

```
In [11]: import numpy as np
import pandas as pd

creating series from scalar values

In [41]: s1=pd.Series([10,20,30])

In [61]: s1
Out[61]:
0    10
1    20
2    30
dtype: int64

In [121]: list1 = [10,20,30]
s1 = pd.Series(list1)

In [141]: s1
Out[141]:
0    10
1    20
2    30
dtype: int64

In [161]: s2=pd.Series(["Raj","Sam","Sia","Kia"], index=[1,2,3,4])

In [181]: s2
Out[181]:
1    Raj
2     Sam
3     Sia
4     Kia
dtype: object

In [201]: s2=pd.Series(["Raj","Sam","Sia","Kia"], index=[3,4,2,1])

In [221]: s2
Out[221]:
3     Raj
4     Sam
2     Sia
1     Kia
dtype: object

In [261]: labels= ['a','b','c','d']
l2= ["Raj","Sam","Sia","Kia"]

In [281]: s3=pd.Series(l2,labels)

In [301]: s3
Out[301]:
a     Raj
b     Sam
c     Sia
d     Kia
dtype: object

In [321]: array=np.array([1,2,3,4])

In [341]: s4=pd.Series(array1)

In [361]: s4
Out[361]:
0     1
1     2
2     3
3     4
dtype: int32

In [401]: s4=pd.Series(array1,index=["Jan","Feb","Mar","Apr"])

In [421]: s4
Out[421]:
Jan     1
Feb     2
Mar     3
Apr     4
dtype: int32

In [441]: #indexing

In [461]: s1
Out[461]:
0     10
1     20
2     30
dtype: int64

In [481]: s1[1]
Out[481]: 20

In [501]: s1[0]
Out[501]: 10

In [521]: s1[2]
Out[521]: 30

In [561]: s5=pd.Series(['New Delhi','WashingtonDC', 'London','Paris'],index=['India','USA','UK','France'])

In [581]: s5
Out[581]:
India      New Delhi
USA        WashingtonDC
UK          London
France      Paris
dtype: object

In [601]: s5[1]
C:\Users\student\AppData\Local\Temp\ipykernel_3036\179420861.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as i
bels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
s5[1]
Out[601]: 'WashingtonDC'

In [621]: s5[2]
C:\Users\student\AppData\Local\Temp\ipykernel_3036\179420861.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as i
bels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
s5[2]
Out[621]: 'London'

In [641]: s5[3]
C:\Users\student\AppData\Local\Temp\ipykernel_3036\1598816226.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as i
bels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
s5[3]
Out[641]: 'Paris'

In [661]: s5[0]
C:\Users\student\AppData\Local\Temp\ipykernel_3036\4103576207.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as i
bels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
s5[0]
Out[661]: 'New Delhi'

In [741]: s5[[1,2]]
C:\Users\student\AppData\Local\Temp\ipykernel_3036\3520924411.py:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as i
bels (consistent with DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
s5[[1,2]]
Out[741]:
USA      WashingtonDC
UK        London
dtype: object

In [761]: s5[['UK','USA']]
Out[761]:
UK      London
USA    WashingtonDC
dtype: object

In [781]: s5.index[10,20,30,40]

In [801]: s5
Out[801]:
10      New Delhi
20    WashingtonDC
30      London
40      Paris
dtype: object

In [821]: #joining

In [841]: s5
Out[841]:
10      New Delhi
20    WashingtonDC
30      London
40      Paris
dtype: object

In [861]: s6=pd.Series(['New Delhi','WashingtonDC', 'London','Paris'],index=['India','USA','UK','France'])

In [881]: s6
Out[881]:
India      New Delhi
USA        WashingtonDC
UK          London
France      Paris
dtype: object

In [1001]: s6[5:8]
Out[1001]: Series([], dtype: object)

In [1021]: s6["USA"|"France"]
Out[1021]:
USA      WashingtonDC
UK        London
France    Paris
dtype: object

In [1201]: s6[:4]
Out[1201]:
