In [1]:

```
# importing the libraries
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as mp
from sklearn.metrics import mean_absolute_error
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

In [2]:

```
# Reading the Data
data = pd.read_csv('http://bit.ly/w-data')
data.head(10)
```

Out[2]:

	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

In [3]:

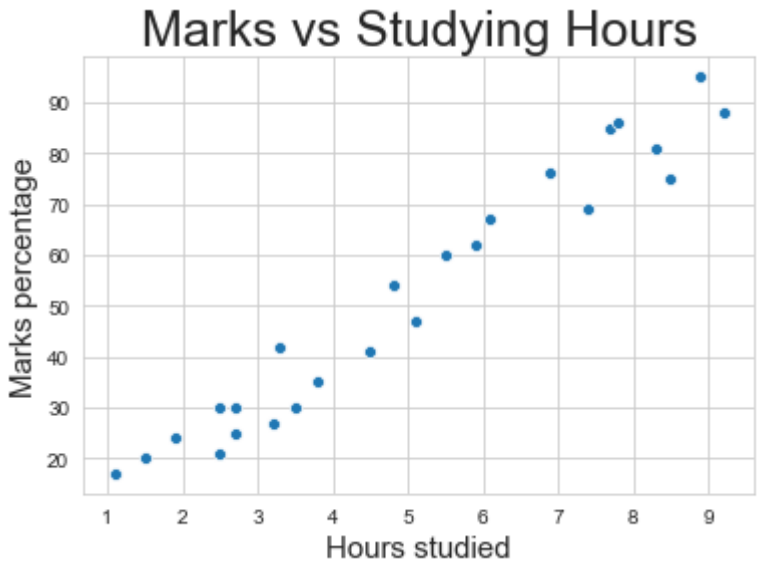
```
#checking if there are any null values in the dataset
data.isnull == True
```

Out[3]:

```
False
```

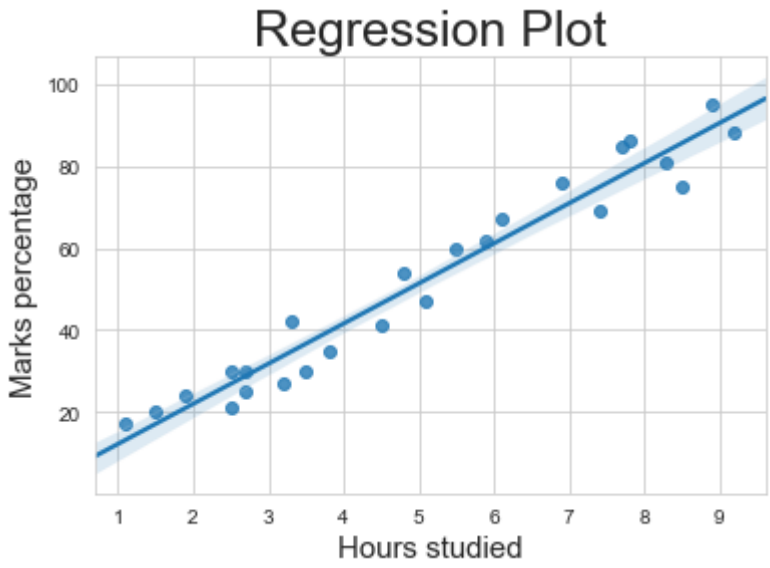
In [4]:

```
sb.set_style('whitegrid')
sb.scatterplot(y= data['Scores'], x= data['Hours'])
mp.title('Marks vs Studying Hours',size=25)
mp.ylabel('Marks percentage', size=15)
mp.xlabel('Hours studied', size=15)
mp.show()
```



In [5]:

```
sb.regplot(x= data['Hours'], y= data['Scores'])
mp.title('Regression Plot',size=25)
mp.ylabel('Marks percentage', size=15)
mp.xlabel('Hours studied', size=15)
mp.show()
print(data.corr())
```




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

In [6]:

```
#training the model
#defining x and y axis from the data
X = data.iloc[:, :-1].values
y = data.iloc[:, 1].values

# Spliting the Data in two
train_X, val_X, train_y, val_y = train_test_split(X, y, random_state = 0)
```

In [7]:

```
#fitting the data into the model
regression = LinearRegression()
regression.fit(train_X, train_y)
print("Model is Trained")
```

```
Model is Trained
```

In [8]:

```
#predicting the perctange of marks
pred_y = regression.predict(val_X)
prediction = pd.DataFrame({'Hours': [i[0] for i in val_X], 'Predicted Marks': [k for k
in pred_y]})
prediction
```

Out[8]:

	Hours	Predicted Marks
0	1.5	16.844722
1	3.2	33.745575
2	7.4	75.500624
3	2.5	26.786400
4	5.9	60.588106
5	3.8	39.710582
6	1.9	20.821393

In [9]:

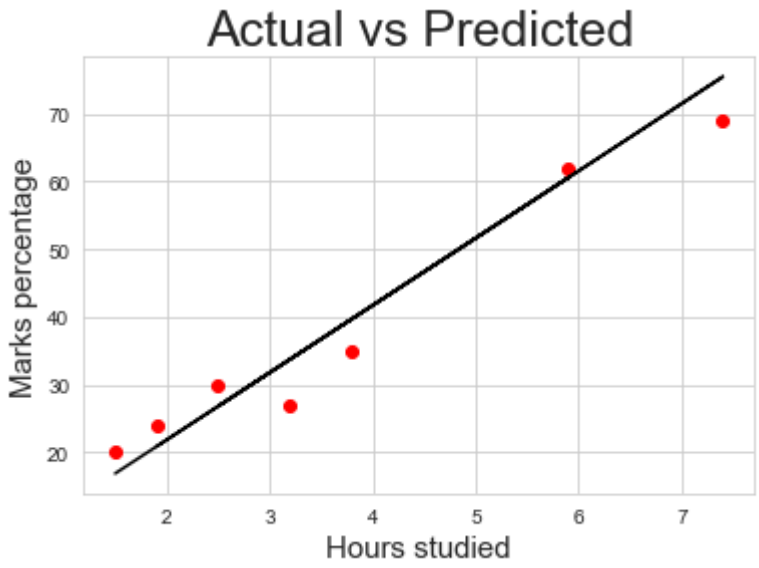
```
#comparing predicted marks to that of actual marks
compare_scores = pd.DataFrame({'Actual Marks': val_y, 'Predicted Marks': pred_y})
compare_scores
```

Out[9]:

	Actual Marks	Predicted Marks
0	20	16.844722
1	27	33.745575
2	69	75.500624
3	30	26.786400
4	62	60.588106
5	35	39.710582
6	24	20.821393

In [10]:

```
mp.scatter(x=val_X, y=val_y, color='red')
mp.plot(val_X, pred_y, color='Black')
mp.title('Actual vs Predicted', size=25)
mp.ylabel('Marks percentage', size=15)
mp.xlabel('Hours studied', size=15)
mp.show()
```



In [11]:

```
#accuracy of the model
print('Mean absolute error: ',mean_absolute_error(val_y,pred_y))
```

```
Mean absolute error:  4.130879918502486
```

In [12]:

```
#Output for the give data of hours
hours = [9.25]
answer = regression.predict([hours])
print("Score = {}".format(round(answer[0],3)))
```

```
Score = 93.893
```

In [ ]:

```

```