```
#https://pandas.pydata.org/
import numpy as np
import pandas as pd
```

```
s=pd.Series([0,3,5,np.nan,6,8]) #capital S
s
```

```
0    0.0
1    3.0
2    5.0
3    NaN
4    6.0
5    8.0
dtype: float64
```

```
s.values
```

```
array([ 0,  1,  1,  2,  3,  5,  8, 13])
```

```
s.index
```

```
RangeIndex(start=0, stop=8, step=1)
```

```
for ele in s :
    print(ele)
```

```
0.0
3.0
5.0
nan
6.0
8.0
```

```
for ele in s.index :
    print(ele)
```

```
0
1
2
3
4
5
```

```
s.dtypes
```

```
dtype('float64')
```

```
list(zip(s.index,s.values)) #not value , its values
```

```
[(0, 0.0), (1, 3.0), (2, 5.0), (3, nan), (4, 6.0), (5, 8.0)]
```

```
1 zip(s.index,s.values)
```

```
<zip at 0x7fee2d42e960>
```

```
1 for item in ((zip(s.index,s.values))):
2     print(item)
```

```
(0, 0.0)
(1, 3.0)
(2, 5.0)
(3, nan)
(4, 6.0)
(5, 8.0)
```

```
1 s[3]
```

```
nan
```

```
1 s[3:]
```

```
3    NaN
4    6.0
5    8.0
dtype: float64
```

```
1 s[3:].values
```

```
array([nan,  6.,  8.])
```

```
1 mercury = pd.Series([0.3,57,4222],index=['mass','diameter','daylength'])
2 mercury
```

```
mass           0.3
diameter      57.0
daylength   4222.0
dtype: float64
```

```
1 mercury.mass
```

```
0.3
```

```
1 mercury['mass']
```

```
0.3
```

```
1 mercury['mass':]
```

```
mass           0.3
diameter      57.0
daylength   4222.0
dtype: float64
```

```
1 mercury['mass':'daylength'] #note last value is also included unlike numpy and python
```

```
mass           0.3
diameter      57.0
daylength   4222.0
dtype: float64
```

```
1 mercury[0] #number index also works
```

```
0.3
```

```
1 mercury[0:3] #3rd value also works
```

```
mass           0.3
diameter      57.0
daylength   4222.0
dtype: float64
```

```
1 mercury[0:2]
```

```
mass         0.3
diameter    57.0
dtype: float64
```

```
1 mercury[0:1]
```

```
mass    0.3
dtype: float64
```

```
1 #loc vs iloc
```

```
1 mercury.loc[0]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/indexes/base.py in get_loc(self, key, method, tolerance)
   3360             try:
-> 3361                 return self._engine.get_loc(casted_key)
   3362             except KeyError as err:
```

7 frames

```
1 mercury.loc['mass']
```

```
0.3
```

```
1 mercury.iloc[0]
```

```
0.3
```

```
1 mercury.iloc['mass']
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-46-dfd412a79964> in <module>()
----> 1 mercury.iloc['mass']
```

1 frames

```
/usr/local/lib/python3.7/dist-packages/pandas/core/indexing.py in _getitem_axis(self, key, axis)
   1561             key = item_from_zerodim(key)
   1562             if not is_integer(key):
-> 1563                 raise TypeError("Cannot index by location index with a non-integer key")
   1564
   1565             # validate the location

TypeError: Cannot index by location index with a non-integer key
```

```
1 #np functions operations can be worked here also
```

```
1 #can create arrays using numpy range , arange , random
```

```
1 #other ways of creating a series object
```

```
1 arr=np.random.randint(30,40,10)
2 arr
```

```
array([39, 34, 33, 38, 38, 36, 34, 32, 31, 33])
```

```
1  arrpd=pd.Series(arr)
2  arrpd
```

```
0    39
```

```
1    34
2    33
3    38
4    38
5    36
6    34
7    32
8    31
9    33
dtype: int64
```

```
1 ind=np.arange(10,20)
2 ind
```

```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19])
```

```
1   sr3=pd.Series(arr,index=ind)
2   sr3
```

```
10    39
11    34
12    33
13    38
14    38
15    36
16    34
17    32
18    31
19    33
dtype: int64
```

```
1 arr=pd.Series(np.random.randint(30,40,(3,3)))#2d array not possible
2 arr
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-56-1a1b6f295733> in <module>()
----> 1 arr=pd.Series(np.random.randint(30,40,(3,3)))#2d array not possible
      2 arr

2 frames
/usr/local/lib/python3.7/dist-packages/pandas/core/construction.py in
_sanitize_ndim(result, data, dtype, index, allow_2d)
    625             if allow_2d:
    626                 return result
--> 627             raise ValueError("Data must be 1-dimensional")
    628         if is_object_dtype(dtype) and isinstance(dtype, ExtensionDtype):
    629             # i.e. PandasDtype("O")

ValueError: Data must be 1-dimensional
```

