Step 1: Import Required Libraries

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

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

import joblib
```

Step 2: Load Dataset

```
df = pd.read_csv("india_housing_prices.csv.csv")

print(df.head())
print(df.shape)
```

Step 3: Dataset Information

```
print(df.info())
print(df.describe())
```

Step 4: Check Missing Values

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

Step 5: Correlation Heatmap

```
plt.figure(figsize=(10, 7))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```

Step 6: Feature Selection

Target column assumed as `Price`

```
X = df.drop('Price', axis=1)
y = df['Price']
```

Step 7: Train-Test Split

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

Step 8: Create Pipeline (Scaling + Model)

```
model = Pipeline([
    ('scaler', StandardScaler()),
    ('lr', LinearRegression())
])
```

Step 9: Train Model

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

Step 10: Prediction

```
y_pred = model.predict(X_test)
```

Step 11: Model Evaluation

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

print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r2)
```

Step 12: Actual vs Predicted Plot

```
plt.figure(figsize=(7,6))
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs Predicted Prices")
plt.grid(True)
plt.show()
```

Step 13: Residual Plot

```
residuals = y_test - y_pred
```

```
plt.figure(figsize=(7,6))
plt.scatter(y_test, residuals)
plt.axhline(y=0, linestyle='--')
plt.xlabel("Actual Prices")
plt.ylabel("Residuals")
plt.title("Residual Plot")
plt.grid(True)
plt.show()
```

Step 14: Save Trained Model

```
joblib.dump(model, "house_price_model.pkl")
print("Model Saved Successfully")
```

Step 15: Load Saved Model (When Needed)

```
model = joblib.load("house_price_model.pkl")
print("Model Loaded Successfully")
```

Step 16: Predict New House Price

Example Input (Change Values As Needed)

```
sample_data = pd.DataFrame([[1200, 3, 2, 1]], columns=X.columns)

predicted_price = model.predict(sample_data)

print("Predicted Price:", predicted_price[0])
```
