Importing

```
import pandas as pd
```

Series

Creating Series

```
mydata1 = [56,78,12,34,55]
s1 = pd.Series(mydata)
print(s1)
```

```
0    56
1    78
2    12
3    34
4    55
dtype: int64
```

Creating Series with Custom Index

```
mydata2 = [56,78,12,34,55]
myindex = ['A', 'B','C', 'D', 'E']
s2 = pd.Series(mydata2, index = myindex)
print(s2)
```

```
A    56
B    78
C    12
D    34
E    55
dtype: int64
```

Access

```
print(s1[4])
print(s2["C"])
```

```
55
12
```

Data frames

Creating Data frame from dictionary

```
dict1 = { 'Name' : ['Ananya' , 'Navya' , 'Aishvika'],
          'Age'  : [19, 21, 20],
          'City' : ["Bangalore", "Shimoga", "Dandeli"]
        }

df_dict = pd.DataFrame(dict1)

print(df_dict)
```

```
       Name  Age       City
0    Ananya   19  Bangalore
1     Navya   21    Shimoga
2  Aishvika   20    Dandeli
```

Saving thi data in a csv file

```
df_dict.to_csv("dictdata.csv", index = False)
```

Loading data

```
diab = pd.read_csv(r"D:\Datasets\Csv_excel_txt\diabetcsvsmall.csv")
print(diab.head(7))
```

```
   preg  plas  pres  skin  insu  mass   pedi  age            class
0   6.0   148  72.0  35.0     0  33.6  0.627   50  tested_positive
1   1.0    85  66.0  29.0     0  26.6  0.351   31  tested_negative
2   8.0   183  64.0   0.0     0  23.3  0.672   32  tested_positive
3   1.0    89  66.0  23.0    94  28.1  0.167   21  tested_negative
4   0.0   137  40.0  35.0   168  43.1  2.288   33  tested_positive
5   5.0   116  74.0   0.0     0  25.6  0.201   30  tested_negative
6   3.0    78  50.0  32.0    88  31.0  0.248   26  tested_positive
```

Clean the Data

```
diab.isnull().sum()
```

```
preg     1
plas     0
pres     1
skin     1
insu     0
mass     1
pedi     1
age      0
class    0
dtype: int64
```

```
diab.dropna(inplace= True)

diab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 98 entries, 0 to 101
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   preg    98 non-null     float64
 1   plas    98 non-null     int64  
 2   pres    98 non-null     float64
 3   skin    98 non-null     float64
 4   insu    98 non-null     int64  
 5   mass    98 non-null     float64
 6   pedi    98 non-null     float64
 7   age     98 non-null     int64  
 8   class   98 non-null     object 
dtypes: float64(5), int64(3), object(1)
memory usage: 7.7+ KB
```

```
diab.drop_duplicates(inplace = True)
```

```
diab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 96 entries, 0 to 101
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   preg    96 non-null     float64
 1   plas    96 non-null     int64  
 2   pres    96 non-null     float64
 3   skin    96 non-null     float64
 4   insu    96 non-null     int64  
 5   mass    96 non-null     float64
 6   pedi    96 non-null     float64
 7   age     96 non-null     int64  
 8   class   96 non-null     object 
dtypes: float64(5), int64(3), object(1)
memory usage: 7.5+ KB
```

Modify data

```
diab['age'] = diab['age'].astype(float)
```

```
filtered = diab[diab['age']>30]
```

```
filtered.head()
```

```
   preg  plas  pres  skin  insu  mass   pedi   age            class
0   6.0   148  72.0  35.0     0  33.6  0.627  50.0  tested_positive
1   1.0    85  66.0  29.0     0  26.6  0.351  31.0  tested_negative
2   8.0   183  64.0   0.0     0  23.3  0.672  32.0  tested_positive
```

```
4   0.0   137  40.0  35.0   168  43.1  2.288  33.0  tested_positive
8   2.0   197  70.0  45.0   543  30.5  0.158  53.0  tested_positive
```

```
sorted_df = diab.sort_values(by= 'age', ascending = False)
print(sorted_df.head(10))
```

```
    preg  plas   pres  skin  insu  mass   pedi   age            class
93   4.0   134   72.0   0.0     0  23.8  0.277  60.0  tested_positive
30   5.0   109   75.0  26.0     0  36.0  0.546  60.0  tested_negative
13   1.0   189   60.0  23.0   846  30.1  0.398  59.0  tested_positive
53   8.0   176   90.0  34.0   300  33.7  0.467  58.0  tested_positive
28  13.0   145   82.0  19.0   110  22.2  0.245  57.0  tested_negative
12  10.0   139   80.0   0.0     0  27.1  1.441  57.0  tested_negative
39   4.0   111   72.0  47.0   207  37.1  1.390  56.0  tested_positive
67   2.0   109   92.0   0.0     0  42.7  0.845  54.0  tested_negative
43   9.0   171  110.0  24.0   240  45.4  0.721  54.0  tested_positive
9    8.0   125   96.0   0.0     0   0.0  0.232  54.0  tested_positive
```

```
#Groupby class and calculate average age
grouped = diab.groupby('class')['age'].mean()
print(grouped)
```

```
class
tested_negative    31.508197
tested_positive    40.371429
Name: age, dtype: float64
```

```
#Groupby age and calculate average insu
grouped1 = diab.groupby('age')['insu'].mean()
print(grouped1)
```

```
age
21.0     39.500000
22.0     38.000000
23.0    167.000000
24.0     32.000000
25.0     19.000000
26.0     67.500000
27.0    115.000000
28.0    111.666667
29.0      0.000000
30.0      7.666667
31.0     68.000000
32.0     24.000000
33.0    110.750000
34.0     88.000000
35.0      0.000000
36.0     35.500000
37.0      0.000000
38.0      0.000000
39.0      0.000000
```

```
40.0    114.000000
41.0    104.750000
42.0     97.500000
43.0     55.000000
44.0      0.000000
45.0      0.000000
46.0      0.000000
48.0      0.000000
50.0      0.000000
51.0    160.500000
53.0    543.000000
54.0     80.000000
56.0    207.000000
57.0     55.000000
58.0    300.000000
59.0    846.000000
60.0      0.000000
Name: insu, dtype: float64
```