India      New Delhi
USA        WashingtonDC
UK          London
France    Paris
dtype: object

In [1221]: s6[-1:]
Out[1221]:
France    Paris
dtype: object

In [1241]: s6[:-1]
Out[1241]:
India      New Delhi
USA        WashingtonDC
UK          London
dtype: object

In [1281]: s6[:-2]
Out[1281]:
India      New Delhi
USA        WashingtonDC
dtype: object

In [1301]: s6[: :-1]
Out[1301]:
France    Paris
UK          London
USA        WashingtonDC
India      New Delhi
dtype: object

In [1321]: s7=pd.Series(np.arange(10,16,1),index=['a','b','c','d','e','f'])

In [1341]: s7
Out[1341]:
a     10
b     11
c     12
d     13
e     14
f     15
dtype: int32

In [1361]: s7[1:3]=50

In [1381]: s7
Out[1381]:
a     10
b     50
c     50
d     13
e     14
f     15
dtype: int32

In [1401]: s7['c':'e']=100

In [1421]: s7
Out[1421]:
a     10
b     50
c    100
d    100
e    100
f     15
dtype: int32

In [1441]: #attributes

In [1461]: s6
Out[1461]:
India      New Delhi
USA        WashingtonDC
UK          London
France    Paris
dtype: object

In [1501]: s6.name="capitals"

In [1521]: s6
Out[1521]:
India      New Delhi
USA        WashingtonDC
UK          London
France    Paris
Name: capitals, dtype: object

In [1541]: s6.index.name="Countries"

In [1561]: s6
Out[1561]:
Countries
India      New Delhi
USA        WashingtonDC
UK          London
France    Paris
Name: capitals, dtype: object

In [1581]: print(s6.values)
['New Delhi' 'WashingtonDC' 'London' 'Paris']

In [1601]: print(s6.size)
4

In [1621]: s6.empty
Out[1621]: False

In [1641]: s7
Out[1641]:
a     10
b     50
c    100
d    100
e    100
f     15
dtype: int32

In [1661]: s8=pd.Series(np.arange(10,16,1))

In [1721]: s8
Out[1721]:
0     10
1     11
2     12
3     13
4     14
5     15
dtype: int32

In [1741]: s8.head(2)
Out[1741]:
0     10
1     11
dtype: int32

In [1761]: s8.tail(3)
Out[1761]:
3     13
4     14
5     15
dtype: int32

In [1781]: s8.count()
Out[1781]: 6

In [1901]: sA=pd.Series(np.arange(1,7,1),index=['a','b','c','d','e','f'])

In [1921]: sA
Out[1921]:
a     1
b     2
c     3
d     4
e     5
f     6
dtype: int32

In [2001]: sA=pd.Series([10,20,-10,-50,100,120],index=['z','y','c','d','x','f'])

In [2021]: sB
Out[2021]:
z     10
y     20
c    -10
d    -50
x    100
f    120
dtype: int64

In [2041]: sA+sB
Out[2041]:
a     NaN
b     NaN
c    -7.0
d   -46.0
e     NaN
f    126.0
x     NaN
y     NaN
z     NaN
dtype: float64

In [2061]: sA.add(sB,fill_value=0)
Out[2061]:
a     1.0
b     2.0
c    -7.0
d   -46.0
e     5.0
f    126.0
x    100.0
y    20.0
z     10.0
dtype: float64

In [2081]: sA-sB
Out[2081]:
a     NaN
b     NaN
c    13.0
d    54.0
e     NaN
f   -114.0
x     NaN
y     NaN
z     NaN
dtype: float64

In [2101]: sA.sub(sB,fill_value=100)
Out[2101]:
a    -99.0
b   -98.0
c    13.0
d    54.0
e   -95.0
f   -114.0
x     0.0
y     80.0
z     90.0
dtype: float64

In [2121]: sA*sB
Out[2121]:
a     NaN
b     NaN
c   -30.0
d -200.0
e     NaN
f    720.0
x     NaN
y     NaN
z     NaN
dtype: float64

In [2141]: sA.mul(sB,fill_value=0)
Out[2141]:
a     0.0
b     0.0
c   -30.0
d -200.0
e     0.0
f    720.0
x     0.0
y     0.0
z     0.0
dtype: float64

In [2161]: sA/sB
Out[2161]:
a     NaN
b     NaN
c   -0.30
d   -0.08
e     NaN
f    0.05
x     NaN
y     NaN
z     NaN
dtype: float64

In [2201]: sA.div(sB,fill_value=10)
Out[2201]:
a     0.10
b     0.20
c   -0.30
d   -0.08
e    0.50
f    0.05
x     0.10
y     0.50
```

