Programming Assignment #3
CS 163 Data Structures

Submit your assignment to the D2L Dropbox (sign on via d2l.pdx.edu)

Background: As we begin the next phase of CS163, we will be working with "Table Abstractions". A Table ADT allows us to search on the "value" of the data without requiring a particular location to be known. It allows us to store the data using non-linear techniques. A hash table comes to mind as a good alternative for programs that require working by "value" but that don't require that the data be sorted.

Lecture on February 7, 2019 and Lab #6 covers the topic of hash tables.

Goal: The goal of the third program is to create a **hash table using chaining**. Hash tables are very useful in situations where an individual wants to quickly find their data by the "value" or "search key". The client program uses a "key" instead of an "index" to get to the data. The key is then mapped through a **hash function** which supplies an index!

Specifics: Have you ever wanted to find information very quickly and you were not interested in sorting the data? I don't actually watch much TV. I usually do something else when my family has TV on. But, during the most recent Portland weather I really wanted to get the most up to the minute forecast. And, I found some of the material on the web was not always updated at the rate desired. So, I spent a lot of time streaming various videos, reading blogs from local meteorologists, watching the trip-check cameras (lots of fun!) and flipping through the main news channels. I think I spent way too much time doing this.

It would have been much better if I could have put in my search criteria (Portland snow storm) and have found the most up to the minute information that just didn't direct me to some useless advertisement. I wanted to quickly find information without having to perform a sequential search.

For this programming assignment, we will create an external data file that consists of information we can retrieve from our cable channels, Netflix, Youtube, as well as online websites. It doesn't have to be about weather – it can be any relevant search criteria. Remember to plan on putting a delimiter in your external data file (e.g., '|') between each field and then have a newline after the last field.

At a minimum, the information that we want to keep track of includes:

- Name (Youtube)

- List of at search keys (this is the information that you will allow the user to search on)
- Number of search keys
- Channel or website address (www.youtube.com/user/TheWeatherChannel)
- Description
- Information you learned from the site
- Rate the usefulness

Data Structures: Write a C++ program that implements and uses a **table abstract data type using a hash table (with chaining). The hash table should be organized by search key. Remember to create a hash function where the value created is larger than the array size so that we can mod by a prime number.**

Implement the following functions:

1. Add a new channel or website (that is not in the external data file)
2. Search for a key (supply matching information back to the client program that matches the search key supplied)
3. Remove by name
4. Display all matches for a search key
5. Display all (not in order!).

Evaluate the performance of working with this table. Monitor the number of collisions that occur for a given set of data that you select. Make sure your hash table's size is a prime number. **Try AT LEAST TWO different table sizes**, and evaluate the performance (i.e., the length of the chains!). **Your efficiency writeup must discuss what you have discovered.**

Realize that the remove by name will need to perform a sequential search since the hash table should be organized by search key!

The data must originally be retrieved from an external data file. This is required for this assignment. In order to properly evaluate the hash function chosen, we need a large data set! **However:** your program will not be removing or modifying the data from the file.

Things you should know...as part of your program:

1) Do not use statically allocated arrays in your classes or structures. All memory must be dynamically allocated and kept to a minimum!
2) All data members in a class must be private
3) Never perform input operations from your class in CS163
4) Global variables are not allowed in CS163
5) **Do not use the String class! (use arrays of characters instead and the cstring library!)**
6) Use modular design, separating the .h files from the .cpp files. Remember, .h files should contain the class header and any necessary prototypes. The .cpp files should contain function definitions. You must have at least 1 .h file and 2 .cpp files. **Never "#include" .cpp files!**
7) Use the iostream library for all I/O; do not use stdio.h.
8) Make sure to define a constructor and destructor for your class. Your destructor must deallocate all dynamically allocated memory.