In [52]:
```
#first of all import all required files
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

In [53]:
```
#now upload the dataset using pandas frame
data=pd.read_csv(r'Position_Salaries.csv.xls')
data
```

Out[53]:

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-Level	9	500000
9	CEO	10	1000000

In [54]:
```
#display the first five dataset
data.head()
```

Out[54]:

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000

In [55]:
```
#here i drop position columns that  the categorical data frame
data=data.drop("Position",axis="columns")
```

In [56]:
```
#now extract the independent variable x and dependent variable y
x=data.iloc[:,:-1].values
x
```

Out[56]:
```
array([[ 1],
       [ 2],
       [ 3],
       [ 4],
       [ 5],
       [ 6],
       [ 7],
       [ 8],
       [ 9],
       [10]], dtype=int64)
```

In [57]:
```
y=data.iloc[:,-1].values
y=np.reshape(y,(10,1))
y
```

Out[57]:
```
array([[  45000],
       [  50000],
       [  60000],
       [  80000],
       [ 110000],
       [ 150000],
       [ 200000],
       [ 300000],
       [ 500000],
       [1000000]], dtype=int64)
```

In [58]:
```
x.shape
```

Out[58]: (10, 1)

In [59]:
```
y.shape
```

Out[59]: (10, 1)

In [60]:
```
#here i import  LinearRegression model from sklearn
from sklearn.linear_model import LinearRegression
#now i call the model and store in different variable
linear=LinearRegression()
linear
```

Out[60]: LinearRegression()

In [61]:
```
#here i fit the linearRegression model using dependent and independent variable
model=linear.fit(x,y)
model
```

Out[61]: LinearRegression()

In [62]:
```
#here i check the score of the model
model.score(x,y)
```

Out[62]: 0.6690412331929895

In [63]:
```
#now i predict the y value using x
y_prd=model.predict(x)
y_prd
```

Out[63]:
```
array([[-114454.54545455],
       [ -33575.75757576],
       [  47303.03030303],
       [ 128181.81818182],
       [ 209060.60606061],
       [ 289939.39393939],
       [ 370818.18181818],
       [ 451696.96969697],
       [ 532575.75757576],
       [ 613454.54545455]])
```

In [64]:
```
#find the coeficient the model
model.coef_
```

Out[64]: array([[80878.78787879]])

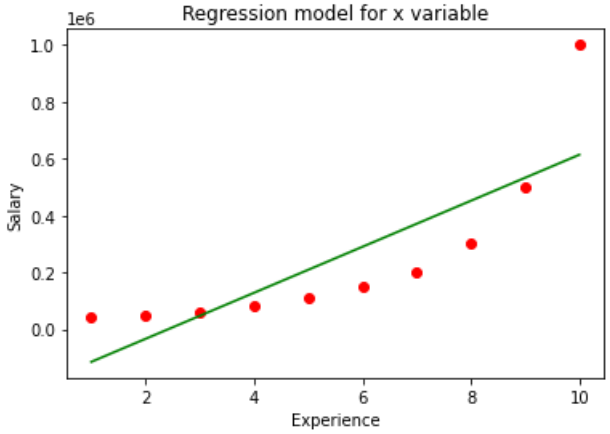
In [65]:
```
model.predict([[7]])
```

Out[65]: array([[370818.18181818]])

In [66]:
```
#find the mean_squared_error and mean_absolute_error
from sklearn.metrics import mean_squared_error,mean_absolute_error
mean=mean_squared_error(y,y_prd)
mean
```

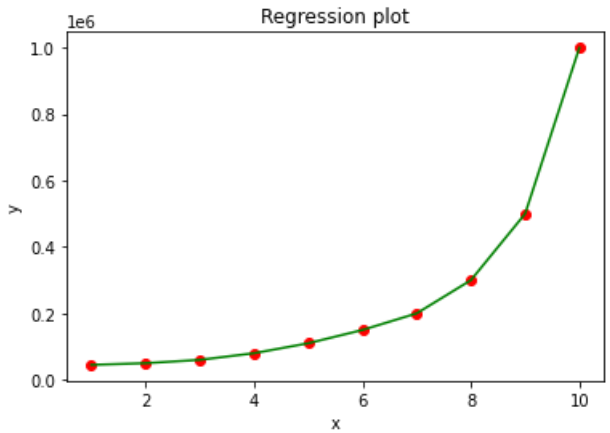
Out[66]: 26695878787.878784

In [67]:
```
#for drawing the scatter plot i have to import matplotlib.pyplot
import matplotlib.pyplot as plt
plt.scatter(x,y,color="Red")
plt.plot(x,model.predict(x),color="green")
plt.title("Regression model for x variable")
plt.xlabel("Experience")
plt.ylabel("Salary")
plt.show()
```



In [68]:

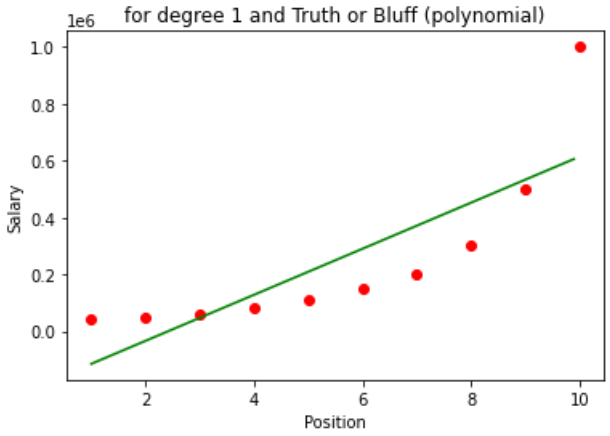
```
#for see the how look polynomial line in this graph
plt.scatter(x,y,color="red")
plt.plot(x,y,color="green")
plt.xlabel('x')
plt.ylabel('y')
plt.title('Regression plot')
plt.show()
```



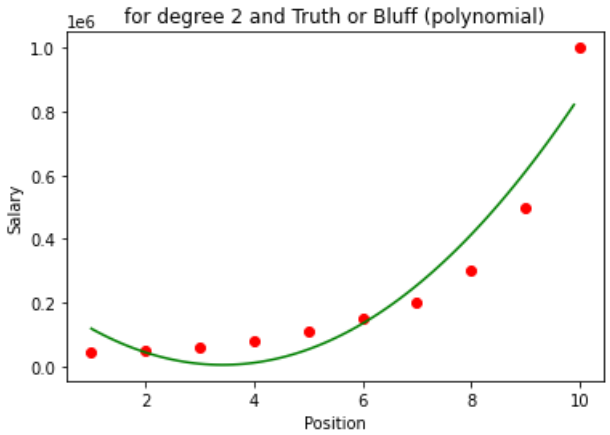
In [94]:

```
#here i import PolynomialFeatures from sklearn.preprocessing
#visualize the polynomial Regression model :-
from sklearn.preprocessing import PolynomialFeatures
for i in range(1,12):
    poly_reg=PolynomialFeatures(degree=i)
    x_poly=poly_reg.fit_transform(x)
    lin_reg2=LinearRegression()
    model=lin_reg2.fit(x_poly,y)
    d= model.predict(x_poly)
    print(d)
    plt.scatter(x,y,color='red')
    y_prd=model.predict(x_poly)
    x_grid=np.arange(min(x),max(x),0.1)
    x_grid=x_grid.reshape(len(x_grid),1)
    plt.plot(x_grid,lin_reg2.predict(poly_reg.fit_transform(x_grid)),color='green')
    plt.title(f"for degree {i} and Truth or Bluff (polynomial)")
    plt.xlabel('Position')
    plt.ylabel('Salary')
    plt.show()
```

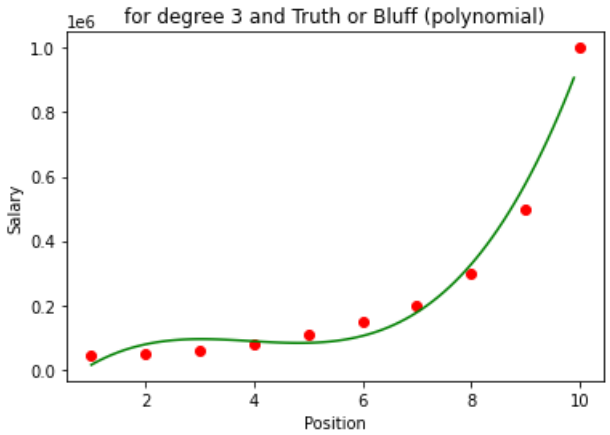
```
[[-114454.54545455]
 [ -33575.75757576]
 [  47303.03030303]
 [ 128181.81818182]
 [ 209060.60606061]
 [ 289939.39393939]
 [ 370818.18181818]
 [ 451696.96969697]
 [ 532575.75757576]
 [ 613454.54545455]]
```



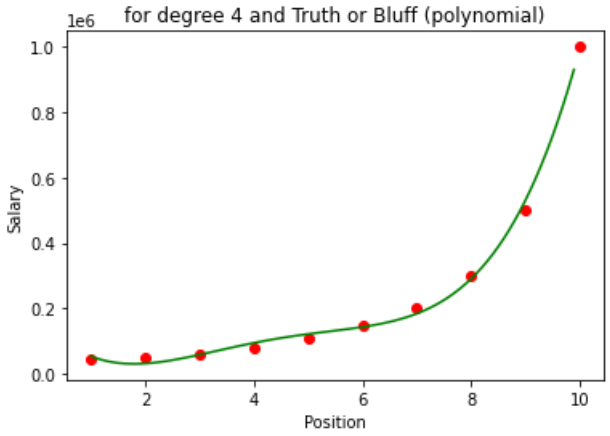

```
[[118727.27272727]
 [ 44151.51515152]
 [  8439.39393939]
 [ 11590.90909091]
 [ 53606.06060606]
 [134484.84848485]
 [254227.27272727]
 [412833.33333333]
 [610303.03030303]
 [846636.36363636]]
```



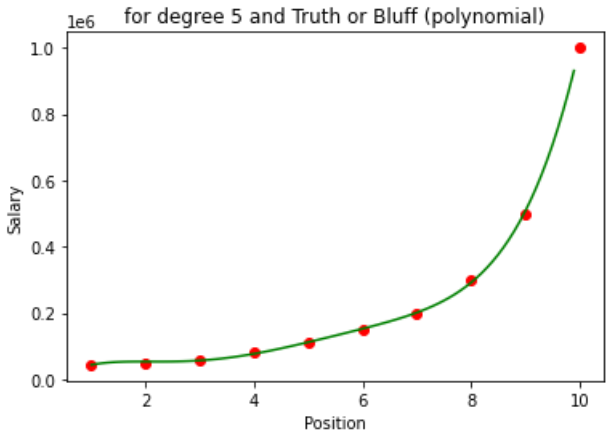

```
[[ 14902.09790208]
 [ 78759.9067599 ]
 [ 94960.37296038]
 [ 88223.77622379]
 [ 83270.39627041]
 [104820.51282052]
 [177594.40559441]
 [326312.35431235]
 [575694.63869463]
 [950461.53846153]]
```



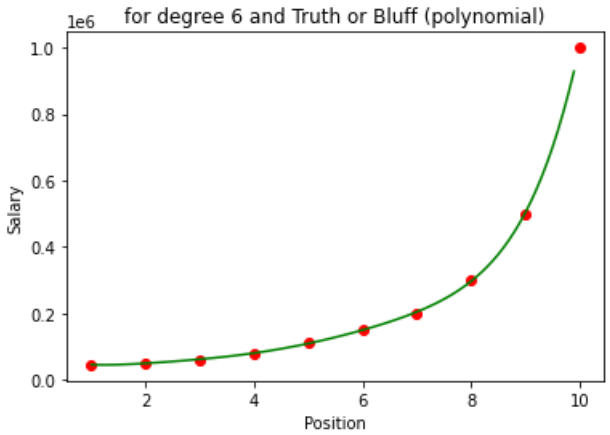

```
[[ 53356.64335691]
 [ 31759.90676   ]
 [ 58642.19114225]
 [ 94632.86713296]
 [121724.94172506]
 [143275.05827517]
 [184003.49650353]
 [289994.17249408]
 [528694.63869438]
 [988916.08391567]]
```



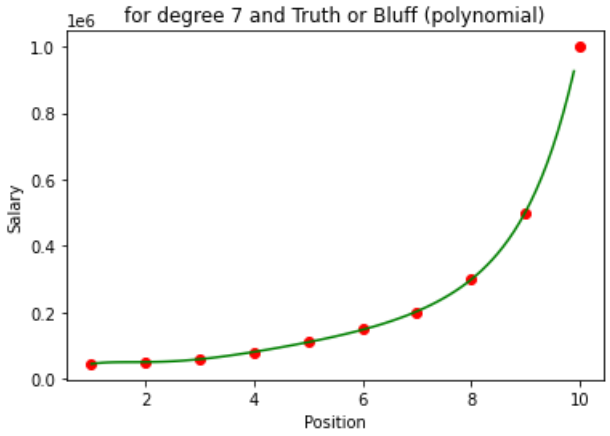

```
[[ 43895.10489574]
 [ 53836.8298369 ]
 [ 57065.2680655 ]
 [ 77286.71328718]
 [112263.40326387]
 [152736.59673683]
 [201349.65034955]
 [291571.0955707 ]
 [506617.7156171 ]
 [998377.62237663]]
```



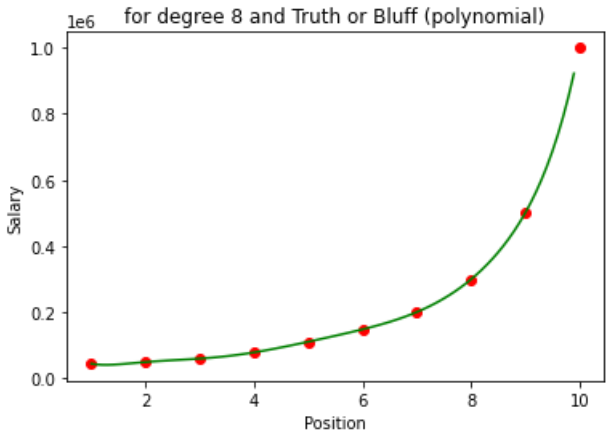

```
[[ 45190.55943885]
 [ 49086.82983563]
 [ 61383.44988144]
 [ 79877.6223754 ]
 [108808.85780719]
 [149282.0512813 ]
 [203940.55944082]
 [295889.27739073]
 [501867.71562078]
 [999673.07692786]]
```



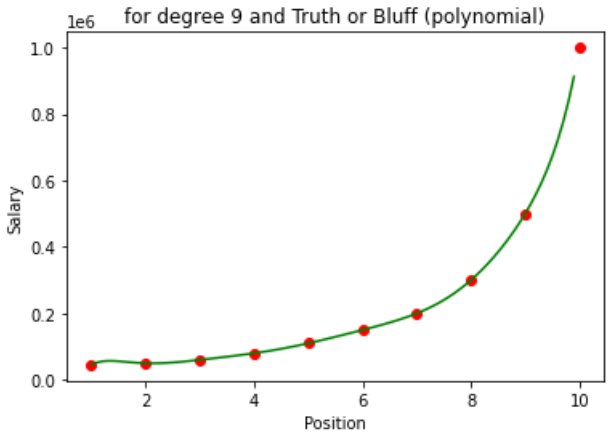

```
[[ 44942.20471817]
 [ 50383.79270358]
 [ 59010.2838458 ]
 [ 81036.6105125 ]
 [110354.17528633]
 [147736.73383964]
 [202781.57135946]
 [298262.44349054]
 [500570.75276409]
 [999921.43148009]]
```



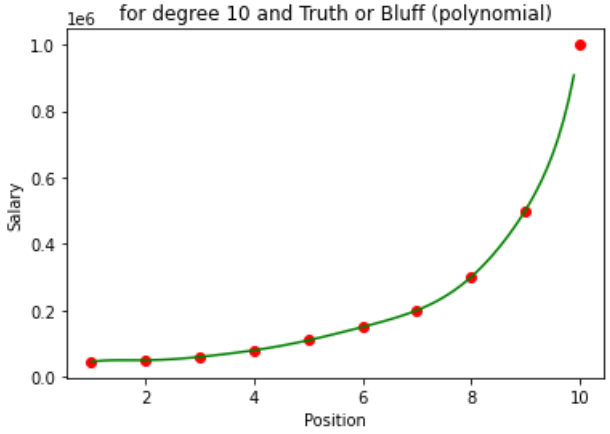

