```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
data = pd.read_csv("/content/raw.githubusercontent.com_AdiPersonalWorks_Random_master_student_scores - student_scores.csv")
```

```
data.shape
```

```
(25, 2)
```

```
data.head(25)
```

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30
5	1.5	20
6	9.2	88
7	5.5	60
8	8.3	81
9	2.7	25
10	7.7	85
11	5.9	62
12	4.5	41
13	3.3	42
14	1.1	17
15	8.9	95
16	2.5	30
17	1.9	24
18	6.1	67
19	7.4	69
20	2.7	30
21	4.8	54
22	3.8	35
23	6.9	76
24	7.8	86

```
data.isnull()
```

	Hours	Scores
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
5	False	False
6	False	False
7	False	False
8	False	False
9	False	False
10	False	False
11	False	False
12	False	False
13	False	False

```
data.info
```

```
<bound method DataFrame.info of     Hours  Scores
0     2.5      21
1     5.1      47
2     3.2      27
3     8.5      75
4     3.5      30
5     1.5      20
6     9.2      88
7     5.5      60
8     8.3      81
9     2.7      25
10    7.7      85
11    5.9      62
12    4.5      41
13    3.3      42
14    1.1      17
15    8.9      95
16    2.5      30
17    1.9      24
18    6.1      67
19    7.4      69
20    2.7      30
21    4.8      54
22    3.8      35
23    6.9      76
24    7.8      86>
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64
dtypes: float64(1), int64(1)
memory usage: 528.0 bytes
```

```
data.describe
```

```
<bound method NDFrame.describe of     Hours  Scores
0     2.5      21
1     5.1      47
2     3.2      27
3     8.5      75
4     3.5      30
5     1.5      20
6     9.2      88
7     5.5      60
8     8.3      81
9     2.7      25
10    7.7      85
11    5.9      62
12    4.5      41
13    3.3      42
14    1.1      17
15    8.9      95
```

```
16    2.5      30
17    1.9      24
18    6.1      67
19    7.4      69
20    2.7      30
21    4.8      54
22    3.8      35
23    6.9      76
24    7.8      86>
```

```
data.describe()
```

	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

```
data.isnull().sum()
```

```
Hours     0
Scores    0
dtype: int64
```

```
data.corr()
```

	Hours	Scores
Hours	1.000000	0.976191
Scores	0.976191	1.000000

```
data.cov
```

```
<bound method DataFrame.cov of     Hours  Scores
0     2.5      21
1     5.1      47
2     3.2      27
3     8.5      75
4     3.5      30
5     1.5      20
6     9.2      88
7     5.5      60
8     8.3      81
9     2.7      25
10    7.7      85
11    5.9      62
12    4.5      41
13    3.3      42
14    1.1      17
15    8.9      95
16    2.5      30
17    1.9      24
18    6.1      67
19    7.4      69
20    2.7      30
21    4.8      54
22    3.8      35
23    6.9      76
24    7.8      86>
```

```
import seaborn as sns
sns.heatmap(data.corr(),annot=True,cmap='inferno')
```

```
<Axes: >
```


```
data['Hours'].value_counts()
```

```
2.5    2
2.7    2
4.5    1
6.9    1
3.8    1
4.8    1
7.4    1
6.1    1
1.9    1
8.9    1
1.1    1
3.3    1
5.9    1
5.1    1
7.7    1
8.3    1
5.5    1
9.2    1
1.5    1
3.5    1
8.5    1
3.2    1
7.8    1
Name: Hours, dtype: int64
```

```
data['Scores'].value_counts()
```

```
30    3
21    1
41    1
76    1
35    1
54    1
69    1
67    1
24    1
95    1
17    1
42    1
62    1
47    1
85    1
25    1
81    1
60    1
88    1
20    1
75    1
27    1
86    1
Name: Scores, dtype: int64
```

```
data['Scores'].value_counts().sum()
```

```
25
```

```
data['Hours'].value_counts().sum()
```

```
25
```

```
data['Hours'].value_counts().mean()
```

