + df5
```

Out[51]:

	0
0	80
1	120
2	160
3	240

In [52]:
```
df4 * df5
```

Out[52]:

	0
0	400
1	900
2	1600
3	3600

In [56]:
```
df4.drop(3)
```

Out[56]:

	0
0	20
1	30
2	40

In [77]:
```
what is the difference between iloc and loc function?
#what is the difference betweenen iat and at function?

dict_ = {
    "A" : [1,2,3],
    "B" : [True,False,True],
    "c" : [2.5,3.5,4.5]

}

df6 = pd.DataFrame(dict_)
df6
```

```
Object `function` not found.
```

Out[77]:

	A	B	c
0	1	True	2.5
1	2	False	3.5
2	3	True	4.5

In [ ]:
```
Iloc --> Index based location

loc ---> Index & columns both can we use loc
```

In [59]:
```
df6
```

Out[59]:

	A	B	c
0	1	True	2.5
1	2	False	3.5
2	3	True	4.5

In [60]:
```
df6.iloc[0]
```

Out[60]:
```
A       1
B    True
c     2.5
Name: 0, dtype: object
```

In [61]:
```
df6.iloc[1]
```

Out[61]:
```
A        2
B    False
c      3.5
Name: 1, dtype: object
```

In [63]:
```
df6.iloc[0: 2]
```

Out[63]:

	A	B	c
0	1	True	2.5
1	2	False	3.5

In [64]:
```
df6.iloc[[0,1]]
```

Out[64]:

	A	B	c
0	1	True	2.5
1	2	False	3.5

In [71]:
```
df6
```

Out[71]:

	A	B	c
0	1	True	2.5
1	2	False	3.5
2	3	True	4.5

In [84]:
```
df6 = pd.DataFrame(dict_,index=["Rank1","Rank2","Rank3"])
df6
```

Out[84]:

	A	B	c
Rank1	1	True	2.5
Rank2	2	False	3.5
Rank3	3	True	4.5

In [85]:
```
df6.loc["Rank1"]
```

Out[85]:
```
A       1
B    True
c     2.5
Name: Rank1, dtype: object
```

In [86]:
```
df6.loc[["Rank1","Rank2"]]
```

Out[86]:

	A	B	c
Rank1	1	True	2.5
Rank2	2	False	3.5

In [87]:
```
df6.loc["Rank1": "Rank2"]
```

Out[87]:

	A	B	c
Rank1	1	True	2.5
Rank2	2	False	3.5

In [82]:
```
df6.loc[0]
```

Out[82]:
```
A       1
B    True
c     2.5
Name: 0, dtype: object
```

In [78]:
```
df6
```

Out[78]:

	A	B	c
0	1	True	2.5
1	2	False	3.5
2	3	True	4.5

In [83]:
```
df6.loc[1]
```

Out[83]:
```
A        3
B     True
c      4.5
Name: 2, dtype: object
```

In [88]:
```
df6.size
```

Out[88]:
```
9
```

In [89]:
```
df6.index
```

Out[89]:
```
Index(['Rank1', 'Rank2', 'Rank3'], dtype='object')
```

In [90]:
```
df6.describe
```

Out[90]:
```
<bound method NDFrame.describe of        A      B    c
Rank1  1   True  2.5
Rank2  2  False  3.5
Rank3  3   True  4.5>
```

In [94]:
```
df6.shape
```

Out[94]:
```
(3, 3)
```

In [95]:
```
df6.empty
```

Out[95]:
```
False
```

In [96]:
```
df6.values
```

Out[96]:
```
array([[1, True, 2.5],
       [2, False, 3.5],
       [3, True, 4.5]], dtype=object)
```

In [ ]:
```
#### Read csv to Dataframe
```

In [97]:

```
# Read the csv file

df_read = pd.read_csv("D:\\Data Analystics Current Batch\\4_April_2024_batch3\
df_read
```

Out[97]:

	Date	Closing price	Return
0	1/1/2020	100	0.010000
1	2/1/2020	120	0.200000
2	3/1/2020	130	0.083333
3	4/1/2020	98	-0.246154
4	5/1/2020	50	-0.489796
5	6/1/2020	102	1.040000
6	7/1/2020	104	0.019608
7	8/1/2020	150	0.442308
8	9/1/2020	160	0.066667
9	10/1/2020	109	-0.318750
10	11/1/2020	95	-0.128440

In [99]:

```
len(df_read)
```

Out[99]:

```
11
```

In [100]:

```
type(df_read)
```

Out[100]:

```
pandas.core.frame.DataFrame
```

In [102]:

```
df_read.head() ## top five records will nbe visiable
```

Out[102]:

	Date	Closing price	Return
0	1/1/2020	100	0.010000
1	2/1/2020	120	0.200000
2	3/1/2020	130	0.083333
3	4/1/2020	98	-0.246154
4	5/1/2020	50	-0.489796

In [104]:
```
df_read.tail() # bottom five records will be visiable
```

Out[104]:

	Date	Closing price	Return
6	7/1/2020	104	0.019608
7	8/1/2020	150	0.442308
8	9/1/2020	160	0.066667
9	10/1/2020	109	-0.318750
10	11/1/2020	95	-0.128440

In [107]:
```
df_read.head(3)
```

Out[107]:

	Date	Closing price	Return
0	1/1/2020	100	0.010000
1	2/1/2020	120	0.200000
2	3/1/2020	130	0.083333

In [ ]: