Assignment: 1 Questions & Response

1. **Collect running time statistics for each operation in both implementations.**

Please Refer to attached Excel Sheet for Execution Times.

(NOTE): Some of my data seems inaccurate, this could be from my lack of experience using the <Chronos> and <Ctime> libraries, but it could also be attributed to the fact that my Linked List Implementation is also more efficiently written than my array implementation. I have more experience making linked lists applications modular and I use reference pointers often with them, as apposed to how I implemented my array application, which was more how I would naturally program it as apposed to what would likely be the more efficient method, IE using reference pointers to quickly transfer data from within the array instead of searching for it.

1. **What are your conclusions about the relative advantages and disadvantages of the two implementations?**

I believe from my trial experience, removing and adding data from linked lists are more efficient in terms of memory usage and in some cases time complexity, but arrays are easier to traverse and faster to traverse. Linked lists appear to be better if you need to add and remove data often and if you only need reference data sparingly, while arrays are more favorable if the user doesn’t have to edit that data often and if its being traversed for reference data often.

1. **Would storing records on the list in alphabetical order by city name speed any of the operations?**

I do not believe it would be faster in the case of the linked list, since regardless the program has to traverse the list in linear order, so sorting it would do practically nothing other than take up time.

In the case of the array, It could increase performance because it would allow the user to implement a sort of binary search, giving priority to an alphabet of constant char variables, and traversing the array based on if the element being compared is ahead or behind the midpoint of its current object and the end of the array or beginning of the array.

1. **Would keeping the list in alphabetical order slow any of the operations?**

Yes, in both cases because, in both the linked list case and array case, if we add an element, we would have to resort the array or linked list every time. The array case would also see performance drops in the delete operations, since it would need to manage where to put the empty objects every time.  
  
The linked list application would only see performance drops in the add element operation.