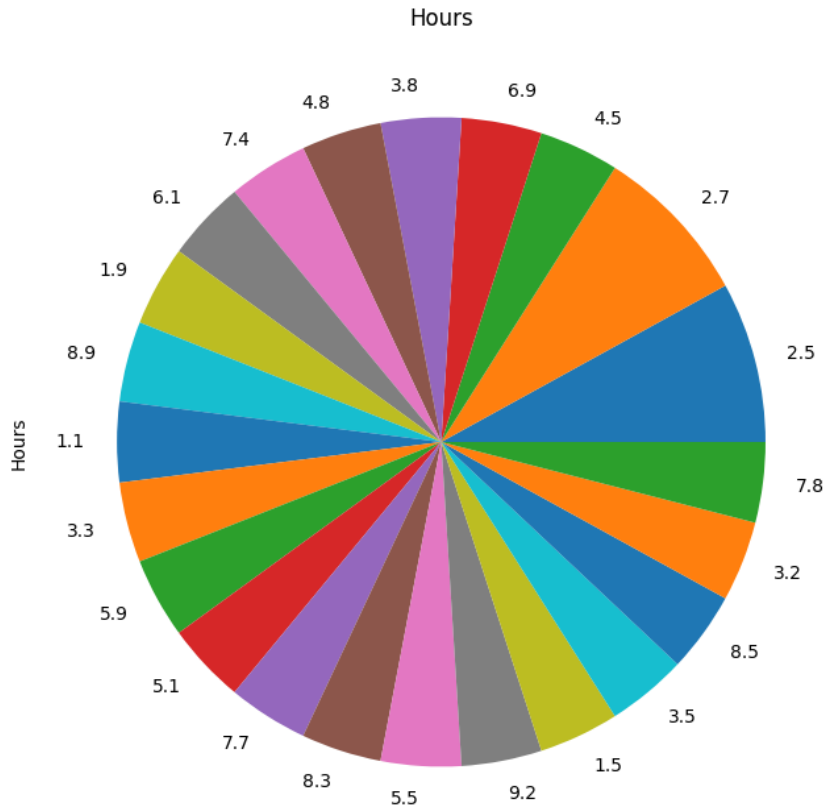
```
1.0869565217391304
```

```
data['Scores'].value_counts().mean()
```

```
1.0869565217391304
```

```
c1 = data['Hours'].value_counts()
c1.plot(kind='pie',figsize=(10,8))
plt.title('Hours')
```

```
Text(0.5, 1.0, 'Hours')
```



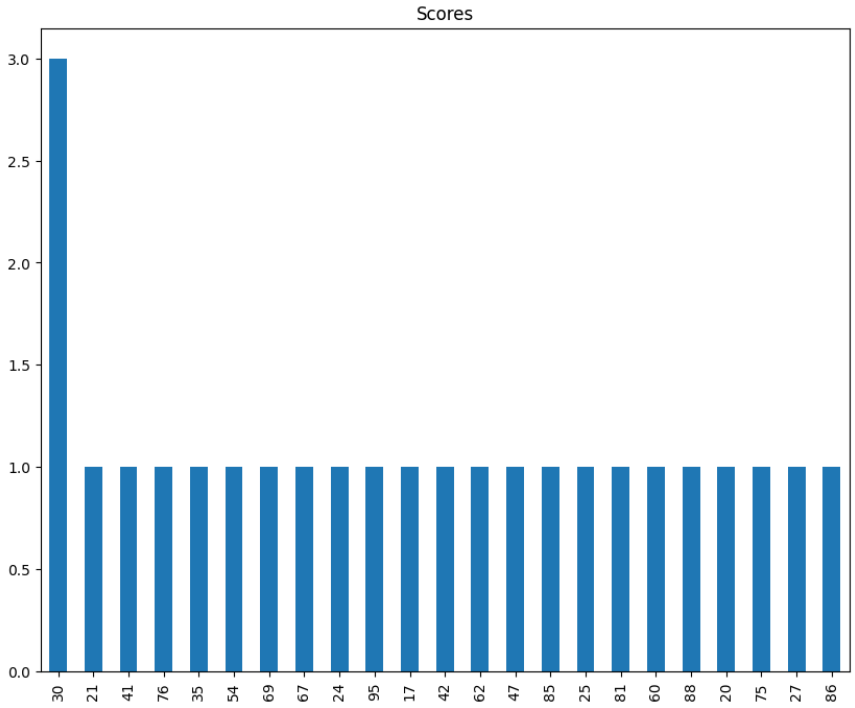

```
c1 = data['Scores'].value_counts()
c1.plot(kind='pie',figsize=(10,8))
plt.title('Scores')
```

```
Text(0.5, 1.0, 'Scores')
```



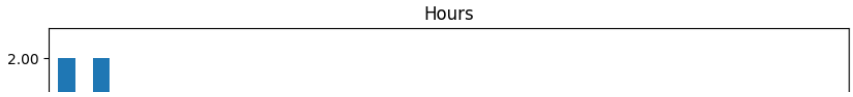
```
R1 = data['Scores'].value_counts()[:150]
R1.plot(kind='bar',figsize=(10,8))
plt.title('Scores')
```

```
Text(0.5, 1.0, 'Scores')
```

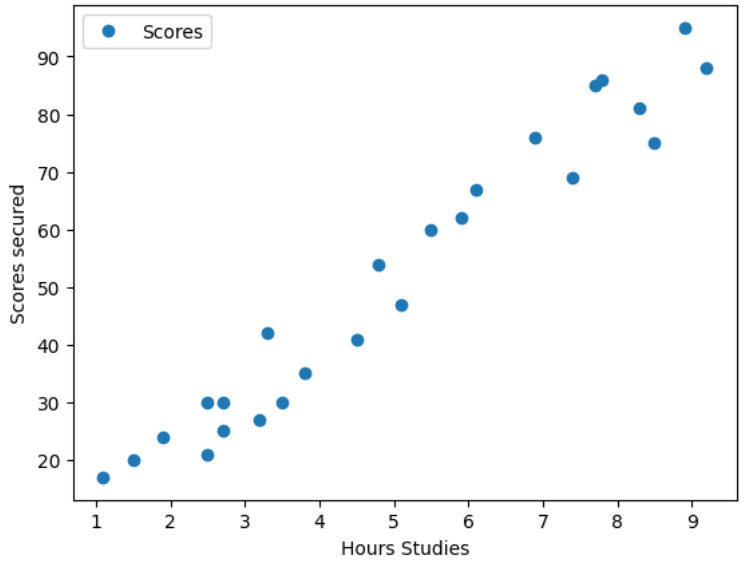



```
R1 = data['Hours'].value_counts()
R1.plot(kind='bar',figsize=(10,8))
plt.title('Hours')
```

```
Text(0.5, 1.0, 'Hours')
```




```
data.plot(x='Hours',y='Scores',style='o')
plt.xlabel('Hours Studies')
plt.ylabel('Scores secured')
plt.show()
```



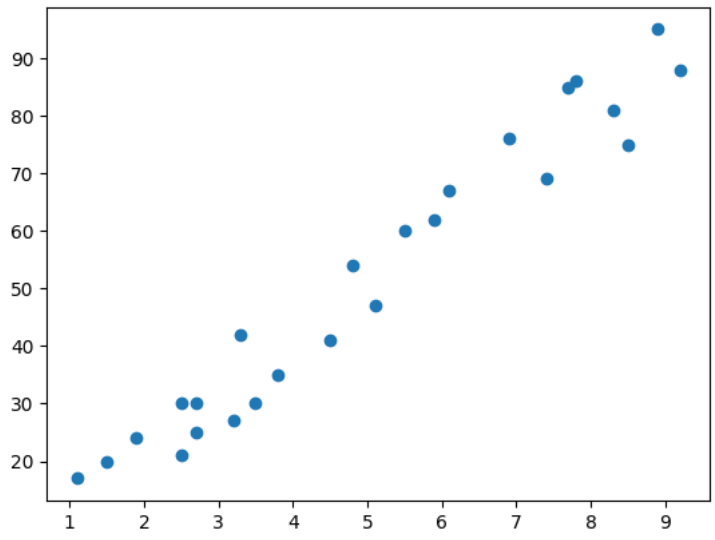

```
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                            test_size=0.2, random_state=0)


from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
plt.scatter(X,y)
plt.show()
```




```
X_train.shape
```

```
(20, 1)
```

```
X_test.shape
```

```
(5, 1)
```

```
y_train.shape
```

```
(20,)
```

```
y_test.shape
```

```
(5,)
```

```
X_test
```

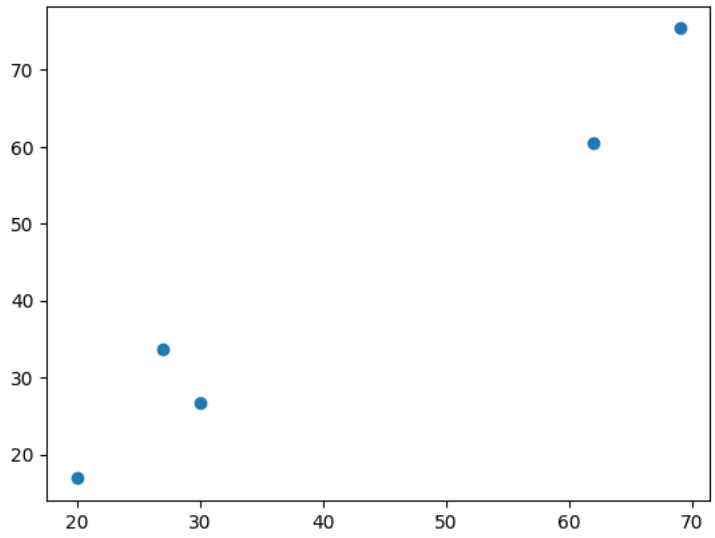
```
array([[1.5],
       [3.2],
       [7.4],
       [2.5],
       [5.9]])
```

```
pred_val = regressor.predict(X_test)
```

```
pred_val
```

```
array([16.88414476, 33.73226078, 75.357018  , 26.79480124, 60.49103328])
```

```
plt.scatter(y_test,pred_val)
plt.show()
```




```
from sklearn.metrics import mean_squared_error
np.sqrt( mean_squared_error( y_test, pred_val ) )
```

```
4.647447612100373
```

```
from sklearn import metrics
print('Mean Absolute Error:',
      metrics.mean_absolute_error(y_test, pred_val))
```

```
Mean Absolute Error: 4.183859899002982
```

```
from sklearn.metrics import r2_score
print( "R-squared: ",r2_score( y_test, pred_val ) )
```

```
R-squared:  0.9454906892105354
```