```
[[ 45010.38938324]
 [ 49906.52079728]
 [ 60373.91976565]
 [ 79127.52041862]
 [111308.71946522]
 [148691.27764034]
 [200872.48060863]
 [299626.07973134]
 [500093.4822849 ]
 [999989.60990512]]
```



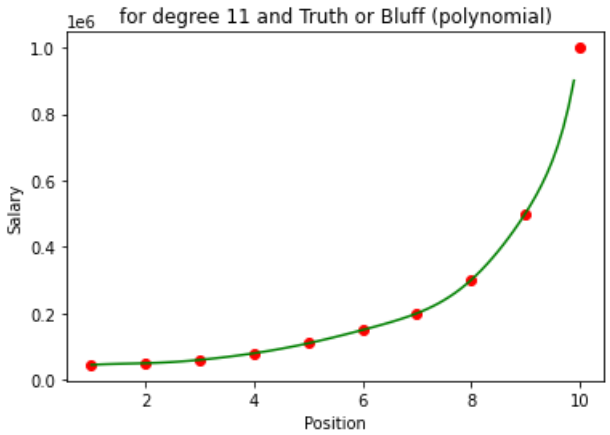

```
[[ 44999.90423478]
 [ 50000.01752543]
 [ 60000.08232445]
 [ 80000.06241003]
 [110000.03370416]
 [150000.0724628 ]
 [200000.03807449]
 [300000.00843906]
 [499999.88833904]
 [999999.89249134]]
```




```
[[ 44999.99989309]
 [ 50000.000408  ]
 [ 60000.0001898 ]
 [ 80000.00010951]
 [110000.00009984]
 [150000.00016969]
 [200000.00015956]
 [299999.99988824]
 [500000.00052911]
 [999999.99854547]]
```




```
[[  44999.99996313]
 [  50000.00043528]
 [  60000.00020107]
 [  80000.00021353]
 [ 110000.00016296]
 [ 150000.00019538]
 [ 199999.9999671 ]
 [ 299999.99960613]
 [ 499999.99911022]
 [1000000.00014782]]
```



In [70]:

```
# here i import r2_score,mean_absolute_error,mean_squared_error from  sklearn.matrics
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error,mean_squared_error
for i in range (1,12):
    poly_reg=PolynomialFeatures(degree=i)# here polynomialFeatures with each degree
    x_poly=poly_reg.fit_transform(x) #
    lin_reg2=LinearRegression()
    models=lin_reg2.fit(x_poly,y)
    y_prd=models.predict(x_poly)
    print(f"RMS= {mean_squared_error(y,y_prd)}  R2(accuracy):{r2_score(y,y_prd)}")
```

```
RMS= 26695878787.878784  R2(accuracy):0.6690412331929895
RMS= 6758833333.333334  R2(accuracy):0.9162082221443942
RMS= 1515662004.6620228  R2(accuracy):0.9812097727913365
RMS= 210343822.84382465  R2(accuracy):0.9973922891706614
RMS= 16382284.382287141  R2(accuracy):0.9997969027099753
RMS= 4075466.200467086  R2(accuracy):0.9999494749253776
RMS= 1854072.3981883056  R2(accuracy):0.9999770143729169
RMS= 524526.9436441854  R2(accuracy):0.9999934972438328
RMS= 0.005208443787466303  R2(accuracy):0.9999999999999354
RMS= 2.698246829267672e-07  R2(accuracy):1.0
RMS= 1.3113537609583035e-07  R2(accuracy):1.0
```

In [73]:

```
model.score(x_poly,y)
```

Out[73]:

```
1.0
```