SEARCH STACK OVERFLOW

```
1 d1={}
2 d1['mass']=0.33
3 d1['diameter']=57
```

```
4 d1['daylength']=4222
5 d1
```

```
{'daylength': 4222, 'diameter': 57, 'mass': 0.33}
```

```
1 d1={'daylength': 4222, 'diameter': 57, 'mass': 0.33}
2 d1
```

```
{'daylength': 4222, 'diameter': 57, 'mass': 0.33}
```

```
1 sr4=pd.Series({'daylength': 4222, 'diameter': 57, 'mass': 0.33})
2 sr4
```

```
daylength    4222.00
diameter       57.00
mass            0.33
dtype: float64
```

```
1 Mass = pd.Series([0.3,4,5,0.6,1000,560,80,102,0.01],index=['merc','ven','eart','mar','
2 Mass
```

```
merc       0.30
ven        4.00
eart       5.00
mar        0.60
jup     1000.00
sat      560.00
ura       80.00
nept     102.00
plu        0.01
dtype: float64
```

```
1 Mass = pd.Series(0.3,4,5,0.6,1000,560,80,102,0.01),index=['merc','ven','eart','mar','j
```

```
  File "<ipython-input-85-74c895585f4b>", line 1
    Mass = pd.Series(0.3,4,5,0.6,1000,560,80,102,0.01),index=
['merc','ven','eart','mar','jup','sat','ura','nept','plu']
           ^
SyntaxError: can't assign to function call
```

SEARCH STACK OVERFLOW

```
1 Mass>100
```

```
merc    False
ven     False
eart    False
mar     False
jup      True
sat      True
ura     False
nept     True
plu     False
dtype: bool
```

```
1 Mass[Mass>100]
```

```
jup     1000.0
sat      560.0
nept     102.0
dtype: float64
```

```
1 Mass[(Mass>100) and (Mass <600)] # error for and , Use &
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-92-53219a62d871> in <module>()
----> 1 Mass[(Mass>100) and (Mass <600)] # error

/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in __nonzero__(self)
   1536     def __nonzero__(self):
   1537         raise ValueError(
-> 1538             f"The truth value of a {type(self).__name__} is ambiguous. "
   1539             "Use a.empty, a.bool(), a.item(), a.any() or a.all()."
   1540         )

ValueError: The truth value of a Series is ambiguous. Use a.empty, a.bool(),
a.item(), a.any() or a.all().
```

SEARCH STACK OVERFLOW

```
1 Mass[(Mass>100) & (Mass <600)]
```

```
sat     560.0
nept    102.0
dtype: float64
```

```
1 Mass*100
```

```
merc        30.0
ven        400.0
eart       500.0
mar         60.0
jup     100000.0
sat      56000.0
ura       8000.0
nept     10200.0
plu          1.0
dtype: float64
```

```
1 Mass**2
```

```
merc          0.0900
ven          16.0000
eart         25.0000
mar           0.3600
jup     1000000.0000
sat      313600.0000
ura        6400.0000
nept      10404.0000
```

```
plu           0.0001
dtype: float64
```

```
1 np.mean(Mass) #numpy works on pandas series object also
```

```
194.65666666666667
```

```
1 np.mean(Mass,axis=0)
```

```
194.65666666666667
```

```
1   np.mean(Mass,axis=1)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in
_get_axis_number(cls, axis)
    545         try:
--> 546             return cls._AXIS_TO_AXIS_NUMBER[axis]
    547         except KeyError:

KeyError: 1

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
                                          6 frames
<__array_function__ internals> in mean(*args, **kwargs)

/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in
_get_axis_number(cls, axis)
    546             return cls._AXIS_TO_AXIS_NUMBER[axis]
    547         except KeyError:
--> 548             raise ValueError(f"No axis named {axis} for object type
{cls.__name__}")
    549
    550     @final

ValueError: No axis named 1 for object type Series
```

```
1 np.median(Mass)
```

```
5.0
```

```
1 np.median(Mass,axis=0)
```

```
5.0
```

```
1   np.median(Mass,axis=1)
```

```
---------------------------------------------------------------------------
AxisError                                 Traceback (most recent call last)
<ipython-input-100-75756a6f4ab4> in <module>()
----> 1 np.median(Mass,axis=1)

<__array_function__ internals> in median(*args, **kwargs)

                                  3 frames
/usr/local/lib/python3.7/dist-packages/numpy/core/numeric.py in <listcomp>(.0)
   1383             pass
   1384     # Going via an iterator directly is slower than via list comprehension.
-> 1385     axis = tuple([normalize_axis_index(ax, ndim, argname) for ax in axis])
   1386     if not allow_duplicate and len(set(axis)) != len(axis):
   1387         if argname:
```

```
1 np.amin(Mass)
```

```
0.01
```

```
1 np.amin(Mass,axis=0)
```

```
0.01
```

```
1   np.amin(Mass,axis=1)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in
_get_axis_number(cls, axis)
    545         try:
--> 546             return cls._AXIS_TO_AXIS_NUMBER[axis]
    547         except KeyError:

KeyError: 1

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
                                  7 frames
<__array_function__ internals> in amin(*args, **kwargs)

/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in
_get_axis_number(cls, axis)
    546             return cls._AXIS_TO_AXIS_NUMBER[axis]
    547         except KeyError:
--> 548             raise ValueError(f"No axis named {axis} for object type
{cls.__name__}")
    549
    550     @final

ValueError: No axis named 1 for object type Series
```

```
1   np.argmax(Mass)
```

```
4
```