In [1]:

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

In [2]:

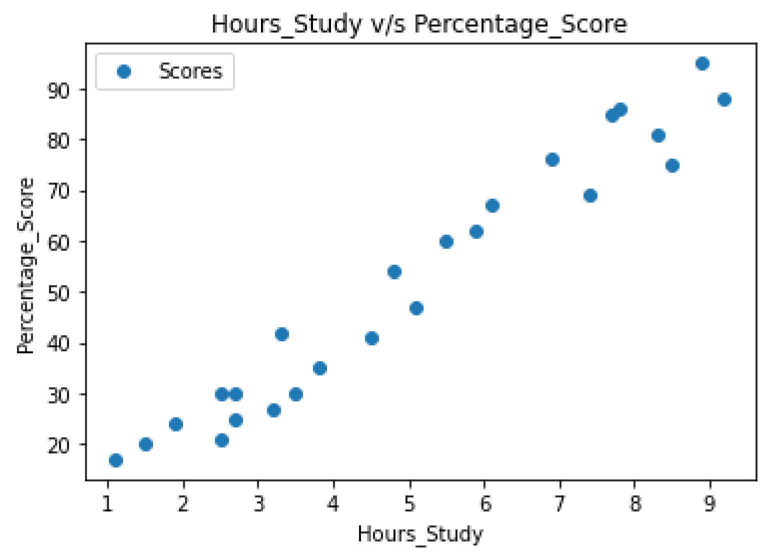
```
url = 'http://bit.ly/w-data'
s_data = pd.read_csv(url)
s_data.head(10)
```

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 [4]:

```
s_data.plot(x='Hours', y='Scores', style = 'o')
plt.title('Hours_Study v/s Percentage_Score')
plt.xlabel('Hours_Study')
plt.ylabel('Percentage_Score')
plt.show()
```



In [10]:

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

In [18]:

```
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)
```

In [8]:

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

In [12]:

```
Regressor1 = LinearRegression()
Regressor1.fit(X,y)
print("Training_model")
```

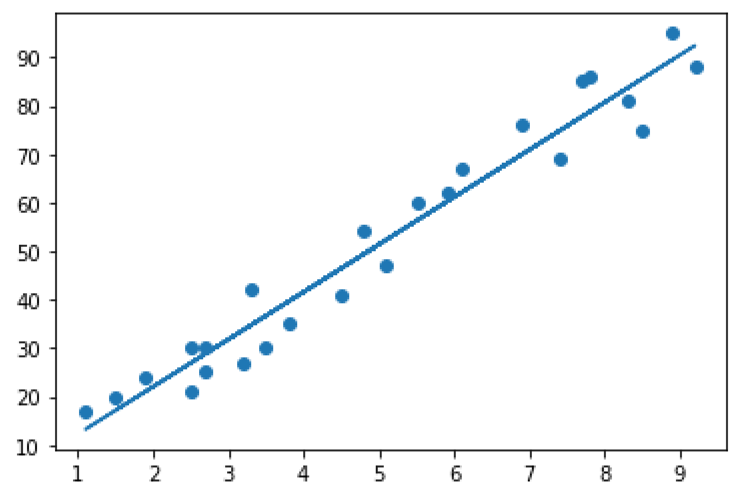
```
Training_model
```

In [13]:

```
#Plotting the regression line
line = Regressor1.coef_*X+Regressor1.intercept_
```

In [14]:

```
#Plotting for Test
plt.scatter(X,y)
plt.plot(X,line)
plt.show()
```



In [15]:

```
#PREDICTIONS
print(X_test)
y_pred = Regressor1.predict(X_test)
```

```
[[1.5]
 [3.2]
 [7.4]
 [2.5]
 [5.9]]
```

In [16]:

```
# Comparing Actual v/s Predicted
data_frame = pd.DataFrame({'Actual':y_test, 'Predicted': y_pred})
data_frame
```

