# Lists and Real Estate Analyzer Using Files

The owner of BDJ Real Estate of Springfield has hired you to design a sales analyzer program for real estate sales. You have been given a file called **RealEstateData.csv** that has this data:

- Street Location data type is string
- City data type is string
- Zip data type is string
- Bedroom count is an integer
- Bathroom count is an integer
- Square Footage is an integer
- Property Type is a string
- Price is a float
- Latitude is a string
- Longitude is a string

Once the data input is complete you need to write code to find out the different analytics for the sales: Min, Max, Average, Median, Total.

You will also have to get a property total by City and by Property Type. To accomplish this you will need some lists and dictionaries. How many you need of each is up to you ☺

1. In a function called **getDataInput** you will need to read the entire file in. Each record should be read in as a whole. That means do not separate the data fields/columns. This function should return all the records in a list. Remember the first record is the fields/columns heading so this record should not be processed.

2. Code a function called **getMedian** that receives a list and returns a float that does the following (you must write your own code you can **<u>NOT</u>** use the median function in the Python statistics module):
   - If the number of entries in the list is odd, divide the count by 2 and use that entry as the median
   - If the number of entries in the list is even divide the count by 2. Take that that entry and the entry before it and average the two elements and use that as the median.
   - Return the calculated median value.

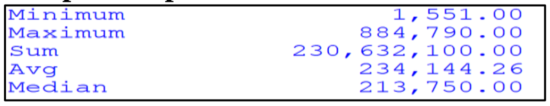
3. Code a **main** function that includes these requirements:
   - Call the **getDataInput** function to get the data you need for this program.
   - Code a loop that reads each records from the list you got in the previous step.
     - Using Python's string methods/functions extract these columns from the record:
       1. City
       2. Property Type
       3. Price
     - Each **Price** that is extracted must be added to a list you will be using to get summary functions such as Min, Max, Avg and Median.
     - For each **City** total up the **Price**. Each city should only have one summary.
     - For each **Zip** total up the **Price**. Each Zip should only have one summary.
     - For each **Property Type** total up the **Price**. Each Type should only have one summary.

   - Sort the list of Property Prices list from smallest value to largest.
   - From the Property Prices list determine the **Minimum** value and output formatted as currency with 2 decimal positions to the screen.
   - From the Property Prices list determine the **Maximum** value and output formatted as currency with 2 decimal positions to the screen.
   - From the Property Prices list determine the **Total** value and output formatted as currency with 2 decimal positions to the screen.

- From the Property Prices list calculate the **Average** and output currency with 2 decimal positions to the screen.
- From the Property Prices list the **Median** by calling the **getMedian** function you coded in Step 2 by passing list of values that was populated in the loop. Receive the Median value returned from the getMedian function and output formatted value as currency with 2 decimal positions.

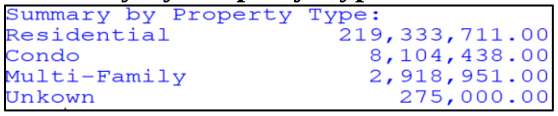
4. Output the Summaries by City and Property Types.

## Sample Output



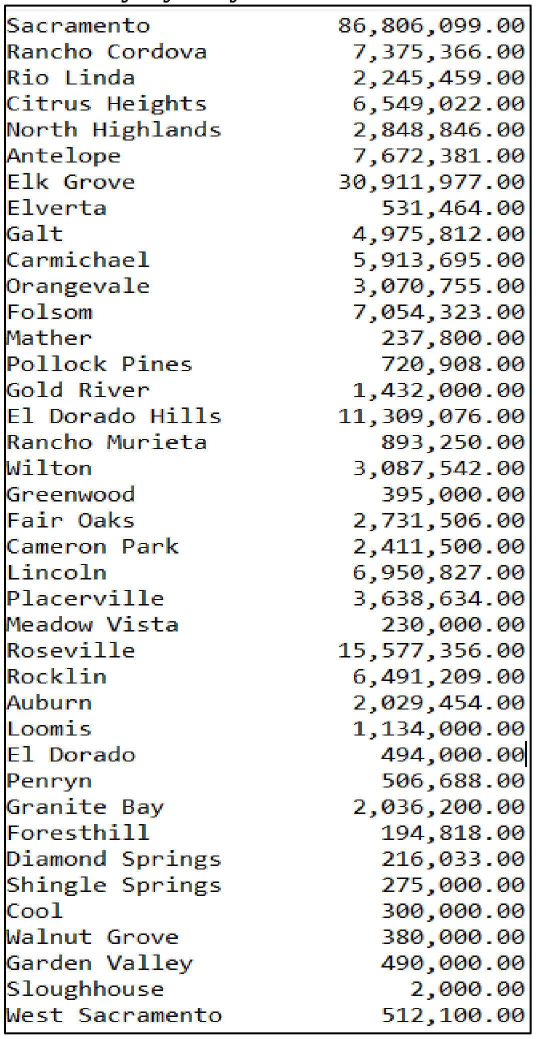
```
Minimum                    1,551.00
Maximum                  884,790.00
Sum                  230,632,100.00
Avg                      234,144.26
Median                   213,750.00
```

## Summary by Property Type



```
Summary by Property Type:
Residential              219,333,711.00
Condo                      8,104,438.00
Multi-Family               2,918,951.00
Unkown                       275,000.00
```

## Summary by City



```
Sacramento              86,806,099.00
Rancho Cordova           7,375,366.00
Rio Linda                2,245,459.00
Citrus Heights           6,549,022.00
North Highlands          2,848,846.00
Antelope                 7,672,381.00
Elk Grove               30,911,977.00
Elverta                    531,464.00
Galt                     4,975,812.00
Carmichael               5,913,695.00
Orangevale               3,070,755.00
Folsom                   7,054,323.00
Mather                     237,800.00
Pollock Pines              720,908.00
Gold River               1,432,000.00
El Dorado Hills         11,309,076.00
Rancho Murieta             893,250.00
Wilton                   3,087,542.00
Greenwood                  395,000.00
Fair Oaks                2,731,506.00
Cameron Park             2,411,500.00
Lincoln                  6,950,827.00
Placerville              3,638,634.00
Meadow Vista               230,000.00
Roseville               15,577,356.00
Rocklin                  6,491,209.00
Auburn                   2,029,454.00
Loomis                   1,134,000.00
El Dorado                  494,000.00
Penryn                     506,688.00
Granite Bay              2,036,200.00
Foresthill                 194,818.00
Diamond Springs            216,033.00
Shingle Springs            275,000.00
Cool                       300,000.00
Walnut Grove               380,000.00
Garden Valley              490,000.00
Sloughhouse                  2,000.00
West Sacramento            512,100.00
```

## Grading Rubric

| Criteria | Meets<br><br>(100%) | Somewhat<br><br>(50%) | Not Present<br><br>(0%) |
|---|---|---|---|
| **Data is imported from the file** 20 points | Data Import Function fully coded.<br><br>Data validation was fully implemented and functional. | Data Import Function somewhat coded.<br><br>Data validation was attempted but not fully functional. | Data Import Function not present.<br><br>Data validation not implemented. |
| **Coded the getMedian function as detailed in the instructions** 10 points | The getMedian Function is present and coded correctly. | The getMedian Function was attempted but with some errors or incorrect results or not coded in the most efficient manner in terms of execution. | The getMedian function is not present and/or the median logic was all placed in the main() function. |
| **Loop, List Coding and Appending Function** 35 points | Loop, List and Appending were done properly with the correct results using Python list functions. | Loop, List and Appending were attempted but with some errors or incorrect results. Or Python list functions not properly used. | Loop, List and Appending were not attempted. |
| **Summary Functions** 35 points | Summary Functions are all present and coded correctly. | Summary Functions were attempted but with some errors or incorrect results or not coded in the most efficient manner in terms of execution. | Summary Functions not attempted. |