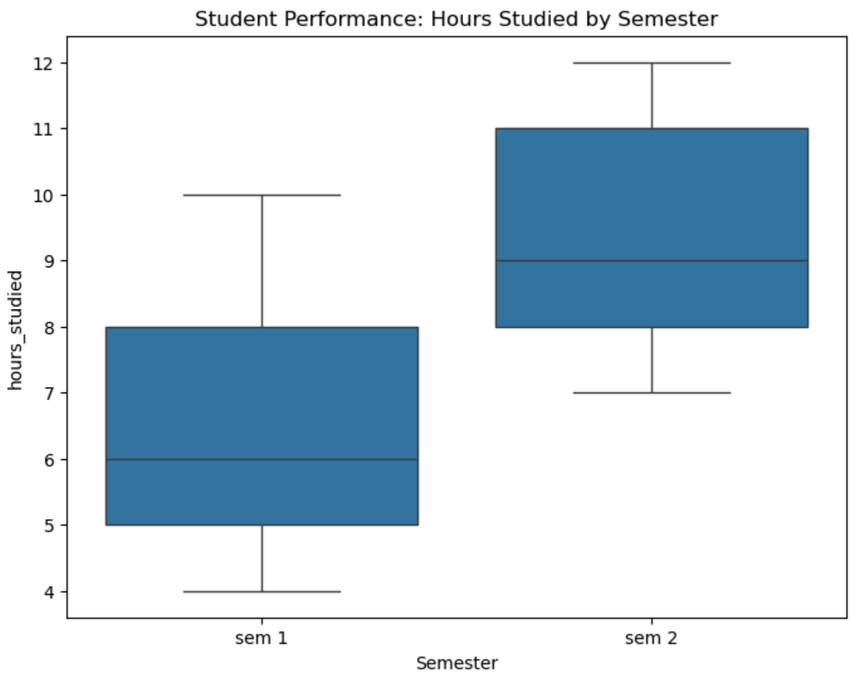
In [2]:
```
#ss


import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Sample data
data = {
    'semester': ['sem 1', 'sem 1', 'sem 1', 'sem 1', 'sem 1', 'sem 2', 'sem 2', 'sem 2', 'sem 2', 'sem 2'],
    'hours_studied': [5, 8, 10, 4, 6, 9, 11, 7, 12, 8]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Create the boxplot
plt.figure(figsize=(8, 6))
sns.boxplot(x='semester', y='hours_studied', data=df)

# Set plot labels and title
plt.title('Student Performance: Hours Studied by Semester')
plt.xlabel('Semester')
plt
```

Out[2]:
```
<module 'matplotlib.pyplot' from 'C:\\Users\\gundr\\anaconda3\\Lib\\site-packages\\matplotlib\\pyplot.py'>
```



In [4]:
```
import pandas as pd

# Sample data with duplicates
data = {
    'Name': ['Alice', 'Bob', 'Alice', 'Charlie', 'Bob'],
    'Age': [25, 30, 25, 35, 30],
    'City': ['hyd', 'goa', 'bangalore', 'hyd', 'chennai']
}

# Create DataFrame
df = pd.DataFrame(data)

# Display the original DataFrame
print("Original DataFrame:")
print(df)

# Identify duplicate rows
duplicates = df[df.duplicated()]

print("\nIdentified Duplicates:")
print(duplicates)

# Remove duplicates, keeping the first occurrence
df_cleaned = df.drop_duplicates()

# Display the DataFrame after removing duplicates
print("\nDataFrame After Removing Duplicates:")
print(df_cleaned)
```

```
Original DataFrame:
      Name  Age       City
0    Alice   25        hyd
1      Bob   30        goa
2    Alice   25  bangalore
3  Charlie   35        hyd
4      Bob   30    chennai

Identified Duplicates:
Empty DataFrame
Columns: [Name, Age, City]
Index: []

DataFrame After Removing Duplicates:
      Name  Age       City
0    Alice   25        hyd
1      Bob   30        goa
2    Alice   25  bangalore
3  Charlie   35        hyd
4      Bob   30    chennai
```

In [6]:
```
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer

# Create a sample dataset (House Price Dataset with missing values)
data = {
    'square_feet_area': [8500, 9600, np.nan, 11250, np.nan, 9550, 14260, np.nan, 13830, 11500],
    'Year_built': [2003, 1976, 2001, np.nan, 1998, 2000, 2006, 1978, 1950, np.nan],
    'over_all_condition': [5, 8, 6, 7, np.nan, 7, 8, 6, np.nan, 7],
    'ready_to_move': ['Yes', 'No', 'Yes', np.nan, 'No', np.nan, 'Yes', 'Yes', 'No', 'Yes'],
    'Sale_price': [200000, 180000, 215000, 250000, 210000, 190000, 230000, 225000, 220000, 240000]
}

# Create DataFrame
df = pd.DataFrame(data)

# Print original dataset with missing values
print("Original Dataset with Missing Values:")
print(df)

# 1. Impute Missing Values in Numeric Columns (Mean)

# Select numeric columns
numeric_cols = df.select_dtypes(include=[np.number]).columns

# Mean Imputation for Numeric Features
imputer_mean = SimpleImputer(strategy='mean')

# Apply mean imputation to numeric columns
df[numeric_cols] = imputer_mean.fit_transform(df[numeric_cols])

# 2. Impute Missing Values in Categorical Columns (Mode)

# Select categorical columns
categorical_cols = df.select_dtypes(include=[object]).columns

# Mode Imputation for Categorical Features
imputer_mode = SimpleImputer(strategy='most_frequent')

# Apply mode imputation to categorical columns
df[categorical_cols] = imputer_mode.fit_transform(df[categorical_cols])

# Print the dataset after imputation
print("\nDataset after Mean, Median, and Mode Imputation:")
print(df)
```

```
Original Dataset with Missing Values:
   square_feet_area  Year_built  over_all_condition ready_to_move  Sale_price
0            8500.0      2003.0                 5.0           Yes      200000
1            9600.0      1976.0                 8.0            No      180000
2               NaN      2001.0                 6.0           Yes      215000
3           11250.0         NaN                 7.0           NaN      250000
4               NaN      1998.0                 NaN            No      210000
5            9550.0      2000.0                 7.0           NaN      190000
6           14260.0      2006.0                 8.0           Yes      230000
7               NaN      1978.0                 6.0           Yes      225000
8           13830.0      1950.0                 NaN            No      220000
9           11500.0         NaN                 7.0           Yes      240000

Dataset after Mean, Median, and Mode Imputation:
   square_feet_area  Year_built  over_all_condition ready_to_move  Sale_price
0       8500.000000      2003.0               5.00           Yes    200000.0
1       9600.000000      1976.0               8.00            No    180000.0
2      11212.857143      2001.0               6.00           Yes    215000.0
3      11250.000000      1989.0               7.00           Yes    250000.0
4      11212.857143      1998.0               6.75            No    210000.0
5       9550.000000      2000.0               7.00           Yes    190000.0
6      14260.000000      2006.0               8.00           Yes    230000.0
7      11212.857143      1978.0               6.00           Yes    225000.0
```

8	13830.000000	1950.0	6.75	No	220000.0
9	11500.000000	1989.0	7.00	Yes	240000.0

In [ ]: