5. Assignment on simple regression: Build an application where it can predict a salary based on year of experience using Single Variable Linear Regression (Use Salary dataset from Kaggle). Display the coefficient and intercept. Also visualize the results by plotting the graphs on both training and testing dataset.

In [1]:
```
#import all the necessory libraries
```

In [2]:
```
import pandas as pd
import matplotlib.pyplot as mlt
import numpy as np
dataset=pd.read_csv('Salary_dataset.csv')
#x=dataset.iloc[:,:-1].values

#assign the values of x and y for the model building.

x=dataset['YearsExperience'].values
x.reshape(30,1)
y=dataset['Salary'].values
y.reshape(30,1)
```

Out[2]:
```
array([[ 39344.],
       [ 46206.],
       [ 37732.],
       [ 43526.],
       [ 39892.],
       [ 56643.],
       [ 60151.],
       [ 54446.],
       [ 64446.],
       [ 57190.],
       [ 63219.],
       [ 55795.],
       [ 56958.],
       [ 57082.],
       [ 61112.],
       [ 67939.],
       [ 66030.],
       [ 83089.],
       [ 81364.],
       [ 93941.],
       [ 91739.],
       [ 98274.],
       [101303.],
       [113813.],
       [109432.],
       [105583.],
       [116970.],
       [112636.],
       [122392.],
       [121873.]])
```

In [39]:
```
#Draw the scattered graph of the given values x and y
mlt.scatter(x,y,color='red')
mlt.title('Salary vs Expirience')
mlt.xlabel('Experience')
mlt.ylabel('salary')
mlt.show()
```

In [25]:
```
#Reshape the x and values to set equal values of x and y
x=x.reshape(30,1)
y=y.reshape(30,1)
x.shape
```

Out[25]:
```
(30, 1)
```

Divide the complete dataset into training and testing data

In [26]:
```
# divide the dataset in some amount of training and testing data
from sklearn.model_selection import train_test_split

# random_state => seed value used by random number generator
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=0)
```

Implement Classifier based on Simple Linear Regression

In [27]:
```
from sklearn.linear_model import LinearRegression
```

In [28]:
```
#model implementation
model=LinearRegression()
```

In [31]:
```
#fit the implemented model with respect to x_train and y_train values
model.fit(x_train,y_train)
```

Out[31]:
```
LinearRegression()
```

In [32]:
```
#predict the model output with the help of predict function
y_predict=model.predict(x_test)
```

Find the mean squared error of the model

In [34]:
```
from sklearn.metrics import mean_squared_error
```

In [35]:
```
mean_squared_error(y_test,y_predict)
```

Out[35]:
```
23370078.80083297
```

Plotting the Best-fit Linear Regression Graph

- Formula for the Linear Regression : Salary = B0 + B1*(Experience)
- B0 = intercept => salary when experience is 0, B1 = slope => increase in salary with unit increase in salary

In [38]:
```
mlt.scatter(x_train, y_train, color='red')
```

```
mlt.plot(x_train, model.predict(x_train))
```

Out[38]:

```
[<matplotlib.lines.Line2D at 0x2b510601910>]
```