Assignment on Linear Regression: The following table shows the results of a recently conducted study on the correlation of the number of hours spent driving with the risk of developing acute backache. Find the equation of the best fit line for this data

Number of hours spent driving (x)	Risk score on a scale 0-100 (y)
10	95
9	80
2	10
15	50
10	45
16	98
11	38
16	93

In [36]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
```

In [37]:
```
data = {
    "driving_hours": [10, 9, 2, 15, 10, 16, 11, 16],
    "risk_score": [95, 80, 10, 50, 45, 98, 38, 93]
}

df = pd.DataFrame.from_dict(data)
df.head()
```

Out[37]:

	driving_hours	risk_score
0	10	95
1	9	80
2	2	10
3	15	50
4	10	45

In [38]:
```
def plotData(X, y, x_label="driving_score", y_label="risk_score"):
    plt.figure(figsize=(6,6))
    plt.xlabel(x_label)
    plt.ylabel(y_label)
    plt.scatter(X, y)
    return plt
```

In [39]:

```
def plotRegressionLine(X, y, y_pred, x_label="driving_score", y_label="risk_score"):
    plt = plotData(X, y, x_label, y_label)
    plt.plot(X, y_pred, color="red", linewidth=3)
    plt.show()
```

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

In [40]:
```
## Not necessary
def calcR2Score(X, y, y_pred):
    y_mean = y.mean()
    SStot = sum((y-y_mean)**2)
    SSres = sum((y-y_pred)**2)
    r2_score = 1 - (SSres/SStot)
    return r2_score
```

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

In [41]:
```
## Not necessary
def calcCorrelationCoeff(X, y):
    X_mean = X.mean()
    y_mean = y.mean()
    num = sum((X-X_mean)*(y-y_mean))
    den = (sum((X-X_mean)**2)*sum((y-y_mean)**2))**0.5
    coeff = num/den
    return coeff
```

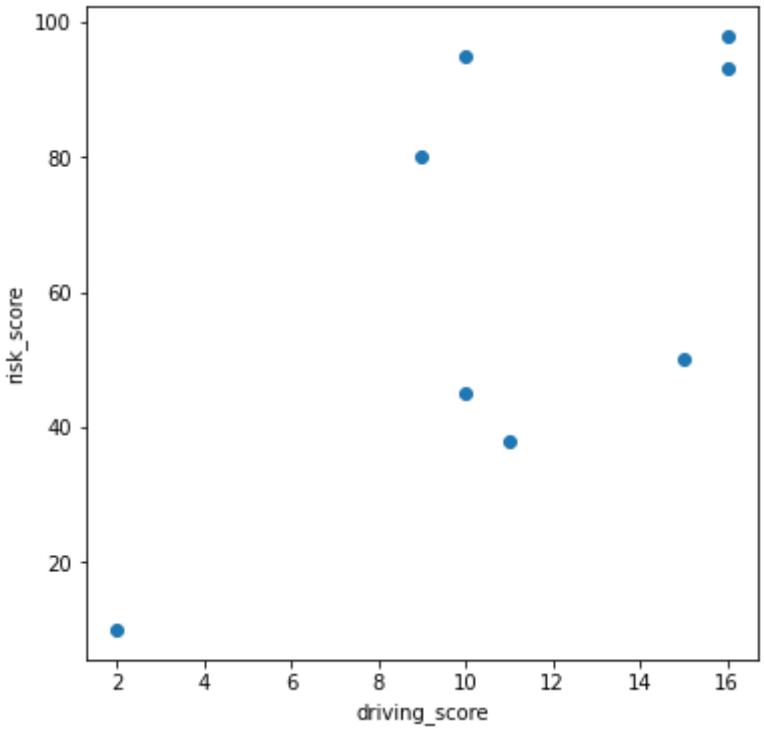
In [42]:
```
X = np.array(df['driving_hours'])
y = np.array(df['risk_score'])
```

In [43]:
```
calcCorrelationCoeff(X, y)
```

Out[43]:
```
0.6611314653759117
```

In [44]:
```
plotData(X, y)
```

Out[44]:
```
<module 'matplotlib.pyplot' from 'D:\\Program_Files\\miniconda3\\lib\\site-packages
\\matplotlib\\pyplot.py'>
```

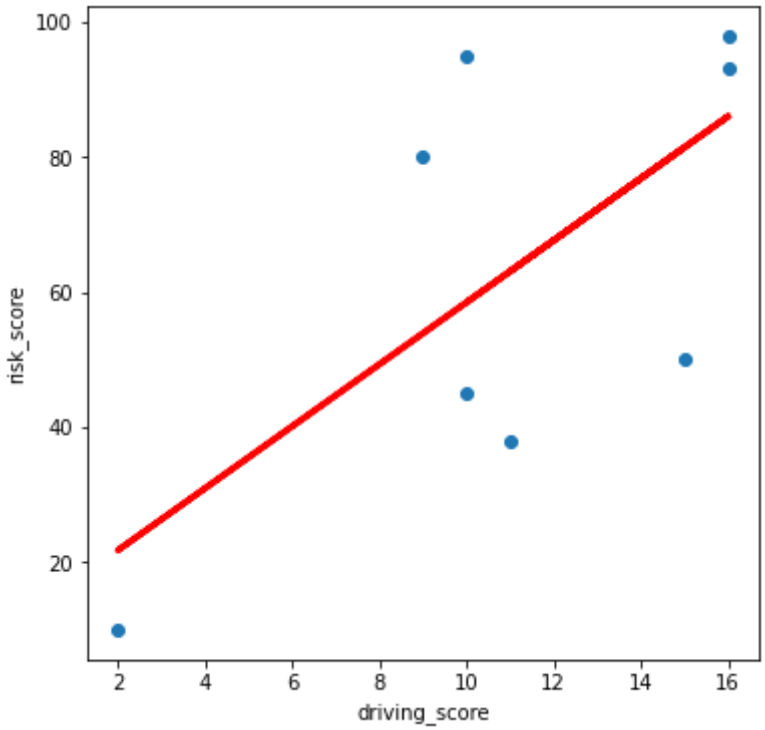


$$\beta_1 = \frac{\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{m}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

In [45]:
```
def calcCoefficients(X, y):
    X_mean = X.mean()
    y_mean = y.mean()
    coeff = sum((X-X_mean)*(y-y_mean))/sum((X-X_mean)**2)
    intercept = y_mean - coeff*X_mean
    return coeff, intercept
```

In [46]:
```
W, W0 = calcCoefficients(X, y)
y_pred = W*X + W0
plotRegressionLine(X, y, y_pred)
```



In [47]:
```
print("Coefficient: ",W)
print("Intercept: ",W0)
print("R2 score: ",calcR2Score(X, y, y_pred))
print(f"Equation: Y = {W}*X+{W0}")
```

