Name: Dipa Jarecha

Task-1 : Predict the percentage of an student based on the no. of study hours.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
```

```
data=pd.read_csv("http://bit.ly/w-data")
data.head()
```

	Hours	Scores
0	2.5	21
1	5.1	47
2	3.2	27
3	8.5	75
4	3.5	30

```
data.shape
```

```
(25, 2)
```

```
data.describe()
```

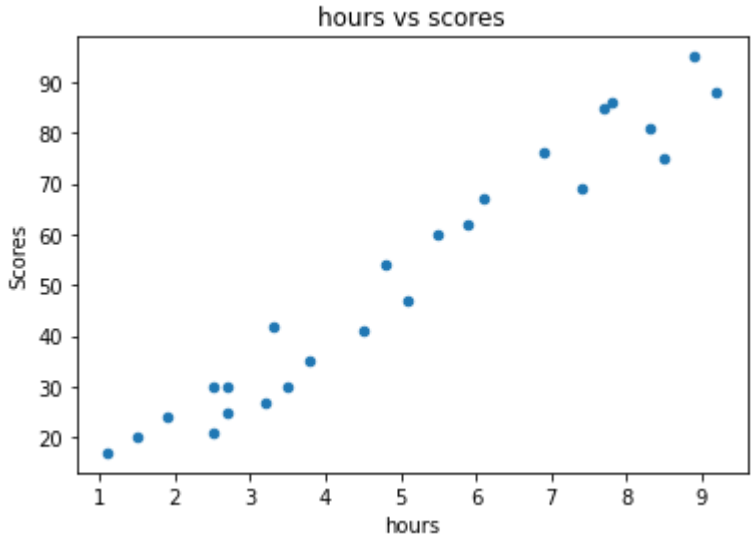
	Hours	Scores
count	25.000000	25.000000
mean	5.012000	51.480000
std	2.525094	25.286887
min	1.100000	17.000000
25%	2.700000	30.000000
50%	4.800000	47.000000
75%	7.400000	75.000000
max	9.200000	95.000000

```
data.info()
```

```
data.info()
```

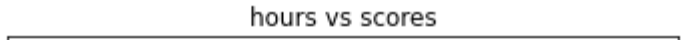
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25 entries, 0 to 24
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   Hours   25 non-null     float64
 1   Scores  25 non-null     int64  
dtypes: float64(1), int64(1)
memory usage: 528.0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

```
data.plot(x='Hours',y='Scores',kind='scatter',style='o')
plt.title('hours vs scores')
plt.xlabel('hours')
plt.ylabel('Scores')
plt.show()
```




```
sns.regplot(x="Hours",y="Scores",data=data)
plt.title('hours vs scores')
```

```
Text(0.5, 1.0, 'hours vs scores')
```




```
X=data.iloc[:,:-1]
y=data.iloc[:,-1]
print(X)
print(y)
```

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

```
22    35
23    76
24    86
Name: Scores, dtype: int64
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=0)
```

```
from sklearn.linear_model import LinearRegression
```

```
model=LinearRegression()
model.fit(X_train,y_train)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

```
z = model.predict(X_test)
```

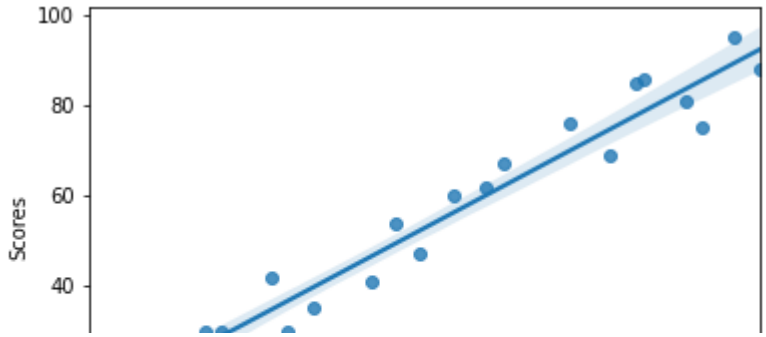
```
predict = pd.DataFrame({'Actual':y_test,'Predicted':z})
```

```
predict
```

	Actual	Predicted
5	20	17.053665
2	27	33.694229
19	69	74.806209
16	30	26.842232
11	62	60.123359
22	35	39.567369
17	24	20.969092
24	86	78.721636

```
sns.regplot(X,y,data=data)
```

```
/usr/local/lib/python3.6/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass th
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f8f8fe102b0>
```



Now model will take a value and gives the output according to requirement.

```
hours = [[9.25]]
result = model.predict(hours)
print("No: of hours ={} ".format(hours))
print("Score = {} ".format(result))
```

```
No: of hours =[[9.25]]
Score = [92.91505723]
```

```
from sklearn import metrics
```

```
print("Root Mean Square",metrics.mean_squared_error(y_test,z))
```

```
Root Mean Square 22.965097212700428
```

```
print("Mean Absolute Error",metrics.mean_absolute_error(y_test,p))
```

```
Mean Absolute Error 4.419727808027651
```

Mean absolute error is less. so model is performing very well.