SIMPLE LINEAR REGRESSION USING ANALYTICAL METHOD

```
import pandas as pd

# Reading csv file from github repo
advertising = pd.read_csv('tvmarketing.csv')


# Display the first 5 rows
advertising.head()
```

	TV	Sales
0	230.1	22.1
1	44.5	10.4
2	17.2	9.3
3	151.5	18.5
4	180.8	12.9

```
# Let's check the columns
advertising.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   TV      200 non-null    float64
 1   Sales   200 non-null    float64
dtypes: float64(2)
memory usage: 3.2 KB
```

```
# Check the shape of the DataFrame (rows, columns)
advertising.shape
```

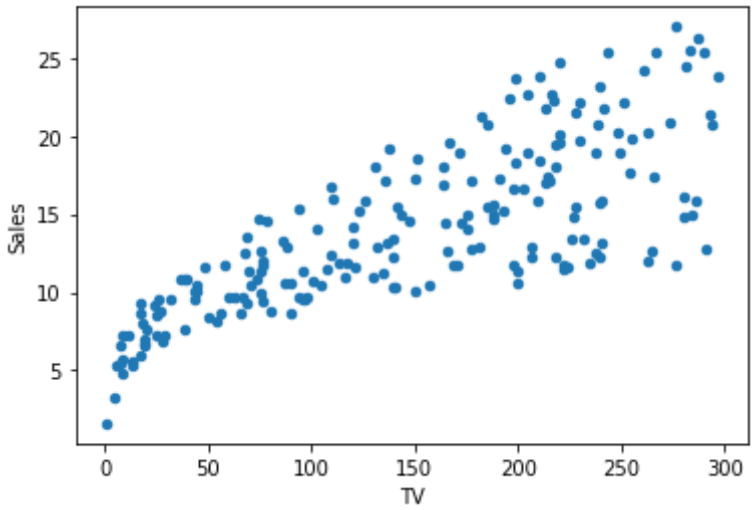
```
(200, 2)
```

```
# Let's look at some statistical information about the dataframe.
advertising.describe()
```

	TV	Sales
count	200.000000	200.000000
mean	147.042500	14.022500
std	85.854236	5.217457
min	0.700000	1.600000
25%	74.375000	10.375000

```
# Visualise the relationship between the features and the response using scatterplots
advertising.plot(x='TV',y='Sales',kind='scatter')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fbcb4d4d690>
```




```
# Putting feature variable to X
x = advertising['TV']

# Print the first 5 rows
x.head()
```

```
0    230.1
1     44.5
2     17.2
3    151.5
4    180.8
Name: TV, dtype: float64
```

```
# Putting response variable to y
y = advertising['Sales']

# Print the first 5 rows
y.head()
```

```
0    22.1
1    10.4
2     9.3
3    18.5
4    12.9
Name: Sales, dtype: float64
```

```
#random_state is the seed used by the random number generator, it can be any integer.

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7 , random_state=00


print(type(x_train))
print(type(x_test))
print(type(y_train))
print(type(y_test))
```

```
<class 'pandas.core.series.Series'>
<class 'pandas.core.series.Series'>
<class 'pandas.core.series.Series'>
<class 'pandas.core.series.Series'>
```

```
train_test_split
```

```
<function sklearn.model_selection._split.train_test_split(*arrays, test_size=None,
train_size=None, random_state=None, shuffle=True, stratify=None)>
```

```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(140,)
(140,)
(60,)
(60,)
```

```
def linear_regression(x, y):
    N = len(x)
    x_mean = x.mean()
    y_mean = y.mean()

    B1_num = ((x - x_mean) * (y - y_mean)).sum()
    B1_den = ((x - x_mean)**2).sum()
    B1 = B1_num / B1_den

    B0 = y_mean - (B1*x_mean)

    reg_line = 'y = {} + {}x'.format(B0, round(B1, 3))

    return (B0, B1, reg_line)


B0, B1, reg_line = linear_regression(x_train, y_train)
print('Regression Line: ', reg_line)
```

```
Regression Line:  y = 7.310810165411682 + 0.046x
```

```
x_input = int(input())
y_output = B0 + round(B1,3) * x_input
print("y = ",y_output)
```

```
177
y =  15.45281016541168
```