''' Antes de leer los datos vamos a importar todas las librerias requeridas para realizar la regresion simple Los datos a analizar están en el archivo dolar.csv. Para su lectura usamos el comando read_csv de la libreria pandas '''

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import linear_model
```

In [3]:
```
xls=pd.read_csv("dolar.csv")
xls
```

Out[3]:

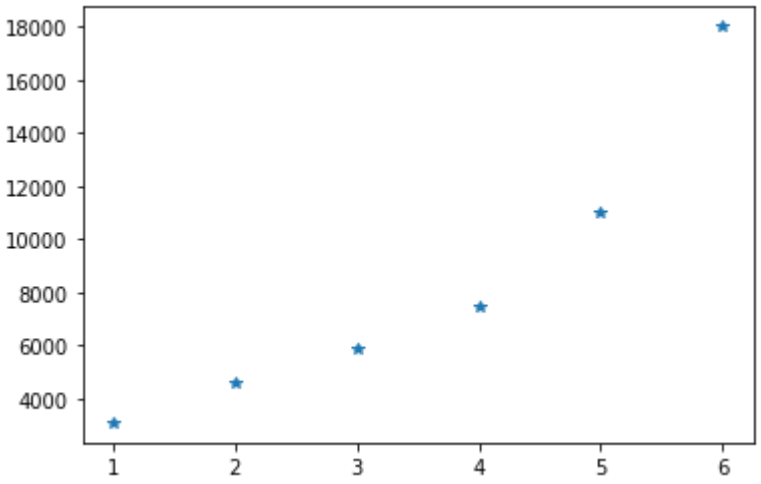
	x	y
0	1	3100
1	2	4629
2	3	5917
3	4	7500
4	5	11000
5	6	18000

In [4]:
```
print(xls.describe()) # Calcula estadisticos basicos
```

```
              x             y
count  6.000000      6.000000
mean   3.500000   8357.666667
std    1.870829   5444.170779
min    1.000000   3100.000000
25%    2.250000   4951.000000
50%    3.500000   6708.500000
75%    4.750000  10125.000000
max    6.000000  18000.000000
```

In [5]:
```
plt.plot(xls.x, xls.y, '*') # Realiza diagrama de dispersion
```

Out[5]:
```
[<matplotlib.lines.Line2D at 0x22f7df9e470>]
```



In [6]:
```
'''
CALCULO DE LA REGRESION LINEAL SIMPLE
'''
x = np.array(xls['x']) # define el vector x requerido para la regresion lineal simple

y = np.array(xls['y']) # define el vector y requerido para la regresion lineal simple

regr = linear_model.LinearRegression() # Invocamos el metodo LinearRegresion de linear_model del sklearn y lo guardamos en regr

regr.fit(x.reshape(-1,1),y.reshape(-1,1)) # Calculamos la regresion lineal para los datos  x,y

 # determinamos  los coeficientes  a y b del modelo y=a+bx
a=regr.intercept_
b=regr.coef_

print(a,b)
```

```
[-1161.93333333] [[2719.88571429]]
```

In [7]:
```
#Valores Predichos
z=regr.predict(x.reshape(-1,1))
print(z)
```

```
[[ 1557.95238095]
 [ 4277.83809524]
 [ 6997.72380952]
 [ 9717.60952381]
 [12437.4952381 ]
 [15157.38095238]]
```

In [8]:
```
'''
RESUMEN DE RESULTADOS'''
import statsmodels.api as sm
results=sm.OLS(y,sm.add_constant(x)).fit()
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.87
4
Model:                            OLS   Adj. R-squared:                  0.84
2
Method:                 Least Squares   F-statistic:                     27.6
4
Date:                Wed, 04 Sep 2019   Prob (F-statistic):           0.0062
6
Time:                        15:57:22   Log-Likelihood:                -53.37
6
No. Observations:                   6   AIC:                             110.
8
Df Residuals:                       4   BIC:                             110.
3
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.97
5]
------------------------------------------------------------------------------
-
const      -1161.9333   2014.697     -0.577      0.595   -6755.628    4431.76
1
x1          2719.8857    517.326      5.258      0.006    1283.558    4156.21
4
==============================================================================
Omnibus:                        nan   Durbin-Watson:                   1.26
5
Prob(Omnibus):                  nan   Jarque-Bera (JB):                0.52
9
Skew:                         0.349   Prob(JB):                        0.76
7
Kurtosis:                     1.723   Cond. No.                         9.3
6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

```
C:\Users\Jesús S\Anaconda3\lib\site-packages\statsmodels\stats\stattools.py:7
1: ValueWarning: omni_normtest is not valid with less than 8 observations; 6
samples were given.
  "samples were given." % int(n), ValueWarning)
```

In [9]:
```
'''CALCULO DE R2 Y MSE'''
from sklearn.metrics import r2_score
print("Mean absolute error: %.2f" % np.mean(np.absolute(y-z)))
print("Residual sum of squares (MSE): %.2f" % np.mean((y - z) ** 2))
print("R2-score: %.2f" % r2_score(y.reshape(-1,1) , z.reshape(-1,1)))
```

```
Mean absolute error: 5551.77
Residual sum of squares (MSE): 46276016.26
R2-score: 0.87
```

In [ ]:
```

```