APS106 Design Problem
Week 9: Real Estate Search

Problem Background

If you want to buy a house, you need to find houses that are for sale. The multiple listing service (MLS) is a real estate listings service in Toronto that is updated throughout the day with new listings. The service has limited capabilities with housing listed by district, rather than particular streets. It would be much more convenient for the user to be able to input a list of streets and see all the houses for sale on those streets. You goal is to take the current listings and create such an app.

1. Define the Problem

We assume that you have basic MLS real estate data in a CSV file that looks like the following (see the file `real_estate.csv` on Quercus):

Number	Street	Type	Size	Floors	Bedrooms	Bathrooms	Lot	Parking	Facing	Age	Taxes	Price
436	perth	semi-detatched	900	1	3	2	1200	0	east	100	2100	479,900
115	perth	semi-detatched	1100	2	3	2	1800	1	west	100	3500	699,900
516	perth	detatched	800	1	1	1	2600	2	east	100	2000	498,900
194	manning	semi-detatched	900	1	2	2	1400	0	west	50	4000	649,880
288	perth	detatched	1300	3	5	3	2000	1	west	100	3200	699,000
753	shaw	semi-detatched	1200	2	3	4	2200	1	east	100	4500	899,000
152	palmerston	semi-detatched	1100	2	4	3	1800	1	west	100	4300	799,000
179	perth	detatched	1100	2	3	2	2200	1	west	50	2800	699,000
295	clinton	semi-detatched	1300	3	5	3	2000	1	east	100	4600	899,000
189	clinton	semi-detatched	1250	2	3	2	2100	0	west	80	4200	849,900
277	clinton	detatched	1100	1	2	2	2200	2	east	50	4800	949,800
264	clinton	detatched	1300	2	3	2	1800	0	east	100	4400	799,900
265	clinton	detatched	1100	1	2	1	1800	0	west	100	4300	790,000
517	delaware	semi-detatched	1200	2	3	1	2100	2	east	80	3200	699,900
713	euclid	semi-detatched	1100	3	3	2	1500	0	west	70	4100	749,000
808	shaw	detatched	1300	3	3	2	1700	1	east	100	3500	929,000
488	delaware	detatched	1100	1	2	2	1900	1	west	80	3400	849,000
308	delaware	semi-detatched	1100	2	3	3	1200	1	east	100	2600	599,000
332	delaware	semi-detatched	1000	3	5	2	1400	1	west	80	3300	799,000
17	mansfield	semi-detatched	900	2	3	2	1300	0	north	50	3500	699,000
13	mansfield	semi-detatched	1300	2	3	2	1300	0	north	50	3800	899,000

You want to create an app that does the following:

```
Enter a street name (type exit when done): manning
Enter a street name (type exit when done): wells
Enter a street name (type exit when done): perth
Enter a street name (type exit when done): exit

Houses on manning
Address: 194 manning Size: 900 Price: $ 649880

Houses on wells
No houses on wells

Houses on perth
Address: 436 perth Size: 900 Price: $ 479900
Address: 115 perth Size: 1100 Price: $ 699900
Address: 516 perth Size: 800 Price: $ 498900
Address: 288 perth Size: 1300 Price: $ 699000
Address: 179 perth Size: 1100 Price: $ 699000
```

In other words, you want the user to input a number of street names and you should print out the information on available houses on each street.

2. Define Test Cases

The above example is a test case. You can also think of some others.

3. Generate Many Creative Solutions

Start with a very high-level Algorithm Plan. Perhaps, something like this:

- Read in the data and store it somehow
- Get the user input and store it somehow
- For each street entered, find houses on the street
- Print-out the address, size, and price of each house found

It is not clear how you are going to do all these steps. Come up with a programming plan about how you are going to both figure out and implement these high-level steps. As you think about this, you probably will need to refine your Algorithm and Programming Plans.

4. Select a Solution

At this point your should have at least one well-thought out approach, perhaps even with bits of code that do part (even a significant part) of what you want to do.

You should think about whether all this work has resulted in a satisfactory solution or if you want to re-think aspects of the above.

5. Implement the Solution

Putting it all together. Add appropriate comments.

6. Perform Final Testing

Run all of our tests to make sure it is working.