In [1]:

```
#Simple Linaer Regression
#Step 1: Importing the dataset
import pandas as pd   #library
df=pd.read_csv("Salary_Data.csv")
```

In [2]:

```
df.head()
```

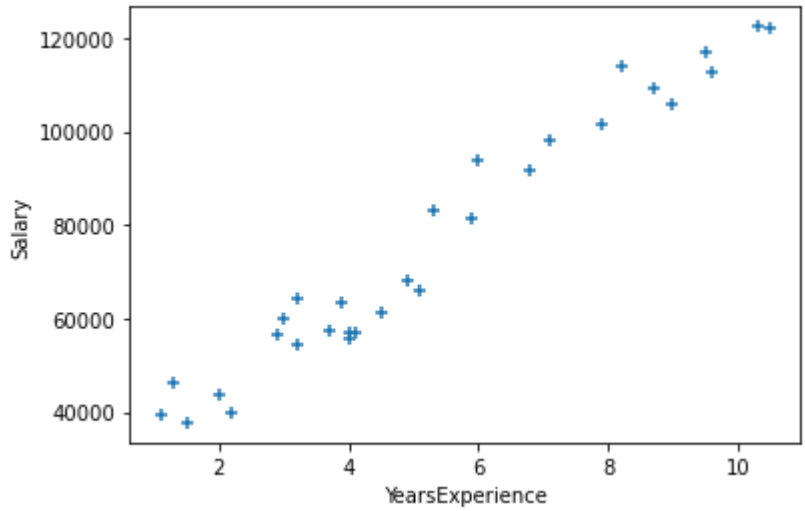
Out[2]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

In [3]:

```
#Visualize the dataset

import matplotlib.pyplot as plt
plt.scatter(df.YearsExperience,df.Salary,marker='+')
plt.xlabel("YearsExperience")
plt.ylabel("Salary")
plt.show()
```



In [ ]:

```
#X -> Independent variiable
#Y -> Dependent Variable
```

In [4]:

```
#Method 1 to load data in form of array - for easy computation by model
X=df.iloc[:, :-1].values
Y=df.iloc[:,1:].values  #we get an array
X
```

Out[4]:

```
array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
```

In [5]:

```
Y
```

Out[5]:

```
array([[ 39343.],
       [ 46205.],
       [ 37731.],
       [ 43525.],
       [ 39891.],
       [ 56642.],
       [ 60150.],
       [ 54445.],
       [ 64445.],
       [ 57189.],
       [ 63218.],
       [ 55794.],
       [ 56957.],
       [ 57081.],
       [ 61111.],
       [ 67938.],
       [ 66029.],
       [ 83088.],
       [ 81363.],
       [ 93940.],
       [ 91738.],
       [ 98273.],
       [101302.],
       [113812.],
       [109431.],
       [105582.],
       [116969.],
       [112635.],
       [122391.],
       [121872.]])
```

In [6]:

```
import numpy as np #method 2 to load data as form of arrays - by library usage
train_x = np.array(df[["YearsExperience"]])
train_y = np.array(df[["Salary"]])
print(train_x)
```

```
[[ 1.1]
 [ 1.3]
 [ 1.5]
 [ 2. ]
 [ 2.2]
 [ 2.9]
 [ 3. ]
 [ 3.2]
 [ 3.2]
 [ 3.7]
 [ 3.9]
 [ 4. ]
 [ 4. ]
 [ 4.1]
 [ 4.5]
 [ 4.9]
 [ 5.1]
 [ 5.3]
 [ 5.9]
 [ 6. ]
 [ 6.8]
 [ 7.1]
 [ 7.9]
 [ 8.2]
 [ 8.7]
 [ 9. ]
 [ 9.5]
 [ 9.6]
 [10.3]
 [10.5]]
```

In [7]:

```
#Splitting the dataset into Train & Test Dataset
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)
```

In [8]:

```
X_train
```

Out[8]:

```
array([[ 7.9],
       [ 2.9],
       [ 5.1],
       [ 3.2],
       [ 4.5],
       [ 8.2],
       [ 6.8],
       [ 1.3],
       [10.5],
       [ 3. ],
       [ 2.2],
       [ 5.9],
       [ 6. ],
       [ 3.7],
       [ 3.2],
       [ 9. ],
       [ 2. ],
       [ 1.1],
       [ 7.1],
       [ 4.9],
       [ 4. ]])
```

In [9]:

```
y_train
```

Out[9]:

```
array([[101302.],
       [ 56642.],
       [ 66029.],
       [ 64445.],
       [ 61111.],
       [113812.],
       [ 91738.],
       [ 46205.],
       [121872.],
       [ 60150.],
       [ 39891.],
       [ 81363.],
       [ 93940.],
       [ 57189.],
       [ 54445.],
       [105582.],
       [ 43525.],
       [ 39343.],
       [ 98273.],
       [ 67938.],
       [ 56957.]])
```

In [10]:

```
#Training the Model - Algorithm Fitting
from sklearn.linear_model import LinearRegression
model=LinearRegression()
model.fit(X_train,y_train)
```

Out[10]:

```
LinearRegression()
```

In [11]:

```
#Testing the Model
y_pred=model.predict(X_test)
```

In [12]:

```
#Evaluating the Model
#Train set results
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, model.predict(X_train), color = 'blue')
plt.title('Salary vs Experience (Training set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

In [13]:

```
#Test Set Results
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_test, model.predict(X_test), color = 'blue')
plt.title('Salary vs Experience (Testing set)')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.show()
```

In [14]:

```
#Finding the accuracy - How Accurate the model is?
import numpy as np
print("Mean sum of squares (MSE): %.2f" % np.mean(((y_pred - y_test)** 2)**0.5))
```

```
Mean sum of squares (MSE): 3737.42
```

In [ ]:

```
#New Prediction
years_exp=float(input("Enter Years Of Experience:"))
sal=model.predict([[years_exp]])
print(sal)
```

In [ ]:

```

```

In [ ]:

```

```