In [24]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import matplotlib.pyplot as plt
```

In [25]:
```
boston = pd.read_csv('/home/csl-4/Documents/7348/ASSIGNMENT4/BostonHousing.cs
boston.shape
```

Out[25]:
```
(506, 14)
```

In [26]:
```
boston.columns
```

Out[26]:
```
Index(['crim', 'zn', 'indus', 'chas', 'nox', 'rm', 'age', 'dis', 'rad', 'ta
x',
       'ptratio', 'b', 'lstat', 'medv'],
      dtype='object')
```

In [27]:
```
X = boston.drop('medv', axis=1)
y = boston['medv']
```

In [29]:
```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42
)
X_train1, X_test1, y_train1, y_test1 = train_test_split(
    X, y,
    test_size=0.2,
    random_state=92
)
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[29]:
```
▾ LinearRegression
LinearRegression()
```

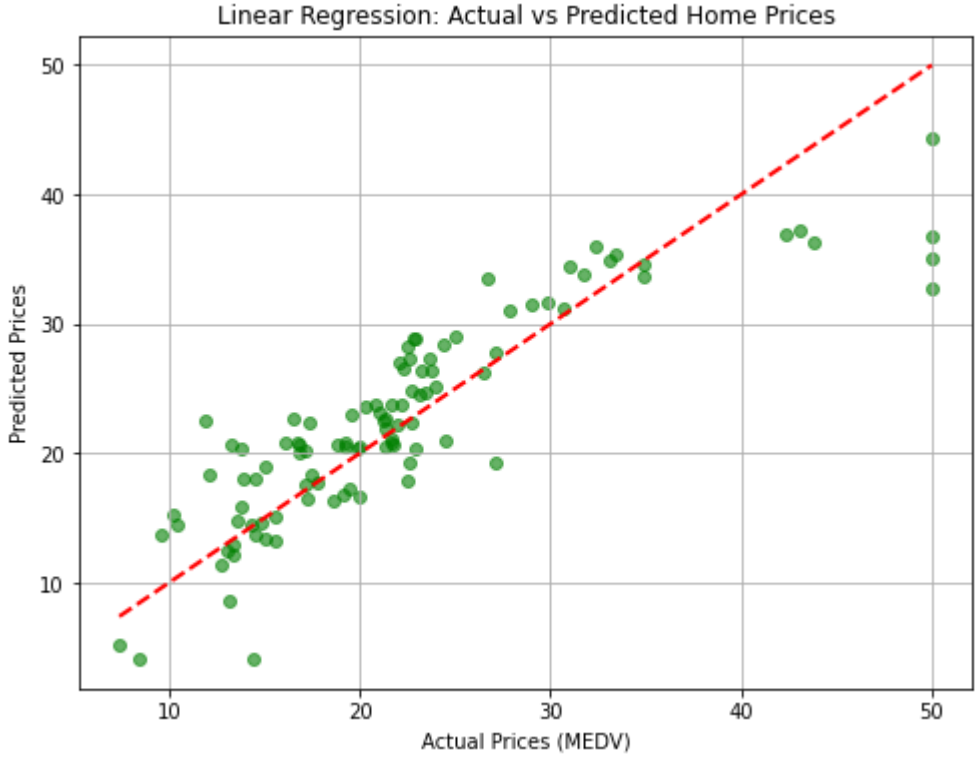
In [30]:
```
y_pred = model.predict(X_test)
y_pred1 = model.predict(X_test1)
plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.6)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', l

plt.xlabel('Actual Prices (MEDV)')
plt.ylabel('Predicted Prices')
plt.title('Linear Regression: Actual vs Predicted Home Prices')
plt.grid(True)
plt.show()
```

In [31]:

```
plt.figure(figsize=(8,6))
plt.scatter(y_test1, y_pred1, color='green', alpha=0.6)
plt.plot([y_test1.min(), y_test1.max()], [y_test1.min(), y_test1.max()], 'r--

plt.xlabel('Actual Prices (MEDV)')
plt.ylabel('Predicted Prices')
plt.title('Linear Regression: Actual vs Predicted Home Prices')
plt.grid(True)
plt.show()
```



In [32]:

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np
```

In [33]:
```
mse = mean_squared_error(y_test, y_pred)

rmse = np.sqrt(mse)

mae = mean_absolute_error(y_test, y_pred)

r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("Mean Absolute Error (MAE):", mae)
print("R² Score:", r2)
```

```
Mean Squared Error (MSE): 24.291119474973478
Root Mean Squared Error (RMSE): 4.928602182665332
Mean Absolute Error (MAE): 3.189091965887837
R² Score: 0.6687594935356326
```

In [34]:
```
selected_features = ['crim', 'indus', 'age', 'rad', 'tax']

X = boston[selected_features]  # Only chosen columns
y = boston['medv']             # Target
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
```

Out[34]:
```
▾ LinearRegression ⓘ ?

LinearRegression()
```

In [35]:
```
y_pred = model.predict(X_test)
```

In [36]:
```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import numpy as np

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error (MSE):", mse)
print("Root Mean Squared Error (RMSE):", rmse)
print("Mean Absolute Error (MAE):", mae)
print("R² Score:", r2)
```

```
Mean Squared Error (MSE): 46.38474398396048
Root Mean Squared Error (RMSE): 6.810634624171266
Mean Absolute Error (MAE): 4.791432034046519
R² Score: 0.36748464370706546
```

In [ ]: