In [118]:
```
# Pandas : DataFrame
```

In [119]:
```
import pandas as pd
import numpy as np
```

In [120]:
```
score = [[12, 20, 18], [13, 14, 6], [12, 8, 19], [20, 16, 9]]
```

In [121]:
```
df = pd.DataFrame(data=score)
df
```

Out[121]:

	0	1	2
0	12	20	18
1	13	14	6
2	12	8	19
3	20	16	9

In [122]:
```
dars = ['Python', 'C++', 'Java']
name = ['Ali', 'Sara', 'Taha', 'Mahsa']
```

In [123]:
```
df = pd.DataFrame(data=score, index=name, columns=dars)
df
```

Out[123]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	6
Taha	12	8	19
Mahsa	20	16	9

In [124]:
```
d = {'Python': [12, 13, 12, 20], 'C++': [20, 14, 8, 16], 'Java': [18, 6, 19,
student = pd.DataFrame(data=d, index=name)
student
```

Out[124]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	6
Taha	12	8	19
Mahsa	20	16	9

In [125]:
```
df.values
```

Out[125]:
```
array([[12, 20, 18],
       [13, 14,  6],
       [12,  8, 19],
       [20, 16,  9]], dtype=int64)
```

In [126]:
```
df.columns
```

Out[126]:
```
Index(['Python', 'C++', 'Java'], dtype='object')
```

In [127]:
```
'Java' in df.columns
```

Out[127]:
```
True
```

In [128]:
```
df.index
```

Out[128]:
```
Index(['Ali', 'Sara', 'Taha', 'Mahsa'], dtype='object')
```

In [129]:
```
'Sara' in df.index
```

Out[129]:
```
True
```

In [130]:
```
df.axes
```

Out[130]:
```
[Index(['Ali', 'Sara', 'Taha', 'Mahsa'], dtype='object'),
 Index(['Python', 'C++', 'Java'], dtype='object')]
```

In [131]:
```
df.dtypes
```

Out[131]:
```
Python    int64
C++       int64
Java      int64
dtype: object
```

In [132]:
```
### Indexing, Selection, and Filtering
```

In [133]:
```
df
```

Out[133]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	6
Taha	12	8	19
Mahsa	20	16	9

In [134]:
```
df['C++']
```

Out[134]:
```
Ali      20
Sara     14
Taha      8
Mahsa    16
Name: C++, dtype: int64
```

In [135]:
```
df[['Python', 'Java']]
```

Out[135]:

	Python	Java
Ali	12	18
Sara	13	6
Taha	12	19
Mahsa	20	9

In [136]:
```
df
```

Out[136]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	6
Taha	12	8	19
Mahsa	20	16	9

In [137]:
```
df[:3]
```

Out[137]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	6
Taha	12	8	19

In [138]:
```
df < 10
```

Out[138]:

	Python	C++	Java
Ali	False	False	False
Sara	False	False	True
Taha	False	True	False
Mahsa	False	False	True

In [139]:
```
df[df < 10] = 0
df
```

Out[139]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [140]:
```
df[df['C++'] > 10]
```

Out[140]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Mahsa	20	16	0

In [141]:
```
# loc , iloc
```

In [142]:
```
df
```

Out[142]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [143]:
```
df.loc['Taha']
```

Out[143]:
```
Python    12
C++        0
Java      19
Name: Taha, dtype: int64
```

In [144]:
```
df.iloc[2]
```

Out[144]:
```
Python    12
C++        0
Java      19
Name: Taha, dtype: int64
```

In [145]:
```
df.iloc[[2]]
```

Out[145]:

	Python	C++	Java
Taha	12	0	19

In [146]:
```
df.loc['Taha', ['Python','Java']]
```

Out[146]:
```
Python    12
Java      19
Name: Taha, dtype: int64
```

In [147]:
```
df.iloc[2, [0, 2]]
```

Out[147]:
```
Python    12
Java      19
Name: Taha, dtype: int64
```

In [148]:
```
df
```

Out[148]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [149]:
```
df.loc[:'Taha', 'Python']
```

Out[149]:
```
Ali     12
Sara    13
Taha    12
Name: Python, dtype: int64
```

In [150]:
```
df.iloc[:3,0]
```

Out[150]:
```
Ali     12
Sara    13
Taha    12
Name: Python, dtype: int64
```

In [151]:
```
df
```

Out[151]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [152]:
```
df.iloc[:2, [0, 1]]
```

Out[152]:

	Python	C++
Ali	12	20
Sara	13	14

In [153]:
```
df.iloc[0, 1]
```

Out[153]: 20

In [154]:
```
df.iloc[[0, 2]]
```

Out[154]:

	Python	C++	Java
Ali	12	20	18
Taha	12	0	19

In [155]:
```
df
```

Out[155]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [156]:
```
df.iloc[[True, False, True, True]]
```

Out[156]:

	Python	C++	Java
Ali	12	20	18
Taha	12	0	19
Mahsa	20	16	0

In [157]:
```
df.iloc[:, [True, False, True]]
```

Out[157]:

	Python	Java
Ali	12	18
Sara	13	0
Taha	12	19
Mahsa	20	0

In [158]:
```
df.iloc[[True, False, True, True], [True, False, True]]
```

Out[158]:

	Python	Java
Ali	12	18
Taha	12	19
Mahsa	20	0

In [159]:
```
# iat : Access a single value for a row/column pair by integer position.
```

In [160]:
```
df
```

Out[160]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [161]:
```
df.iat[2, 1]
```

Out[161]: 0

In [162]:
```
df.iat[0, 1]
```

Out[162]: 20

reindex

In [163]:
```
df
```

Out[163]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [164]:
```
df.reindex(['Ali', 'Sara', 'Taha', 'Mahsa', 'Omid'], fill_value=0)
```

Out[164]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0
Omid	0	0	0

In [165]:
```
df.reindex(columns=['Python', 'C++', 'Java', 'Pascal'], fill_value=0)
```

Out[165]:

	Python	C++	Java	Pascal
Ali	12	20	18	0
Sara	13	14	0	0
Taha	12	0	19	0
Mahsa	20	16	0	0

In [166]:
```
### sort_values
```

In [167]:
```
df
```

Out[167]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [168]:
```
df.sort_values(by='Python', ascending=False)
```

Out[168]:

	Python	C++	Java
Mahsa	20	16	0
Sara	13	14	0
Ali	12	20	18
Taha	12	0	19

In [169]:
```
df.sort_values(by='Python')
```

Out[169]:

	Python	C++	Java
Ali	12	20	18
Taha	12	0	19
Sara	13	14	0
Mahsa	20	16	0

In [170]:
```
df.sort_values(by=['Python', 'C++'])
```

Out[170]:

	Python	C++	Java
Taha	12	0	19
Ali	12	20	18
Sara	13	14	0
Mahsa	20	16	0

In [171]:
```
### sort_index
```

In [172]:
```
df
```

Out[172]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [173]:
```
df.sort_index()
```

Out[173]:

	Python	C++	Java
Ali	12	20	18
Mahsa	20	16	0
Sara	13	14	0
Taha	12	0	19

In [174]:
```
df.sort_index(axis=1)
```

Out[174]:

	C++	Java	Python
Ali	20	18	12
Sara	14	0	13
Taha	0	19	12
Mahsa	16	0	20

In [175]:
```
df.sort_index(axis=1, ascending=False)
```

Out[175]:

	Python	Java	C++
Ali	12	18	20
Sara	13	0	14
Taha	12	19	0
Mahsa	20	0	16

idxmax

In [176]:
```
df
```

Out[176]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [177]:
```
df.idxmax()
```

Out[177]:
```
Python    Mahsa
C++         Ali
Java       Taha
dtype: object
```

sum() , mean() , describe()

In [178]:
```
df
```

Out[178]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [179]:
```
df.sum()
```

Out[179]:
```
Python    57
C++       50
Java      37
dtype: int64
```

In [180]:
```
df.mean()
```

Out[180]:
```
Python    14.25
C++       12.50
Java       9.25
dtype: float64
```

In [181]:
```
df.sum(axis='columns')
```

Out[181]:
```
Ali      50
Sara     27
Taha     31
Mahsa    36
dtype: int64
```

In [182]:
```
df.mean(axis='columns')
```

Out[182]:
```
Ali      16.666667
Sara      9.000000
Taha     10.333333
Mahsa    12.000000
dtype: float64
```

In [183]:
```
df.mean(axis='columns', skipna=False)
```

Out[183]:
```
Ali      16.666667
Sara      9.000000
Taha     10.333333
Mahsa    12.000000
dtype: float64
```

In [184]:
```
df.describe()
```

Out[184]:

	Python	C++	Java
count	4.00000	4.000000	4.000000
mean	14.25000	12.500000	9.250000
std	3.86221	8.698659	10.688779
min	12.00000	0.000000	0.000000
25%	12.00000	10.500000	0.000000
50%	12.50000	15.000000	9.000000
75%	14.75000	17.000000	18.250000
max	20.00000	20.000000	19.000000

Transpose

In [185]:
```
df
```

Out[185]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [186]:
```
df.T
```

Out[186]:

	Ali	Sara	Taha	Mahsa
Python	12	13	12	20
C++	20	14	0	16
Java	18	0	19	0

apply

In [187]:
```
df
```

Out[187]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [188]:
```
df.apply(lambda x: x-1)
```

Out[188]:

	Python	C++	Java
Ali	11	19	17
Sara	12	13	-1
Taha	11	-1	18
Mahsa	19	15	-1

In [189]:
```
df
```

Out[189]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [190]:
```
def myfunc(x):
    return pd.Series([x.min(), x.max()], index=['min', 'max'])

df.apply(myfunc)
```

Out[190]:

	Python	C++	Java
min	12	0	0
max	20	20	19

In [191]:
```
### map
```

In [192]:
```
df
```

Out[192]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [193]:
```
df['Python'].map(lambda x: x - 1)
```

Out[193]:
```
Ali      11
Sara     12
Taha     11
Mahsa    19
Name: Python, dtype: int64
```

applymap

In [194]:
```
df
```

Out[194]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [195]:
```
df.applymap(lambda x: '%.4f' % x)
```

Out[195]:

	Python	C++	Java
Ali	12.0000	20.0000	18.0000
Sara	13.0000	14.0000	0.0000
Taha	12.0000	0.0000	19.0000
Mahsa	20.0000	16.0000	0.0000

drop

In [196]:
```
df
```

Out[196]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [197]:
```
a = df.drop(['Java'], axis=1)
```

In [198]:
```
a
```

Out[198]:

	Python	C++
Ali	12	20
Sara	13	14
Taha	12	0
Mahsa	20	16

In [199]:
```
df
```

Out[199]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [200]:
```
df.drop(['Sara'])
```

Out[200]:

	Python	C++	Java
Ali	12	20	18
Taha	12	0	19
Mahsa	20	16	0

In [201]:
```
df
```

Out[201]:

	Python	C++	Java
Ali	12	20	18
Sara	13	14	0
Taha	12	0	19
Mahsa	20	16	0

In [202]:
```
df.drop(['Sara'], inplace=True)
```

In [203]:
```
df
```

Out[203]:

	Python	C++	Java
Ali	12	20	18
Taha	12	0	19
Mahsa	20	16	0

empty

In [204]:
```
df.empty
```

Out[204]: False

Arithmetic methods with fill values

In [205]:
```
arr1 = np.arange(12).reshape((4, 3))
arr1
```

Out[205]:
```
array([[ 0,  1,  2],
       [ 3,  4,  5],
       [ 6,  7,  8],
       [ 9, 10, 11]])
```

In [206]:
```
df1 = pd.DataFrame(data=arr1, columns=list('abc'))
df1
```

Out[206]:

	a	b	c
0	0	1	2
1	3	4	5
2	6	7	8
3	9	10	11

In [207]:
```
arr2 = np.arange(10).reshape((5, 2))
df2 = pd.DataFrame(data=arr2, columns=list('ab'))
df2
```

Out[207]:

	a	b
0	0	1
1	2	3
2	4	5
3	6	7
4	8	9

In [208]:
```
df2.loc[1, 'b'] = np.nan
df2
```

Out[208]:

	a	b
0	0	1.0
1	2	NaN
2	4	5.0
3	6	7.0
4	8	9.0

In [209]:
```
df1
```

Out[209]:

	a	b	c
0	0	1	2
1	3	4	5
2	6	7	8
3	9	10	11

In [210]:
```
df2
```

Out[210]:

	a	b
0	0	1.0
1	2	NaN
2	4	5.0
3	6	7.0
4	8	9.0

In [211]:
```
df1 + df2
```

Out[211]:

	a	b	c
0	0.0	2.0	NaN
1	5.0	NaN	NaN
2	10.0	12.0	NaN
3	15.0	17.0	NaN
4	NaN	NaN	NaN

In [212]:
```
df1.add(df2)
```

Out[212]:

	a	b	c
0	0.0	2.0	NaN
1	5.0	NaN	NaN
2	10.0	12.0	NaN
3	15.0	17.0	NaN
4	NaN	NaN	NaN

In [213]:
```
df1.add(df2, fill_value=0)
```

Out[213]:

	a	b	c
0	0.0	2.0	2.0
1	5.0	4.0	5.0
2	10.0	12.0	8.0
3	15.0	17.0	11.0
4	8.0	9.0	NaN

In [214]:
```
df1.sub(df2)
```

Out[214]:

	a	b	c
0	0.0	0.0	NaN
1	1.0	NaN	NaN
2	2.0	2.0	NaN
3	3.0	3.0	NaN
4	NaN	NaN	NaN

In [215]:
```
df1
```

Out[215]:

	a	b	c
0	0	1	2
1	3	4	5
2	6	7	8
3	9	10	11

In [216]:
```
df1.cumsum()
```

Out[216]:

	a	b	c
0	0	1	2
1	3	5	7
2	9	12	15
3	18	22	26

index.name, columns.name

In [217]:
```
mydict = {'City':['Hamedan', 'Hamedan', 'Hamedan', 'Tehran', 'Tehran', 'Tehra
          'Year':[1396, 1397, 1398, 1397, 1398, 1399],
          'Pop': [2, 2.2, 3, 8, 8.5, 9]}

f = pd.DataFrame(data=mydict)
f
```

Out[217]:

	City	Year	Pop
0	Hamedan	1396	2.0
1	Hamedan	1397	2.2
2	Hamedan	1398	3.0
3	Tehran	1397	8.0
4	Tehran	1398	8.5
5	Tehran	1399	9.0

In [218]:
```
d = {'Tehran': {1397: 8, 1398: 8.5, 1399: 9},'Hamedan': {1396: 2, 1397: 2.2,
frame = pd.DataFrame(data=d)
frame
```

Out[218]:

	Tehran	Hamedan
1397	8.0	2.2
1398	8.5	3.0
1399	9.0	NaN
1396	NaN	2.0

In [219]:
```
frame.index.name = 'year'
frame
```

Out[219]:

	Tehran	Hamedan
year		
1397	8.0	2.2
1398	8.5	3.0
1399	9.0	NaN
1396	NaN	2.0

In [220]:
```
frame.columns.name = 'city'
frame
```

Out[220]:

city	Tehran	Hamedan
year		
1397	8.0	2.2
1398	8.5	3.0
1399	9.0	NaN
1396	NaN	2.0

Operations between DataFrame and Series

In [221]:
```
frame
```

Out[221]:

city	Tehran	Hamedan
year		
1397	8.0	2.2
1398	8.5	3.0
1399	9.0	NaN
1396	NaN	2.0

In [222]:
```
myser = frame.iloc[1]
myser
```

Out[222]:
```
city
Tehran     8.5
Hamedan    3.0
Name: 1398, dtype: float64
```

In [223]:
```
frame + myser
```

Out[223]:

city	Tehran	Hamedan
year		
1397	16.5	5.2
1398	17.0	6.0
1399	17.5	NaN
1396	NaN	5.0

Hierarchical Indexing

In [224]:
```
ame({'C++': [6, 7, 4, 16, 14, 13, 15, 7],
     'Python': [3, 2, 5, 14, 19, 18, 16, 10],
     'K1': ['one', 'one', 'one', 'two', 'two', 'two', 'two', 'two'],
     'K2': ['ali', 'reza', 'sara', 'ali', 'reza', 'sara', 'taha', 'farid']})
```

Out[224]:

	C++	Python	K1	K2
0	6	3	one	ali
1	7	2	one	reza
2	4	5	one	sara
3	16	14	two	ali
4	14	19	two	reza
5	13	18	two	sara
6	15	16	two	taha
7	7	10	two	farid

In [225]:
```
df = frame.set_index(['K1', 'K2'])
df
```

Out[225]:

		C++	Python
K1	**K2**		
one	**ali**	6	3
	reza	7	2
	sara	4	5
two	**ali**	16	14
	reza	14	19
	sara	13	18
	taha	15	16
	farid	7	10

In [226]:
```
df.sort_index(level=1)
```

Out[226]:

		C++	Python
K1	**K2**		
one	**ali**	6	3
two	**ali**	16	14
	farid	7	10
one	**reza**	7	2
two	**reza**	14	19
one	**sara**	4	5
two	**sara**	13	18
	taha	15	16

In [227]:
```
df.mean(level='K2')
```

Out[227]:

	C++	Python
K2		
ali	11.0	8.5
reza	10.5	10.5
sara	8.5	11.5
taha	15.0	16.0
farid	7.0	10.0

In [228]:
```
df
```

Out[228]:

		C++	Python
K1	**K2**		
one	**ali**	6	3
	reza	7	2
	sara	4	5
two	**ali**	16	14
	reza	14	19
	sara	13	18
	taha	15	16
	farid	7	10

In [229]:
```
df.unstack()
```

Out[229]:

	C++					Python				
K2	**ali**	**farid**	**reza**	**sara**	**taha**	**ali**	**farid**	**reza**	**sara**	**taha**
K1										
one	6.0	NaN	7.0	4.0	NaN	3.0	NaN	2.0	5.0	NaN
two	16.0	7.0	14.0	13.0	15.0	14.0	10.0	19.0	18.0	16.0

In [230]:
```
df
```

Out[230]:

		C++	Python
K1	**K2**		
one	**ali**	6	3
	reza	7	2
	sara	4	5
two	**ali**	16	14
	reza	14	19
	sara	13	18
	taha	15	16
	farid	7	10

In [231]:
```
df.swaplevel('K1', 'K2')
```

Out[231]:

		C++	Python
K2	**K1**		
ali	**one**	6	3
reza	**one**	7	2
sara	**one**	4	5
ali	**two**	16	14
reza	**two**	14	19
sara	**two**	13	18
taha	**two**	15	16
farid	**two**	7	10

In [232]:
```
##
```

In [233]:
```
d = np.arange(12).reshape((4, 3))
c = [['Ohio', 'Ohio', 'Colorado'], ['Green', 'Red', 'Green']]
i = [['a', 'a', 'b', 'b'], [1, 2, 1, 2]]

frame = pd.DataFrame(data=d,index=i, columns=c)
frame
```

Out[233]:

		Ohio		Colorado
		Green	**Red**	**Green**
a	**1**	0	1	2
	2	3	4	5
b	**1**	6	7	8
	2	9	10	11

In [234]:
```
frame.index.names = ['key1', 'key2']
frame.columns.names = ['state', 'color']
frame
```

Out[234]:

state		Ohio		Colorado
color		Green	Red	Green
key1	key2			
a	1	0	1	2
	2	3	4	5
b	1	6	7	8
	2	9	10	11

دانشگاه شهید مدنی آذربایجان
برنامه نویسی پیشرفته با پایتون
امین گلزاری اسکوئی
۱۴۰۰-۱۴۰۱

Codes and Projects (click here) (https://github.com/Amin-Golzari-Oskouei/Python-Programming-Course-Advanced-2021) slides and videos (click here) (https://drive.google.com/drive/folders/1Dx3v7fD1QBWL-MNP2hd7ilxaRbeALkkA)