Project 1 – Address Book   
with Multiple Field Search



Create an address book that can provide search functionality on multiple fields.  The structure is very simple – first name, last name, phone number and city name.  **You can search on ANY of these fields, and can search partially using “begins with” syntax.** For example, you can search for – first name starts with “Ron…”.  Another search can be – phone number starts with “202…”.

Obviously, binary search will play a role here. So, you can try to keep the list sorted. But one interesting aspect to consider is that we can search on ANY of those attributes. That is, the user will give it a few searches (some on first name, some on last name, some on telephone number, some on city).

# The Main Point

# Sample Program Run

AddressBook mylist.csv

* Welcome to Addressbook, initialized.
* What would you like to search on? (F,L,P,C)
* F
* Enter the partial First Name
* Ron
* Results: Ronald Reagan 202-994-3000 Washington DC, Ron de Silva… (43 total matches), 71 ms
* Another Search? (Y,N)
* Y
* What would you like to search on? (F,L,P,C)
* L
* Enter the partial Last Name
* Hug
* Results: Paul Hughes 703-384-1222 New York, Victor Hugo … (11 total matches), 86 ms
* Another Search? (Y,N)
* Y
* What would you like to search on? (F,L,P,C)
* C
* Enter the partial city name
* Bo
* Results: Paul McDermitt 302-384-9888 Boston, … (102 total matches), 91 ms
* Another Search? (Y,N)
* N
* Bye

The time to search in an unsorted list is O(n). The time to search in a sorted list is O(log n). But the time to sort a list is O(n log n). **If you sort it on demand, and then search, that will take O(n log n + log n), that is O(n log n) time, which is worse than O(n).** If the user were to change the “criteria” on every query, you would need to sort it every time. **How can we make the searching work, without having to sort it on every request?**

# Simplifying Assumptions

1. For convenience, assume that none of the fields have spaces or commas in them, so you can read a comma separated file without worrying about those (non-algorithm related) complications.
2. No need to consider a “contains” or “ends with” search.
3. Consider data and search to be all case sensitive, so don’t worry about transforming data.
4. Don’t worry about duplicate results.

# Submission Method

Soft copy: Write up, test results data, and source code. Be ready to demonstrate the program.

# What you can share with each other

* Data files etc.
* Ideas
* Debugging help