```
Coefficient:  4.58789860997547
Intercept:  12.584627964022893
R2 score:  0.43709481451010035
Equation: Y = 4.58789860997547*X+12.584627964022893
```

Sklearn Implementation

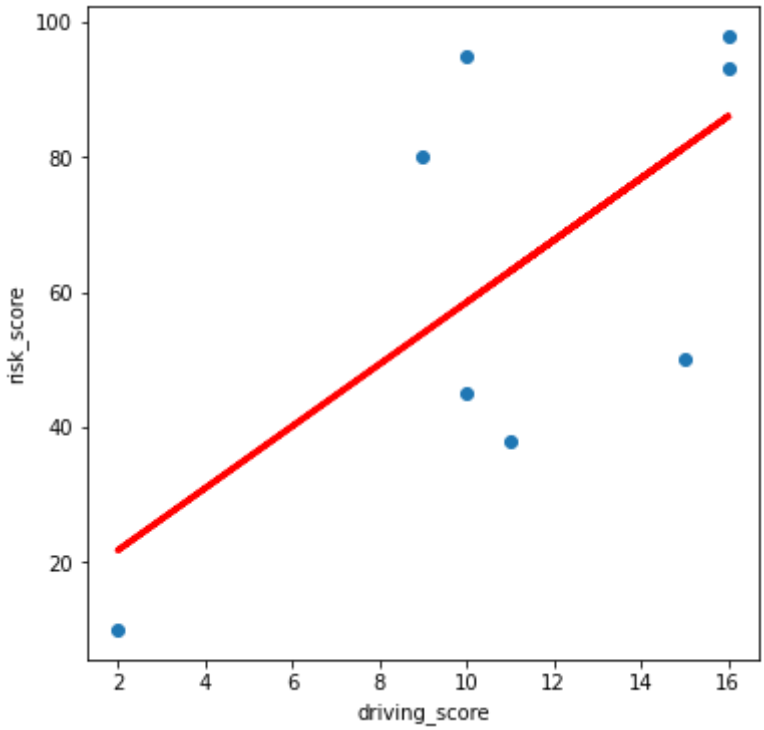
In [48]:
```
X = np.array(X).reshape(-1, 1)
```

In [49]:
```
model = LinearRegression()
model.fit(X, y)
y_pred = model.predict(X)
```

In [50]:
```
print("Coefficient: ",model.coef_[0])
print("Intercept: ",model.intercept_)
print("R2 score: ",calcR2Score(X, y, y_pred))
print(f"Equation: Y = {model.coef_[0]}*X+{model.intercept_}")
```

```
Coefficient:  4.587898609975469
Intercept:  12.584627964022907
R2 score:  0.43709481451010035
Equation: Y = 4.587898609975469*X+12.584627964022907
```

In [51]:
```
plotRegressionLine(X, y, y_pred)
```



In [ ]: