In [28]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

In [3]:

```
#importing data

marks= pd.read_csv('student_scores.csv')
print (marks)
```

```
    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
```

In [4]:

```
#exploring data
marks.describe()
```

Out[4]:

	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

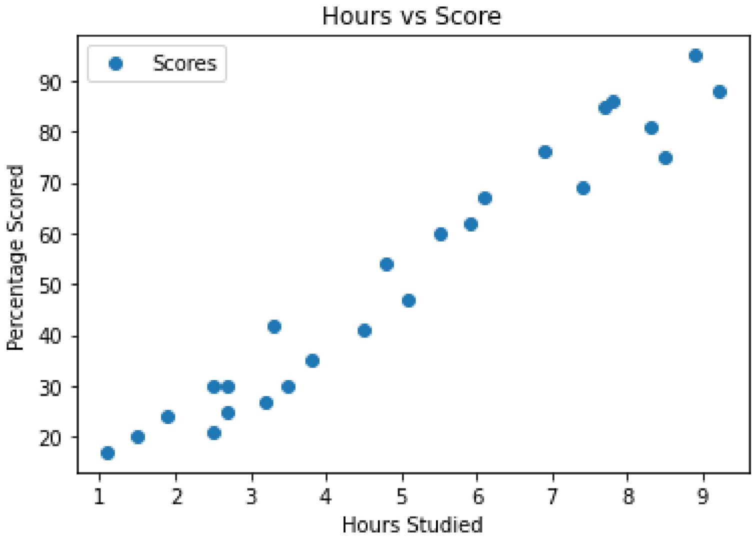
In [5]:

```
marks.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: 464.0 bytes
```

In [6]:

```
marks.plot(x='Hours',y='Scores',style='o')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Scored')
plt.title('Hours vs Score')
plt.show()
print(marks.corr())
```



```
           Hours    Scores
Hours   1.000000  0.976191
Scores  0.976191  1.000000
```

In [7]:

```
#defining X and y and splitting the data
X = marks.iloc[:, :-1].values
y = marks.iloc[:, 1].values

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=0)
```

In [30]:

```
#Training on Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[30]:

```
LinearRegression()
```

In [31]:

```
#Predicting the test set
pred_y = regression.predict(X_test)
prediction = pd.DataFrame({'Hours': [i[0] for i in X_test],
                           'Predicted % Marks': [k for k in pred_y]})
prediction
```

Out[31]:

	Hours	Predicted % Marks
0	1.5	16.884145
1	3.2	33.732261
2	7.4	75.357018
3	2.5	26.794801

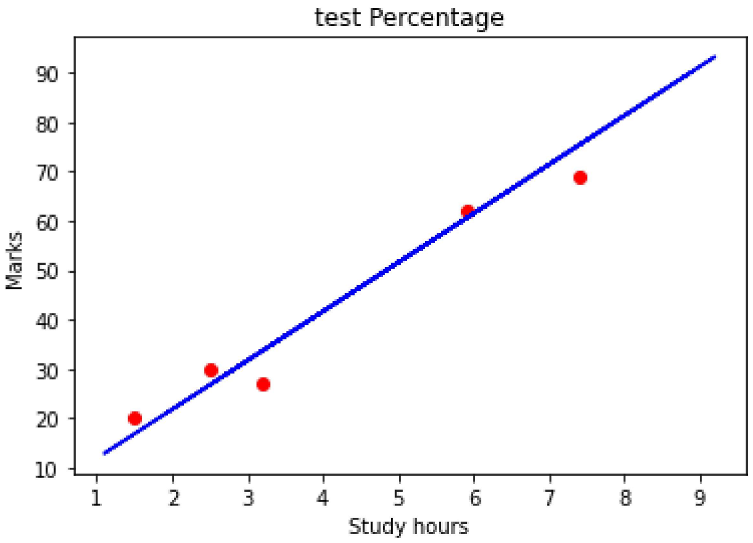
	Hours	Predicted % Marks
4	5.9	60.491033

In [43]:

```
#Visualise test
plt.scatter(X_test, y_test, color ='red')
plt.plot(X_train, regressor.predict(X_train), color='blue')
plt.title('test Percentage')
plt.xlabel("Study hours")
plt.ylabel("Marks")
```

Out[43]:

```
Text(0, 0.5, 'Marks')
```



In [37]:

```
# predicting training set

pred_y = regression.predict(X_train)
prediction = pd.DataFrame({'Hours': [i[0] for i in X_train],
                           'Predicted % Marks': [k for k in pred_y]})
prediction
```

Out[37]:

	Hours	Predicted % Marks
0	3.8	39.678655
1	1.9	20.848407
2	7.8	79.321281
3	6.9	70.401690
4	1.1	12.919882
5	5.1	52.562508
6	7.7	78.330215
7	3.3	34.723326
8	8.3	84.276609
9	9.2	93.196200
10	6.1	62.473165
11	3.5	36.705458
12	2.7	28.776933

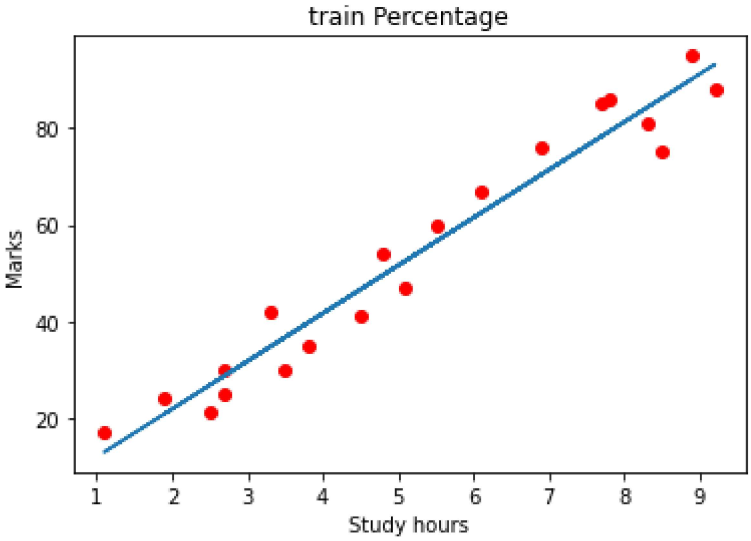
	Hours	Predicted % Marks
13	5.5	56.526771
14	2.7	28.776933
15	8.5	86.258740
16	2.5	26.794801
17	4.8	49.589311
18	8.9	90.223003
19	4.5	46.616114

In [44]:

```
#train Visualisation
plt.scatter(X_train, y_train, color ='red')
plt.plot(X_train, regressor.predict(X_train))
plt.title('train Percentage')
plt.xlabel("Study hours")
plt.ylabel("Marks")
```

Out[44]:

```
Text(0, 0.5, 'Marks')
```



What will be predicted score if a student studies for 9.25 hrs/ day?

In [51]:

```
hours = [9.25]
answer = regression.predict([hours])
```

In [50]:

```
print (answer)
```

```
[93.69173249]
```

Hence the student scores 93.69% if he/she studies for 9.25hrs/ day

In [ ]:

In [ ]: