EDS ASSIGNMENT-1

Name: Himanshu Manoj Gharde

Division: CS7

Roll No: CS7-67

PRN: 202401110007

Google Collab: https://colab.research.google.com/drive/1fDDBEMDDrOfOrljYk8-iTilyiq402XVv?usp=drive_link

Dataset: House Price

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	date	price	bedrooms	bathroom	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_baser	yr_built	yr_renova	street	city	statezip	country
2	########	313000	3	1.5	1340	7912	1.5	0	0	3	1340	0	1955	2005	18810 Der	Shoreline	WA 98133	USA
3	########	2384000	5	2.5	3650	9050	2	0	4	5	3370	280	1921	0	709 W Bla	Seattle	WA 9811	USA
4	########	342000	3	2	1930	11947	1	0	0	4	1930	0	1966	0	26206-26	Kent	WA 98042	USA
5	########	420000	3	2.25	2000	8030	1	0	0	4	1000	1000	1963	0	857 170th	Bellevue	WA 98008	USA
6	########	550000	4	2.5	1940	10500	1	0	0	4	1140	800	1976	1992	9105 170t	Redmond	WA 98052	USA
7	########	490000	2	1	880	6380	1	0	0	3	880	0	1938	1994	522 NE 88	Seattle	WA 9811	USA
8	########	335000	2	2	1350	2560	1	0	0	3	1350	0	1976	0	2616 174t	Redmond	WA 98052	USA
9	########	482000	4	2.5	2710	35868	2	0	0	3	2710	0	1989	0	23762 SE	Maple Val	WA 98038	USA
10	########	452500	3	2.5	2430	88426	1	0	0	4	1570	860	1985	0	46611-46	North Ben	WA 98045	USA
11	########	640000	4	2	1520	6200	1.5	0	0	3	1520	0	1945	2010	6811 55th	Seattle	WA 9811	USA
12	########	463000	3	1.75	1710	7320	1	0	0	3	1710	0	1948	1994	Burke-Gil	Lake Fores	WA 98155	USA
13	########	1400000	4	2.5	2920	4000	1.5	0	0	5	1910	1010	1909	1988	3838-409	Seattle	WA 9810	USA
14	########	588500	3	1.75	2330	14892	1	0	0	3	1970	360	1980	0	1833 220t	Sammami	WA 98074	USA
15	########	365000	3	1	1090	6435	1	0	0	4	1090	0	1955	2009	2504 SW	Seattle	WA 9810	USA
16	########	1200000	5	2.75	2910	9480	1.5	0	0	3	2910	0	1939	1969	3534 46th	Seattle	WA 9810	USA
17	########	242500	3	1.5	1200	9720	1	0	0	4	1200	0	1965	0	14034 SE	Kent	WA 98042	USA
18	########	419000	3	1.5	1570	6700	1	0	0	4	1570	0	1956	0	15424 SE	Bellevue	WA 98007	USA
19	########	367500	4	3	3110	7231	2	0	0	3	3110	0	1997	0	11224 SE	Auburn	WA 98092	USA
20	########	257950	3	1.75	1370	5858	1	0	0	3	1370	0	1987	2000	1605 S 24	Des Moine	WA 98198	USA
21	########	275000	3	1.5	1180	10277	1	0	0	3	1180	0	1983	2009	12425 41	North Ben	WA 98045	USA
22	########	750000	3	1.75	2240	10578	2	0	0	5	1550	690	1923	0	3225 NE 9	Seattle	WA 9811	USA

Importing Pandas And num.py in python

```
[1] import pandas as pd
    import numpy as np

[2] from google.colab import files
    uploaded = files.upload()

    Choose Files  data.csv
    • data.csv(text/csv) - 526795 bytes, last modified: 4/28/2025 - 100% done
    Saving data.csv to data.csv

    df = pd.read_csv('data.csv')
```

```
df.head()
df.info()
df.describe()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4600 entries, 0 to 4599
Data columns (total 18 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           4600 non-null   object
 1   price          4600 non-null   float64
 2   bedrooms       4600 non-null   float64
 3   bathrooms      4600 non-null   float64
 4   sqft_living    4600 non-null   int64
 5   sqft_lot       4600 non-null   int64
 6   floors         4600 non-null   float64
 7   waterfront     4600 non-null   int64
 8   view           4600 non-null   int64
 9   condition      4600 non-null   int64
 10  sqft_above     4600 non-null   int64
 11  sqft_basement  4600 non-null   int64
 12  yr_built       4600 non-null   int64
 13  yr_renovated   4600 non-null   int64
 14  street         4600 non-null   object
 15  city           4600 non-null   object
 16  statezip       4600 non-null   object
 17  country        4600 non-null   object
dtypes: float64(4), int64(9), object(5)
memory usage: 647.0+ KB
```