Out[16]:

	Actual	Predicted
0	20	17.147378
1	27	33.766244
2	69	74.824618
3	30	26.923182
4	62	60.160913

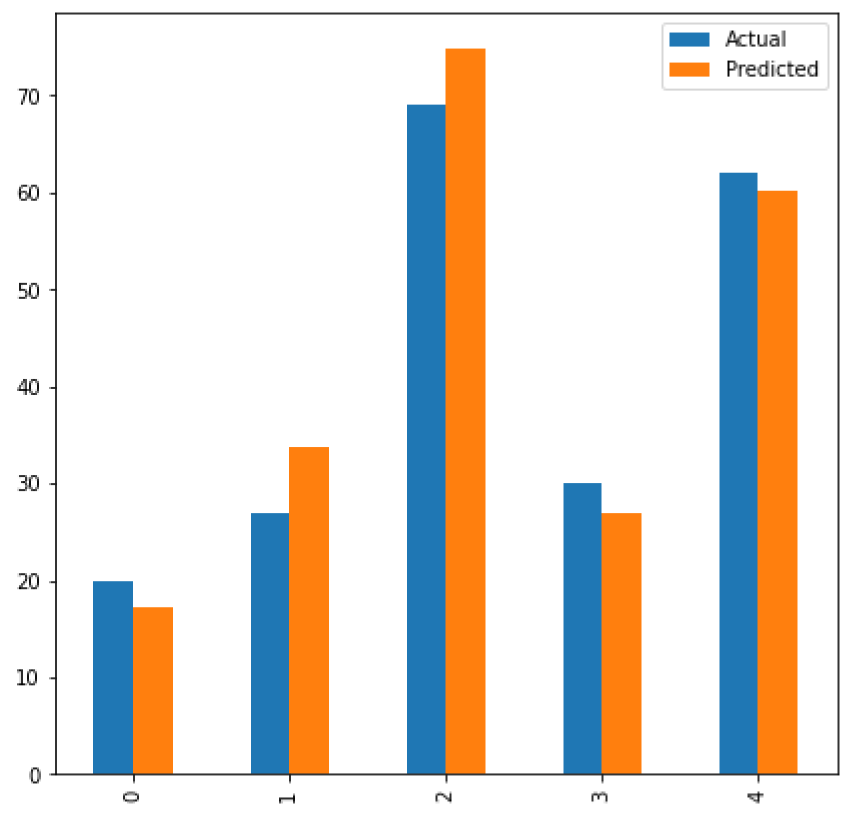
In [19]:

```
print("Training Score", Regressor1.score(X_train,y_train))
print("Testing Score", Regressor1.score(X_test,y_test))
```

```
Training Score 0.9512837351709387
Testing Score 0.9491748734859171
```

In [21]:

```
# Plotting the bar to depict the actual and predicted value
data_frame.plot(kind = 'bar', figsize = (7,7))
plt.show()
```



In [22]:

```
#predicted for 9.25 hr
hours = 9.25
test = np.array([hours])
test = test.reshape(-1,1)
own_pred = Regressor1.predict(test)
print("No of Hours = {}".format(hours))
print("Predicted Score = {}".format(own_pred[0]))
```

```
No of Hours = 9.25
Predicted Score = 92.90985477015731
```

In [26]:

```
#Evaluating Model
import numpy as np
from sklearn import metrics
print("Mean Absolute Error: ", metrics.mean_absolute_error(y_test,y_pred))
print("Mean Squared Error: ", metrics.mean_squared_error(y_test,y_pred))
print('Root Mean Squared Error: ', np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
print('Explained Variance Score: ', metrics.explained_variance_score(y_test,y_pred))
```

```
Mean Absolute Error:  4.071877793635608
Mean Squared Error:  20.1389481299402
Root Mean Squared Error:  4.487643939746134
Explained Variance Score:  0.951522433518808
```

In [ ]: