In [2]:

```
import pandas as pd
import numpy as np
```

In [3]:

```
df=pd.read_excel("StudentsPerformanceTest1.xlsx")
```

In [4]:

```
df
```

Out[4]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72.0	72.0	74.0	78.0	1	Pune
1	female	69.0	90.0	88.0	NaN	2	NaN
2	female	90.0	95.0	93.0	74.0	2	Nashik
3	male	47.0	57.0	NaN	78.0	1	NaN
4	male	NaN	78.0	75.0	81.0	3	Pune
5	female	71.0	NaN	78.0	70.0	4	NaN
6	male	12.0	44.0	52.0	12.0	2	Nashik
7	male	NaN	65.0	67.0	49.0	1	Pune
8	male	5.0	77.0	89.0	55.0	0	NaN

In [5]:

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

Out[5]:

```
gender                  0
math score              2
reading score           1
writing score           1
Placement Score         1
placement offer count   0
Region                  4
dtype: int64
```

In [6]:

```
df.isna().sum()
```

Out[6]:

```
gender                  0
math score              2
reading score           1
writing score           1
Placement Score         1
placement offer count   0
Region                  4
dtype: int64
```

In [7]:

```
df.mean()
```

```
/tmp/ipykernel_17187/3698961737.py:1: FutureWarning: The default value
of numeric_only in DataFrame.mean is deprecated. In a future version,
it will default to False. In addition, specifying 'numeric_only=None'
is deprecated. Select only valid columns or specify the value of numer
ic_only to silence this warning.
  df.mean()
```

Out[7]:

```
math score              52.285714
reading score           72.250000
writing score           77.000000
Placement Score         62.125000
placement offer count    1.777778
dtype: float64
```

In [8]:

```
df['math score'].fillna(52.285714,axis=0,inplace=True)
```

In [9]:

```
df['math score'][4]=52.285714
```

```
/tmp/ipykernel_17187/2649418978.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas
-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
 (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.htm
l#returning-a-view-versus-a-copy)
  df['math score'][4]=52.285714
```

In [10]:

```
df
```

Out[10]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72.000000	72.0	74.0	78.0	1	Pune
1	female	69.000000	90.0	88.0	NaN	2	NaN
2	female	90.000000	95.0	93.0	74.0	2	Nashik
3	male	47.000000	57.0	NaN	78.0	1	NaN
4	male	52.285714	78.0	75.0	81.0	3	Pune
5	female	71.000000	NaN	78.0	70.0	4	NaN
6	male	12.000000	44.0	52.0	12.0	2	Nashik
7	male	52.285714	65.0	67.0	49.0	1	Pune
8	male	5.000000	77.0	89.0	55.0	0	NaN

In [11]:

```
df['reading score'].fillna(72.25,axis=0,inplace=True)
```

In [12]:

```
df
```

Out[12]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72.000000	72.00	74.0	78.0	1	Pune
1	female	69.000000	90.00	88.0	NaN	2	NaN
2	female	90.000000	95.00	93.0	74.0	2	Nashik
3	male	47.000000	57.00	NaN	78.0	1	NaN
4	male	52.285714	78.00	75.0	81.0	3	Pune
5	female	71.000000	72.25	78.0	70.0	4	NaN
6	male	12.000000	44.00	52.0	12.0	2	Nashik
7	male	52.285714	65.00	67.0	49.0	1	Pune
8	male	5.000000	77.00	89.0	55.0	0	NaN

In [13]:

```
df['writing score'].fillna(77.0,axis=0,inplace=True)
```

In [14]:

```
df
```

Out[14]:

	gender	math score	reading score	writing score	Placement Score	placement offer count	Region
0	female	72.000000	72.00	74.0	78.0	1	Pune
1	female	69.000000	90.00	88.0	NaN	2	NaN
2	female	90.000000	95.00	93.0	74.0	2	Nashik
3	male	47.000000	57.00	77.0	78.0	1	NaN
4	male	52.285714	78.00	75.0	81.0	3	Pune
5	female	71.000000	72.25	78.0	70.0	4	NaN
6	male	12.000000	44.00	52.0	12.0	2	Nashik
7	male	52.285714	65.00	67.0	49.0	1	Pune
8	male	5.000000	77.00	89.0	55.0	0	NaN

In [15]:

```
df['Placement Score'].fillna(62.125,axis=0,inplace=True)
```

In [16]:

```
df.drop('Region',axis=1,inplace=True)
```

In [17]:

```
df2=df.drop('gender',axis=1)
```

In [19]:

```
df2
```

Out[19]:

	math score	reading score	writing score	Placement Score	placement offer count
0	72.000000	72.00	74.0	78.000	1
1	69.000000	90.00	88.0	62.125	2
2	90.000000	95.00	93.0	74.000	2
3	47.000000	57.00	77.0	78.000	1
4	52.285714	78.00	75.0	81.000	3
5	71.000000	72.25	78.0	70.000	4
6	12.000000	44.00	52.0	12.000	2
7	52.285714	65.00	67.0	49.000	1
8	5.000000	77.00	89.0	55.000	0

In [20]:

```
df2.to_csv("raw_data.csv",index=False)
```

In [ ]:

```

```