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condition	sqft_above	sqft_basement	yr_built	yr_renovated
count	4.600000e+03	4600.000000	4600.000000	4600.000000	4.600000e+03	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000	4600.000000
mean	5.519630e+05	3.400870	2.160815	2139.346957	1.485252e+04	1.512065	0.007174	0.240652	3.451739	1827.265435	312.081522	1970.786304	808.608261
std	5.638347e+05	0.908848	0.783781	963.206916	3.588444e+04	0.538288	0.084404	0.778405	0.677230	862.168977	464.137228	29.731848	979.414536
min	0.000000e+00	0.000000	0.000000	370.000000	6.380000e+02	1.000000	0.000000	0.000000	1.000000	370.000000	0.000000	1900.000000	0.000000
25%	3.228750e+05	3.000000	1.750000	1460.000000	5.000750e+03	1.000000	0.000000	0.000000	3.000000	1190.000000	0.000000	1951.000000	0.000000
50%	4.609435e+05	3.000000	2.250000	1980.000000	7.683000e+03	1.500000	0.000000	0.000000	3.000000	1590.000000	0.000000	1976.000000	0.000000
75%	6.549625e+05	4.000000	2.500000	2620.000000	1.100125e+04	2.000000	0.000000	0.000000	4.000000	2300.000000	610.000000	1997.000000	1999.000000
max	2.659000e+07	9.000000	8.000000	13540.000000	1.074218e+06	3.500000	1.000000	4.000000	5.000000	9410.000000	4820.000000	2014.000000	2014.000000

Pandas :

1. Find the total number of rows and columns (shape).

Pandas :

1. Find the total number of rows and columns (shape).

```
[5] shape = df.shape
    print("Shape of dataset:", shape)

    Shape of dataset: (4600, 18)
```

2. Check for any missing values.

2. Check for any missing values.

```
[6] missing_values = df.isnull().sum()
    print("Missing values in each column:\n", missing_values)
```

```
Missing values in each column:
 date             0
price             0
bedrooms          0
bathrooms         0
sqft_living       0
sqft_lot          0
floors            0
waterfront        0
view              0
condition         0
sqft_above        0
sqft_basement     0
yr_built          0
yr_renovated      0
street            0
city              0
statezip          0
country           0
dtype: int64
```

3. Find the average price.

3. Find the average price.

```
[7] average_price = df['price'].mean()
    print("Average price:", average_price)
```

```
Average price: 551962.9884732141
```

4. Find the maximum number of bedrooms.

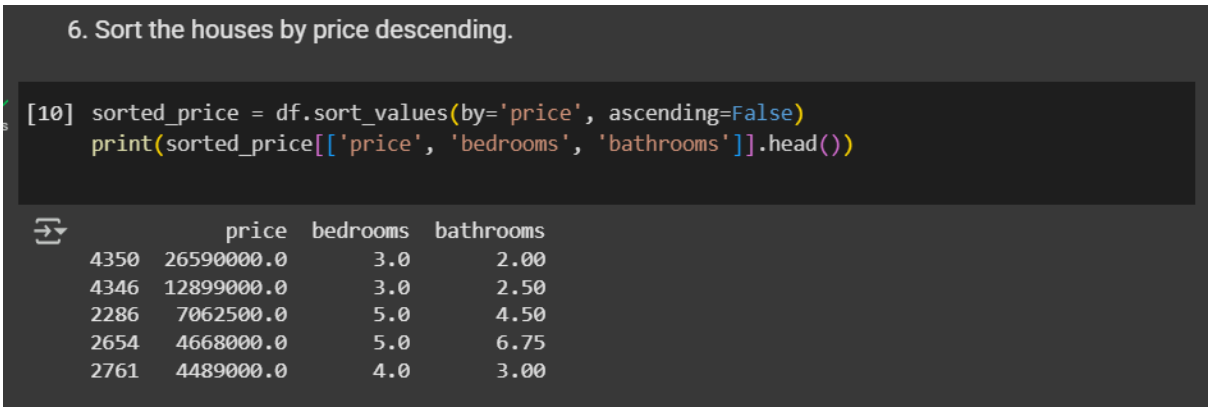
4. Find the maximum number of bedrooms.

Generate | a slider using jupyter widgets | Close

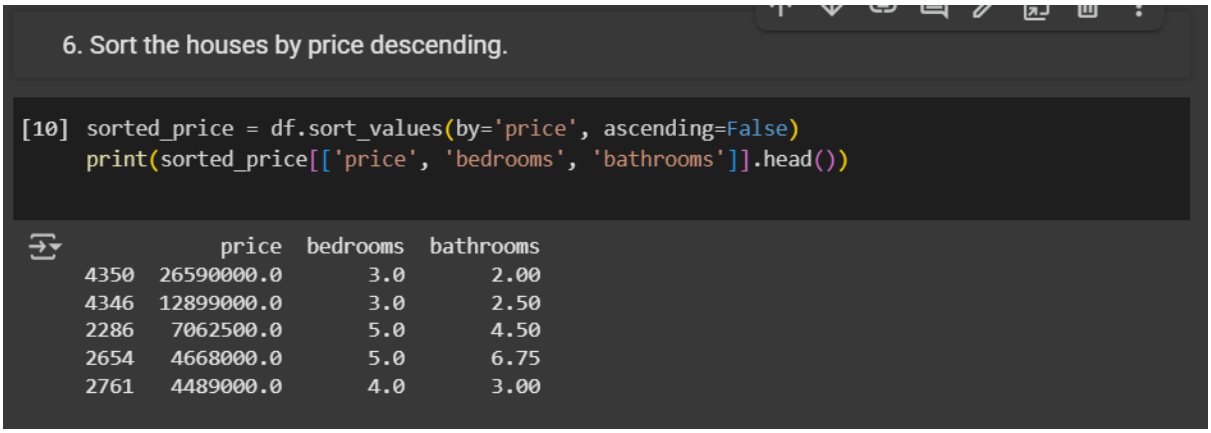
```
[8] max_bedrooms = df['bedrooms'].max()
    print("Maximum bedrooms:", max_bedrooms)
```

```
Maximum bedrooms: 9.0
```

5. Find the minimum number of bathrooms. python Copy code

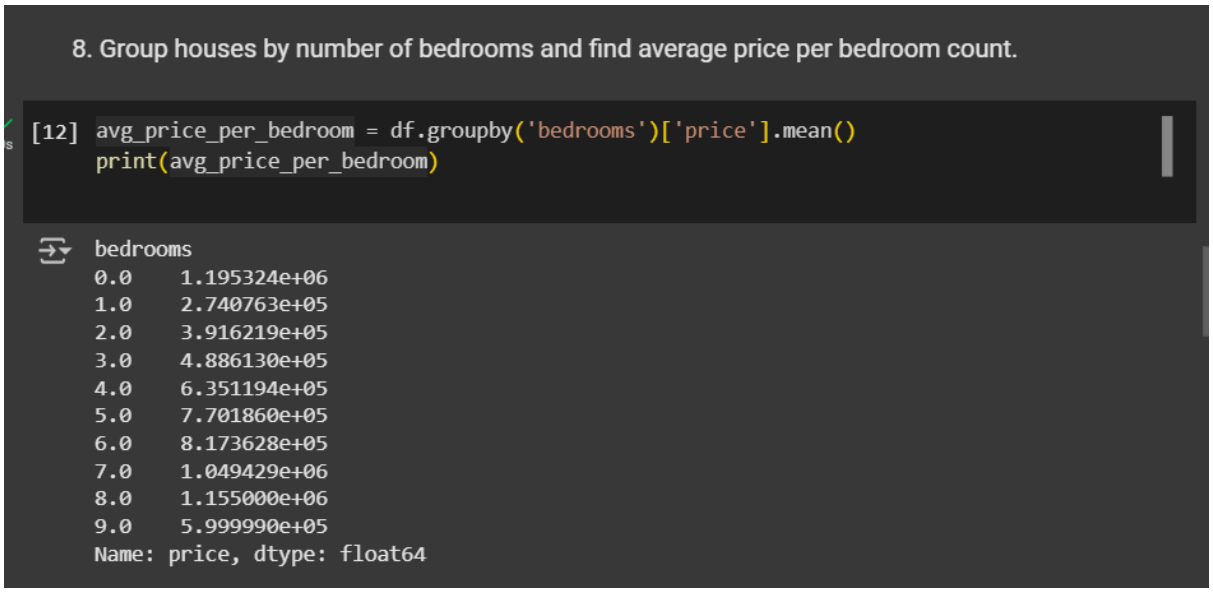


6. Sort the houses by price descending.

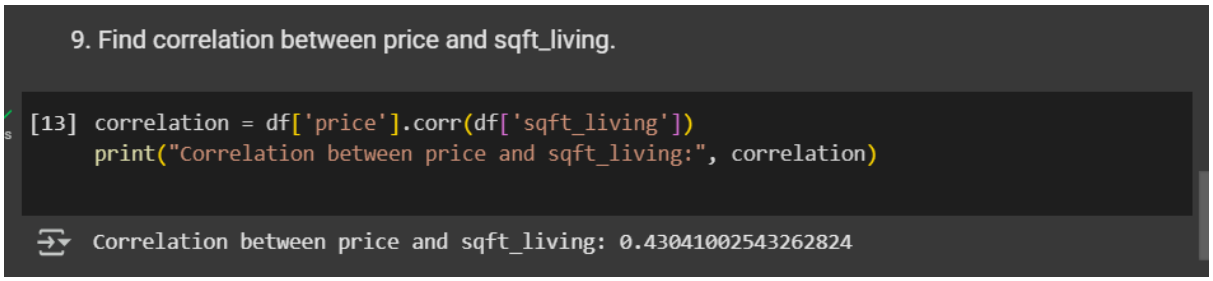


7. Count how many unique cities are present

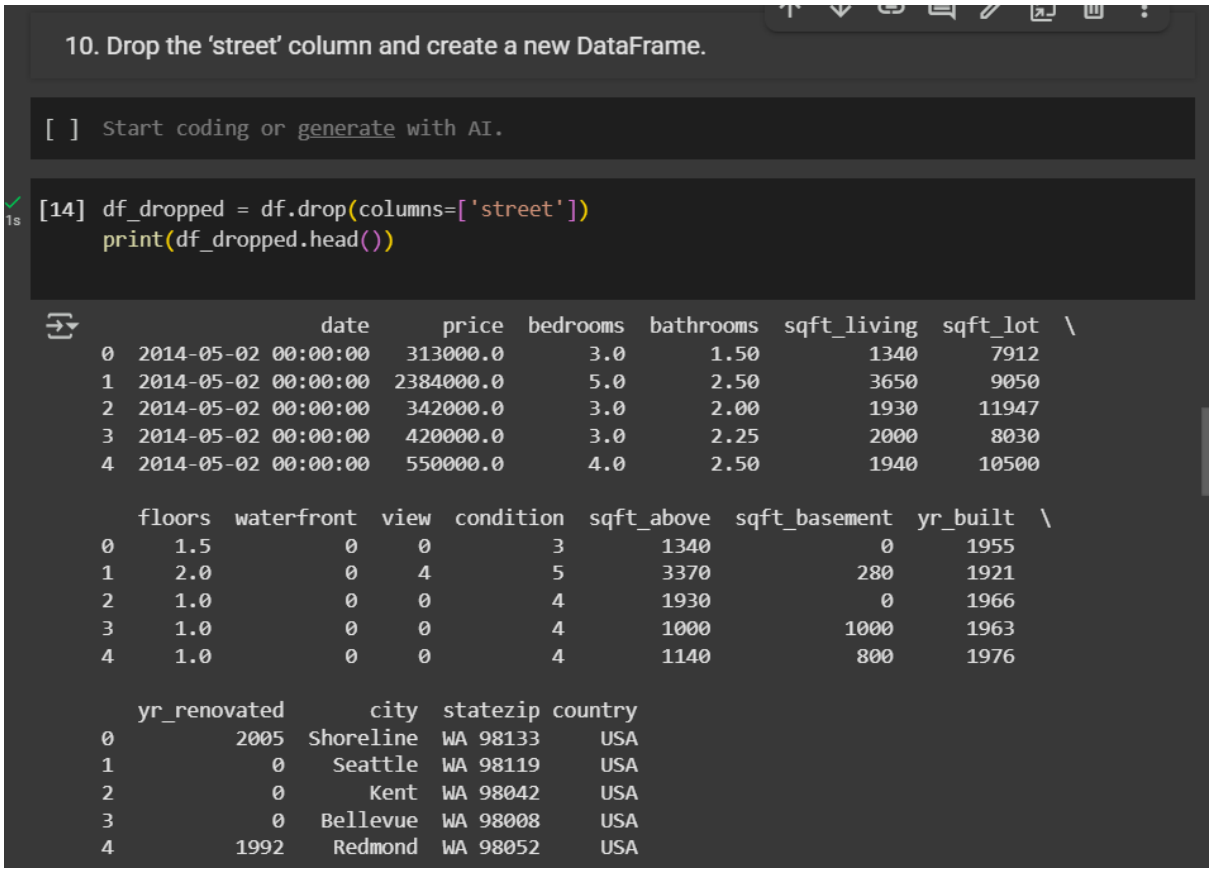
8. Group houses by number of bedrooms and find average price per bedroom count



9. Find correlation between price and sqft_living.



10. Drop the ‘street’ column and create a new DataFrame.



10. Drop the ‘street’ column and create a new DataFrame.

```
[ ] Start coding or generate with AI.
```

```
[14] df_dropped = df.drop(columns=['street'])
     print(df_dropped.head())
```

```
                  date      price  bedrooms  bathrooms  sqft_living  sqft_lot  \
0  2014-05-02 00:00:00   313000.0       3.0       1.50         1340      7912
1  2014-05-02 00:00:00  2384000.0       5.0       2.50         3650      9050
2  2014-05-02 00:00:00   342000.0       3.0       2.00         1930     11947
3  2014-05-02 00:00:00   420000.0       3.0       2.25         2000      8030
4  2014-05-02 00:00:00   550000.0       4.0       2.50         1940     10500

   floors  waterfront  view  condition  sqft_above  sqft_basement  yr_built  \
0     1.5           0     0          3        1340              0      1955
1     2.0           0     4          5        3370            280      1921
2     1.0           0     0          4        1930              0      1966
3     1.0           0     0          4        1000           1000      1963
4     1.0           0     0          4        1140            800      1976

   yr_renovated       city  statezip country
0          2005  Shoreline  WA 98133     USA
1             0    Seattle  WA 98119     USA
2             0       Kent  WA 98042     USA
3             0   Bellevue  WA 98008     USA
4          1992    Redmond  WA 98052     USA
```

Numpy

1. Calculate the standard deviation of prices.

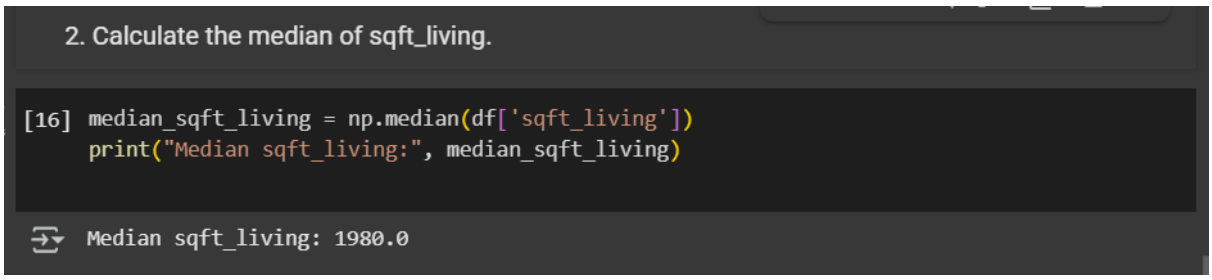
Numpy

1. Calculate the standard deviation of prices.

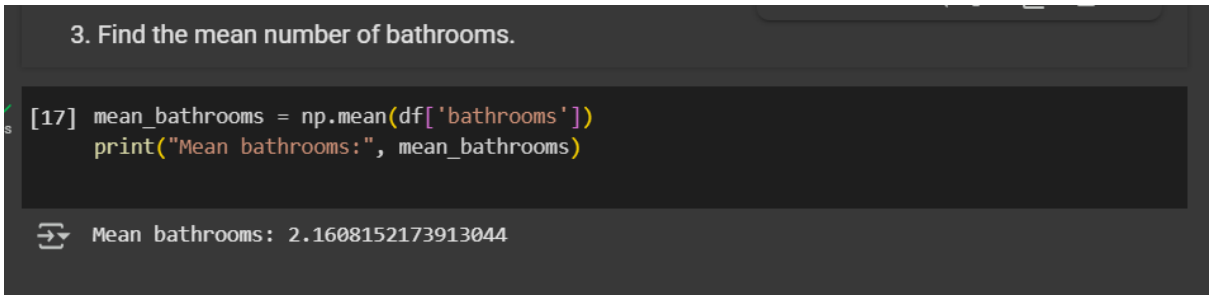
```
[15] price_std = np.std(df['price'])
     print("Standard deviation of price:", price_std)
```

```
Standard deviation of price: 563773.4128352863
```

2. Calculate the median of sqft_living.

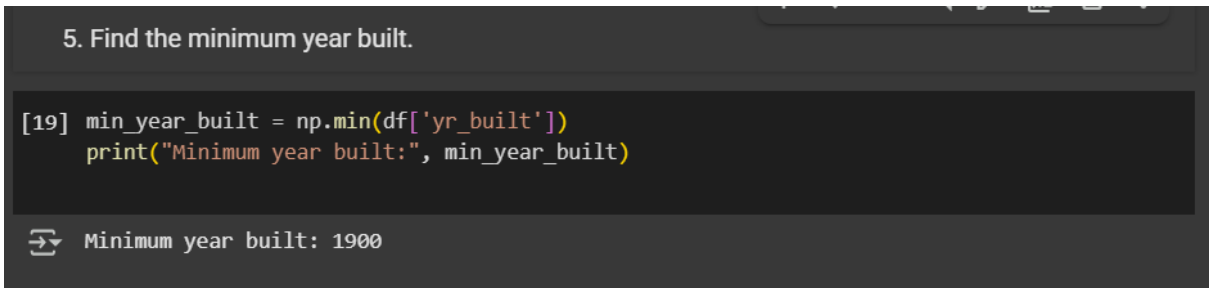


3. Find the mean number of bathrooms.



4. Find the maximum lot size (sqft_lot).

5. Find the minimum year built.



6. Create an array of prices greater than 1 million.

```
6. Create an array of prices greater than 1 million.

[20] price_array = np.array(df['price'])
     million_plus = price_array[price_array > 1000000]
     print("Houses priced above 1 million:", len(million_plus))

     Houses priced above 1 million: 340
```

7. Add 100 sqft extra to all sqft_living.

```
7: Add 100 sqft extra to all sqft_living.

[21] new_sqft_living = np.add(df['sqft_living'], 100)
     print(new_sqft_living.head())

     0    1440
     1    3750
     2    2030
     3    2100
     4    2040
     Name: sqft_living, dtype: int64
```

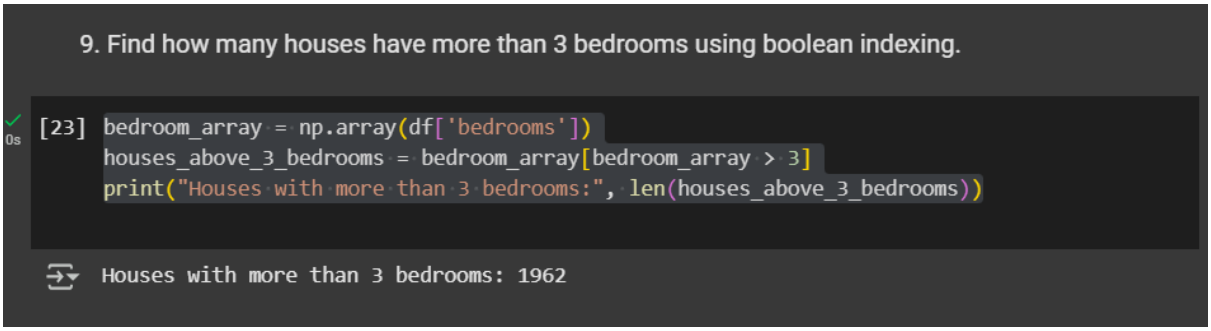
8. Multiply all prices by 1.1 to simulate price increase.

```
8: Multiply all prices by 1.1 to simulate price increase.

[22] increased_prices = np.multiply(df['price'], 1.1)
     print(increased_prices.head())

     0     344300.0
     1    2622400.0
     2     376200.0
     3     462000.0
     4     605000.0
     Name: price, dtype: float64
```

9. Find how many houses have more than 3 bedrooms using boolean indexing.



10. Calculate the variance of sqft_basement.