```
import pandas as pd
import numpy as np


df=pd.read_csv("/content/Heart.csv")


shape=df.shape
print("The shape of Dataset is:",shape)
```

```
The shape of Dataset is: (303, 15)
```

```
Missing_values=df.isnull()
print("Missing Values are:",Missing_values)
```

```
Missing Values are:      Unnamed: 0    Age    Sex  ChestPain  RestBP   Chol    Fbs  RestECG  \
0         False  False  False      False   False  False  False    False
1         False  False  False      False   False  False  False    False
2         False  False  False      False   False  False  False    False
3         False  False  False      False   False  False  False    False
4         False  False  False      False   False  False  False    False
..          ...    ...    ...        ...     ...    ...    ...      ...
298       False  False  False      False   False  False  False    False
299       False  False  False      False   False  False  False    False
300       False  False  False      False   False  False  False    False
301       False  False  False      False   False  False  False    False
302       False  False  False      False   False  False  False    False

     MaxHR  ExAng  Oldpeak  Slope     Ca   Thal    AHD
0    False  False    False  False  False  False  False
1    False  False    False  False  False  False  False
2    False  False    False  False  False  False  False
3    False  False    False  False  False  False  False
4    False  False    False  False  False  False  False
..     ...    ...      ...    ...    ...    ...    ...
298  False  False    False  False  False  False  False
299  False  False    False  False  False  False  False
300  False  False    False  False  False  False  False
301  False  False    False  False  False  False  False
302  False  False    False  False   True  False  False

[303 rows x 15 columns]
```

```
total_missing_values=Missing_values.sum()
print("Sum of Total Values:\n",total_missing_values)
```

```
Sum of Total Values is:
 Unnamed: 0    0
Age            0
Sex            0
ChestPain      0
RestBP         0
Chol           0
Fbs            0
RestECG        0
MaxHR          0
ExAng          0
Oldpeak        0
Slope          0
Ca             4
Thal           2
AHD            0
dtype: int64
```

```
data_type=df.dtypes
print("Data Types:\n",data_type)
```

```
Data Types:
 Unnamed: 0      int64
Age              int64
Sex              int64
ChestPain       object
RestBP           int64
Chol             int64
Fbs              int64
RestECG          int64
MaxHR            int64
ExAng            int64
Oldpeak        float64
Slope            int64
Ca             float64
Thal            object
AHD             object
dtype: object
```

```
zero_count=(df==0).sum()
print("Zero Count:\n",zero_count)
```

```
Zero Count:
 Unnamed: 0      0
Age              0
Sex             97
ChestPain        0
RestBP           0
Chol             0
Fbs            258
RestECG        151
MaxHR            0
ExAng          204
Oldpeak         99
Slope            0
Ca             176
Thal             0
AHD              0
dtype: int64
```

```
mean_age= df['Age'].mean()
print("Mean of Age is:\t",mean_age)
```

```
Mean of Age is:  54.43894389438944
```

```
from sklearn.model_selection import train_test_split
```

```
print(df.columns)
```

```
Index(['Unnamed: 0', 'Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs',
       'RestECG', 'MaxHR', 'ExAng', 'Oldpeak', 'Slope', 'Ca', 'Thal', 'AHD'],
      dtype='object')
```

```
A = df[['Age', 'Sex', 'ChestPain', 'RestBP', 'Chol']]
B = df['Oldpeak']
A_train, A_test, B_train, B_test = train_test_split(A, B, test_size=0.25, random_state=42)
```

```
print("Shape of Traning Dataset:\t",A_train.shape,B_train.shape)
print("Shape of Testing Dataset:\t",A_test.shape,B_test.shape)
```

```
Shape of Traning Dataset:        (227, 5) (227,)
Shape of Testing Dataset:        (76, 5) (76,)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score,precision_score,recall_score,f1_score
```

```
total_samples=500
pred_positive=100
actual_positive=50
true_positive=45
```

```
Matrix=confusion_matrix(y_true=[0]*(total_samples-actual_positive) + [1] * actual_positive,
                        y_pred=[0]*(total_samples-pred_positive) + [1] * pred_positive)
```

```
print(Matrix)
```

```
[[400  50]
 [  0  50]]
```

```
Accuracy=accuracy_score([1]*pred_positive+[0]*(total_samples-pred_positive),
                        [1]*actual_positive+[0]*(total_samples-actual_positive))
print("Accuracy is:",Accuracy)
```

```
Accuracy is: 0.9
```

```
Precision=precision_score([1]*pred_positive+[0]*(total_samples-pred_positive),
                        [1]*actual_positive+[0]*(total_samples-actual_positive))
print("Precision is:",Precision)
```

```
Precision is: 1.0
```

```
F1_score=f1_score([1]*pred_positive+[0]*(total_samples-pred_positive),
                        [1]*actual_positive+[0]*(total_samples-actual_positive))
print("F1_Score is:",F1_score)
```

```
F1_Score is: 0.6666666666666666
```

```
Recall=recall_score([1]*pred_positive+[0]*(total_samples-pred_positive),
                        [1]*actual_positive+[0]*(total_samples-actual_positive))
print("Recall is:",Recall)
```

```
Recall is: 0.5
```