In [46]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [48]:

```
data = pd.read_csv("BostonHousing.csv")
data
```

Out[48]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	396.9
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	396.9
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	392.8
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	394.6
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3	222	18.7	396.9
...	...	...	...	...	...	...	...	...	...	...	...	.
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1	273	21.0	391.9
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1	273	21.0	396.9
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1	273	21.0	396.9
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1	273	21.0	393.4
505	0.04741	0.0	11.93	0.0	0.573	6.030	NaN	2.5050	1	273	21.0	396.9

506 rows × 14 columns

In [50]:

```
print(data.isnull().sum())
```

```
CRIM       20
ZN         20
INDUS      20
CHAS       20
NOX         0
RM          0
AGE        20
DIS         0
RAD         0
TAX         0
PTRATIO     0
B           0
LSTAT      20
MEDV        0
dtype: int64
```

In [52]:

```
data.fillna(data.mean(), inplace=True)
```

In [54]:

```
print(data.isnull().sum())
```

```
CRIM       0
ZN         0
INDUS      0
CHAS       0
NOX        0
RM         0
AGE        0
DIS        0
RAD        0
TAX        0
PTRATIO    0
B          0
LSTAT      0
MEDV       0
dtype: int64
```

In [56]:
```
X = data[['RM']]      # Single independent variable
y = data['MEDV']      # Target variable
```

In [58]:
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

In [60]:
```
single_model = LinearRegression()
single_model.fit(X_train, y_train)
```

Out[60]:

▾ LinearRegression ⓘ ?

▶ Parameters

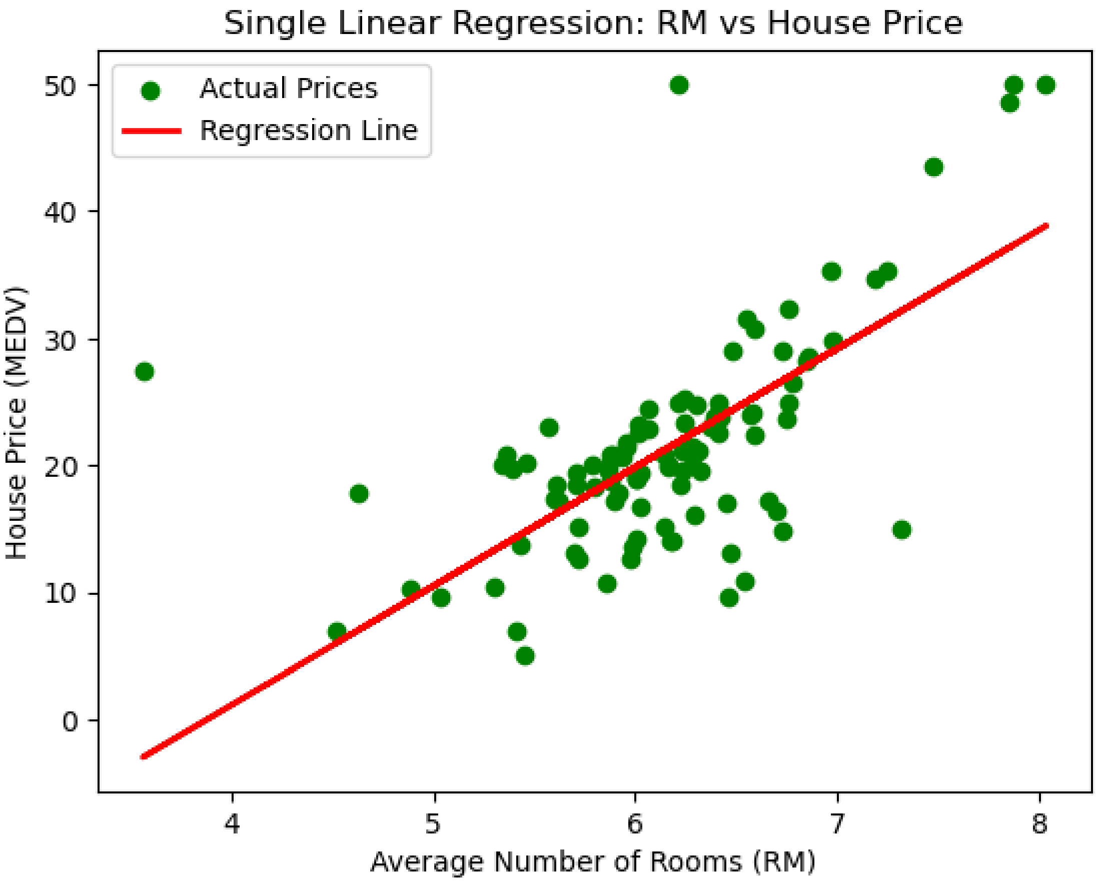
In [62]:
```
print("Intercept:", single_model.intercept_)
print("Coefficient:", single_model.coef_[0])
```

```
Intercept: -36.24631889813795
Coefficient: 9.348301406497727
```

In [64]:
```
y_pred_single = single_model.predict(X_test)

mse_single = mean_squared_error(y_test, y_pred_single)
r2_single = r2_score(y_test, y_pred_single)

print("Mean Squared Error:", mse_single)
print("R² Score:", r2_single)
```

```
Mean Squared Error: 46.144775347317264
R² Score: 0.3707569232254778
```

In [66]:
```
plt.scatter(X_test, y_test, color='green', label='Actual Prices')
plt.plot(X_test, y_pred_single, color='red', linewidth=2, label='Regression Line')
plt.xlabel("Average Number of Rooms (RM)")
plt.ylabel("House Price (MEDV)")
plt.title("Single Linear Regression: RM vs House Price")
plt.legend()
plt.show()
```



In [ ]:

In [69]:
```
X = data.drop('MEDV', axis=1)
y = data['MEDV']
```

In [71]:
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

In [73]:
```
multi_model = LinearRegression()
multi_model.fit(X_train, y_train)
```

Out[73]:
```
▾ LinearRegression ⓘ ?

▶ Parameters
```

In [75]:
```
coefficients = pd.DataFrame({
    'Feature': X.columns,
    'Coefficient': multi_model.coef_
})

print(coefficients)
```

```
    Feature  Coefficient
0      CRIM    -0.114140
1        ZN     0.029016
2     INDUS    -0.027133
3      CHAS     3.240673
4       NOX   -16.017543
5        RM     4.752410
6       AGE    -0.019520
7       DIS    -1.521593
8       RAD     0.222823
9       TAX    -0.009151
10  PTRATIO    -0.886045
11        B     0.012263
12    LSTAT    -0.438440
```

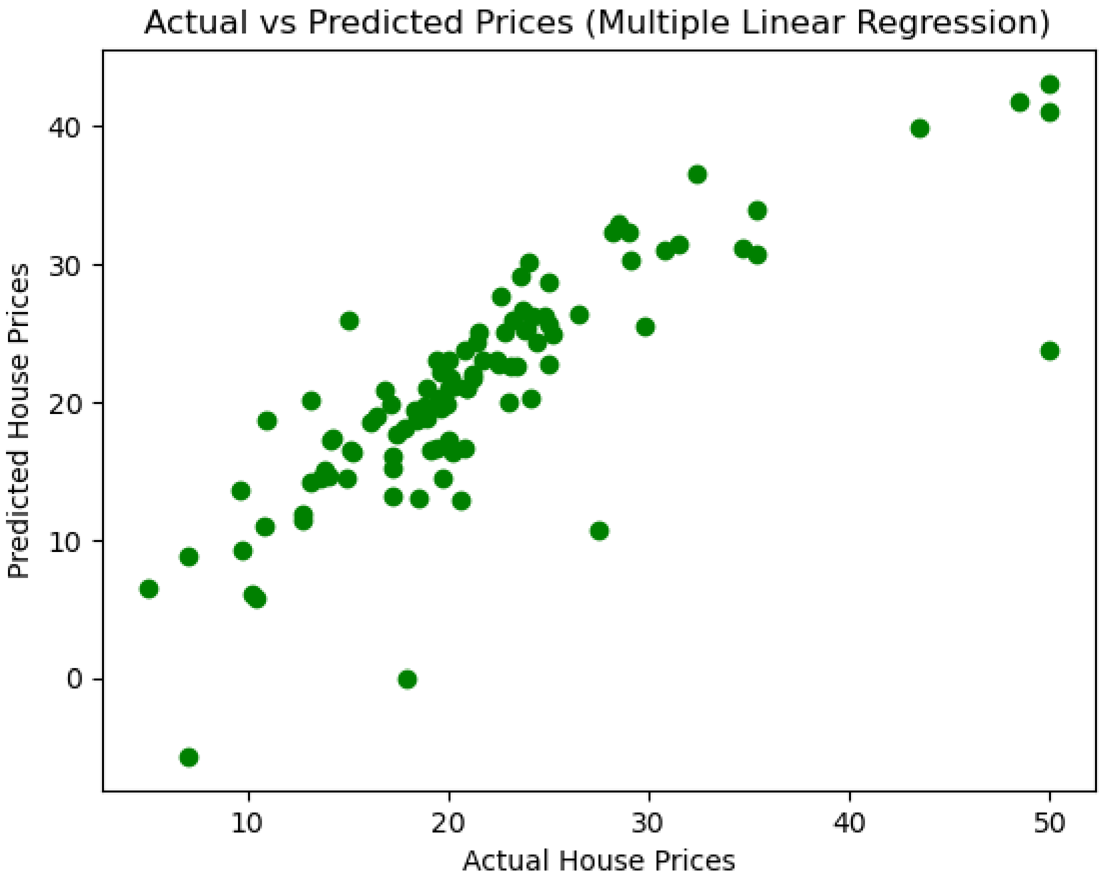
In [77]:

```python
y_pred_multi = multi_model.predict(X_test)

mse_multi = mean_squared_error(y_test, y_pred_multi)
r2_multi = r2_score(y_test, y_pred_multi)

print("Mean Squared Error:", mse_multi)
print("R² Score:", r2_multi)
```

```
Mean Squared Error: 25.017672023842582
R² Score: 0.6588520195508156
```

In [79]:

```python
plt.scatter(y_test, y_pred_multi, color='green')
plt.xlabel("Actual House Prices")
plt.ylabel("Predicted House Prices")
plt.title("Actual vs Predicted Prices (Multiple Linear Regression)")
plt.show()
```



In [ ]: