Importing Libraries

In [466]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sb
import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
```

Reading Data

In [467]:

```
df=pd.read_excel("slr-cyrogenic-flows.xls")
df.head()
```

```
*** No CODEPAGE record, no encoding_override: will use 'ascii'
```

Out[467]:

	X	Y
0	75.1	577.8
1	74.3	577.0
2	88.7	570.9
3	114.6	578.6
4	98.5	572.4

shape of the data(rows and columns)

In [468]:

```
df.shape
```

Out[468]:

```
(30, 2)
```

Statistical values of data

In [469]:

```
df.describe()
```

Out[469]:

	X	Y
count	30.000000	30.000000
mean	90.273333	514.963333
std	16.986078	39.535096
min	62.200000	406.700000
25%	75.300000	505.250000
50%	89.150000	510.100000

	X	Y
75%	104.375000	519.850000
max	120.000000	578.600000

Checking for null values

In [470]:

```
df.isnull().sum()
```

Out[470]:

```
X    0
Y    0
dtype: int64
```

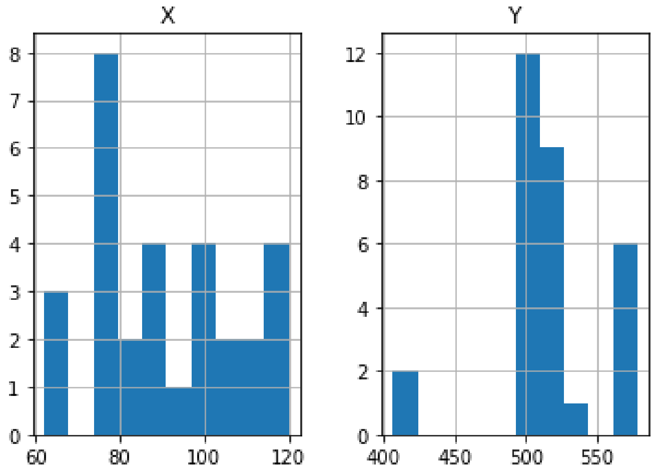
Histogram

In [471]:

```
df.hist()
```

Out[471]:

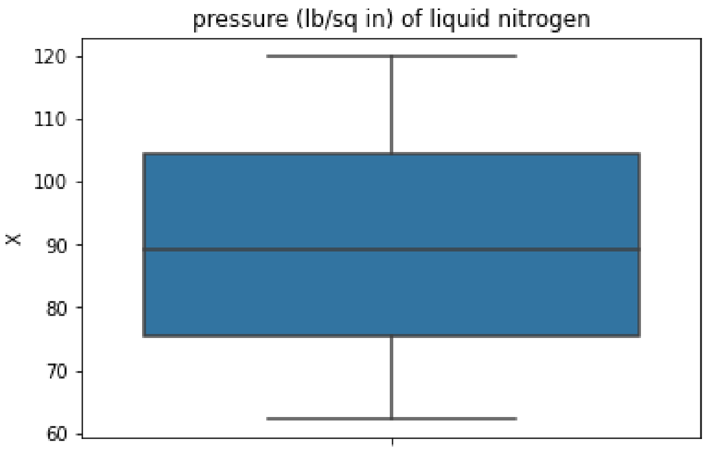
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000002360918A7C0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x0000023608CBCAF0>]],
      dtype=object)
```



Boxplot

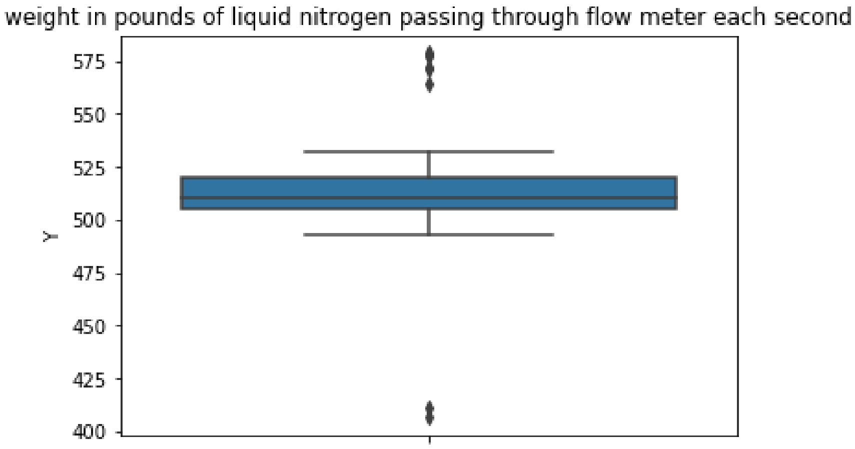
In [472]:

```
plt.figure()
plt.title("pressure (lb/sq in) of liquid nitrogen")
sb.boxplot(y=df["X"],data=df)
plt.show()
```



In [473]:

```
plt.figure()
plt.title("weight in pounds of liquid nitrogen passing through flow meter each second")
sb.boxplot(y=df["Y"],data=df)
plt.show()
```



Outlier detection and handeling

In [474]:

```
q1=np.quantile(df["Y"],.25)
print("Q1 is",q1)
q3=np.quantile(df["Y"],.75)
print("Q3 is",q3)
interqar=q3-q1
print("Interquartile range ",interqar)
lowerb=q1-(interqar*1.5)
print("Lower bound ",lowerb)
upperb=q3+(interqar*1.5)
print("Upper bound",upperb)
```

```
Q1 is 505.25
Q3 is 519.85
Interquartile range  14.600000000000023
Lower bound  483.34999999999997
Upper bound 541.75
```

In [475]:

```
outlier =[]
for x in df["Y"]:
    if ((x> upperb) or (x<lowerb)):
        outlier.append(x)
if outlier==[]:
    print("No outlier present")
else:
    print(outlier)
```

```
[577.8, 577.0, 570.9, 578.6, 572.4, 411.2, 563.9, 406.7]
```

In [476]:

```
df["Y"].median()
```

Out[476]:

```
510.1
```

In [477]:

```
df["Y"]=df["Y"].replace([577.8, 577.0, 570.9, 578.6, 572.4, 411.2, 563.9, 406.7],510.1)
df.head()
```

Out[477]:

	X	Y
0	75.1	510.1
1	74.3	510.1
2	88.7	510.1
3	114.6	510.1
4	98.5	510.1

In [478]:

```
outlier2 =[]
for x in df["Y"]:
    if ((x>upperb) or (x<lowerb)):
         outlier2.append(x)
if outlier2==[]:
    print("No outlier Present")
else:
    print(outlier2)
```

```
No outlier Present
```

Corelation and Heatmap

In [479]:

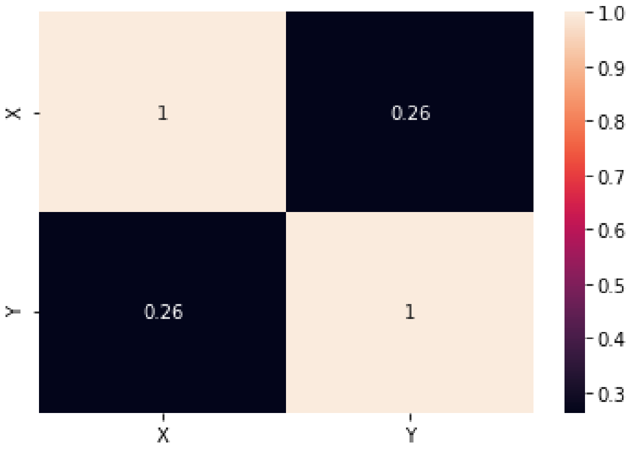
```
corr=df.corr()
corr
```

Out[479]:

	X	Y
X	1.000000	0.262252
Y	0.262252	1.000000

In [480]:

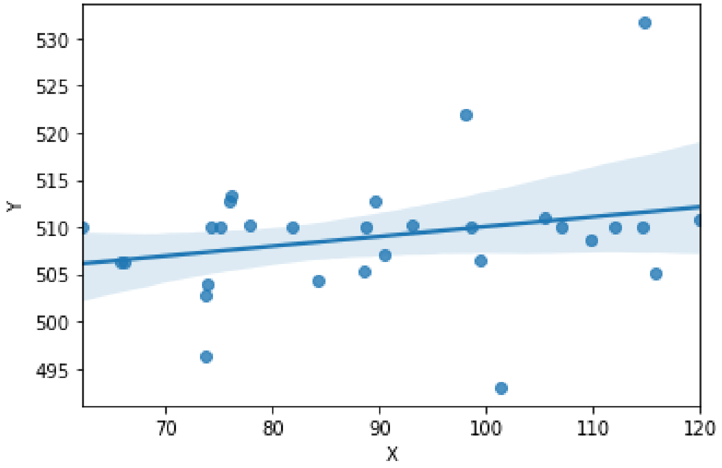
```
plt.figure()
sb.heatmap(corr,annot=True)
plt.show()
```



Scatter plot

In [481]:

```
plt.figure()
plt.title("pressure (lb/sq in) of liquid nitrogen VS weight in pounds of liquid nitrogen
passing through flow meter each second")
sb.regplot(x="X",y="Y",data=df)
plt.show()
```



pressure (lb/sq in) of liquid nitrogen VS weight in pounds of liquid nitrogen passing through flow meter each second

Summary

In [495]:

```
X=sm.add_constant(df["X"])
y=df["Y"]
a=sm.OLS(y,X)
lrmodel=a.fit()
print(lrmodel.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.069
Model:                            OLS   Adj. R-squared:                  0.036
Method:                 Least Squares   F-statistic:                     2.068
Date:                Sat, 08 May 2021   Prob (F-statistic):              0.162
Time:                        00:52:05   Log-Likelihood:                -98.117
No. Observations:                  30   AIC:                             200.2
Df Residuals:                      28   BIC:                             203.0
Df Model:                           1                                         
Covariance Type:            nonrobust                                         
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        499.6824      6.618     75.508      0.000     486.127     513.238
X              0.1037      0.072      1.438      0.162      -0.044       0.251
==============================================================================
Omnibus:                        8.443   Durbin-Watson:                   1.526
Prob(Omnibus):                  0.015   Jarque-Bera (JB):               11.698
Skew:                           0.450   Prob(JB):                      0.00288
Kurtosis:                       5.924   Cond. No.                         505.
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
```

Note

- **We can't Perform Linear regression on the given data set , because Adj. R-squared: 0.071 which is less than 0.65.**
    - **As mention in the description**
- **Find weight in pounds of liquid nitrogen passing through flow meter each second when pressure is60, 70, 80, 90, 100, 110 and 120**

In [496]:

```
X=df["X"].values.reshape(-1,1)
y=df["Y"].values
```

In [497]:

```
# ### Linear Regression Model
model=LinearRegression()
model.fit(X,y)
```

Out[497]:

```
LinearRegression()
```

In [498]:

```
predict=model.predict(X)
df["predict"]=predict
```

In [499]:

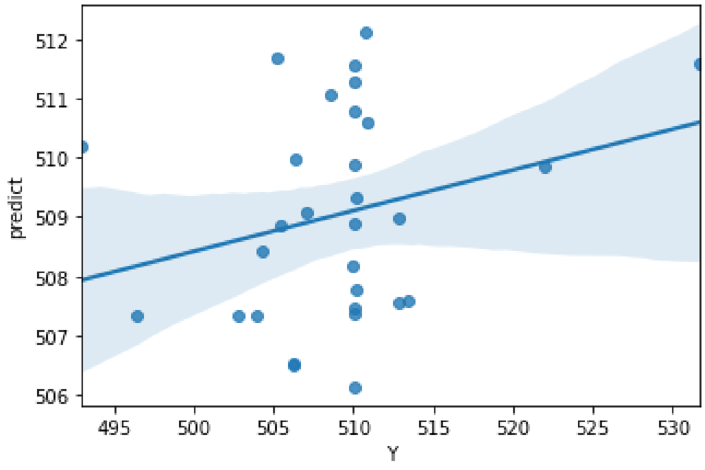
```
df.head()
```

Out[499]:

	X	Y	predict
0	75.1	510.1	507.467151
1	74.3	510.1	507.384224
2	88.7	510.1	508.876910
3	114.6	510.1	511.561673
4	98.5	510.1	509.892766

In [500]:

```
plt.figure()
sb.regplot(x="Y",y="predict",data=df)
plt.show()
```



In [501]:

```
mae = mean_absolute_error(df["Y"], df['predict'])
print(mae)
```

```
4.2242746041896355
```

In [502]:

```
mse = mean_squared_error(df["Y"], df['predict'])
print(mse)
```

```
40.57814586388558
```

In [503]:

In [503]:

```
rmse = np.sqrt(mse)
print(rmse)
```

```
6.370097790763151
```

In [504]:

```
print(df["Y"].mean())
print(df['predict'].mean())
```

```
509.03999999999985
509.03999999999996
```

In [505]:

```
scatterin = rmse/df["Y"].mean()
print(scatterin)
```

```
0.012513943483347386
```

In [506]:

```
find= np.array([[60],[70],[80],[90],[100],[110],[120]])
print(find.flatten())
```

```
[ 60  70  80  90 100 110 120]
```

In [507]:

```
prd2 = model.predict(find)
print(prd2)
```

```
[505.90190283 506.93849075 507.97507868 509.0116666  510.04825452
 511.08484244 512.12143036]
```

In [ ]:

```

```