```
import pandas as pd
```

making a series

```
a=pd.Series([1,2,3,4,5], index=["A","B","C","D","E"])
a
```

```
A    1
B    2
C    3
D    4
E    5
dtype: int64
```

Making a data frame

```
b=pd.DataFrame({"manahil": 18, "sara": 21, "faris": 16}, index=["A","B","C"])
b
```

	manahil	sara	faris
A	18	21	16
B	18	21	16
C	18	21	16

Working on DataSet from Seaborn library

```
import seaborn as sns
```

```
df=sns.load_dataset("titanic")
df
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_
0	0	3	male	22.0	1	0	7.2500	S	Third	man	
1	1	1	female	38.0	1	0	71.2833	C	First	woman	F
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	F
3	1	1	female	35.0	1	0	53.1000	S	First	woman	F
4	0	3	male	35.0	0	0	8.0500	S	Third	man	
...	...	...	...	...	...	...	...	...	...	...	
886	0	2	male	27.0	0	0	13.0000	S	Second	man	
887	1	1	female	19.0	0	0	30.0000	S	First	woman	F
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	F
889	1	1	male	26.0	0	0	30.0000	C	First	man	
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	

891 rows × 15 columns

Checking information about data

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
```

```
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    category
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

Checking first five entries

```
df.head()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True

Checking last 5 entries

```
df.tail()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_ma
886	0	2	male	27.0	0	0	13.00	S	Second	man	Tr
887	1	1	female	19.0	0	0	30.00	S	First	woman	Fal
888	0	3	female	NaN	1	2	23.45	S	Third	woman	Fal
889	1	1	male	26.0	0	0	30.00	C	First	man	Tr
890	0	3	male	32.0	0	0	7.75	Q	Third	man	Tr

Summary Statistics

```
df.describe()
```

	survived	pclass	age	sibsp	parch	fare
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Checking number of rows and columns

```
df.shape[0]
```

```
891
```

```
df.shape[1]
```

```
15
```

```
df.shape
```

```
(891, 15)
```

```
name="the number of rows are", df.shape[0]
print(name)
```

```
('the number of rows are', 891)
```

Checking columns name

```
df.columns
```

```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck', 'embark_town',
       'alive', 'alone'],
      dtype='object')
```

checking row headings

```
df.index
```

```
RangeIndex(start=0, stop=891, step=1)
```

Removing Specific Columns

```
df1=df.drop(["deck","alone"], axis=1)
df1
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_
0	0	3	male	22.0	1	0	7.2500	S	Third	man	

Checking missing values

```
df.isnull().sum()
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

Checking unique values

```
df.age.unique()
```

```
array([22.  , 38.  , 26.  , 35.  ,   nan, 54.  ,  2.  , 27.  , 14.  ,
        4.  , 58.  , 20.  , 39.  , 55.  , 31.  , 34.  , 15.  , 28.  ,
        8.  , 19.  , 40.  , 66.  , 42.  , 21.  , 18.  ,  3.  ,  7.  ,
       49.  , 29.  , 65.  , 28.5 ,  5.  , 11.  , 45.  , 17.  , 32.  ,
       16.  , 25.  ,  0.83, 30.  , 33.  , 23.  , 24.  , 46.  , 59.  ,
       71.  , 37.  , 47.  , 14.5 , 70.5 , 32.5 , 12.  ,  9.  , 36.5 ,
       51.  , 55.5 , 40.5 , 44.  ,  1.  , 61.  , 56.  , 50.  , 36.  ,
       45.5 , 20.5 , 62.  , 41.  , 52.  , 63.  , 23.5 ,  0.92, 43.  ,
       60.  , 10.  , 64.  , 13.  , 48.  ,  0.75, 53.  , 57.  , 80.  ,
       70.  , 24.5 ,  6.  ,  0.67, 30.5 ,  0.42, 34.5 , 74.  ])
```

Colab paid products - Cancel contracts here