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

|  |  |  |
| --- | --- | --- |
|  | Array | Linked List |
| Implement | 0.000982 | 0.001039 |
| Insert | 0.016686 | 0.002636 |
| Search by name | 0.005997 | 0.003291 |
| Search by coordinate | 0.001413 | 0.000883 |
| Delete by name | 0.021009 | 0.00591 |
| Delete by coordinate | 0.004456 | 0.01549 |
| Print records within a distance | 0.039385 | 0.035488 |
| Print all | 0.105423 | 0.032561 |

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

The advantage of using the array is that it is easier to use and less error prone than linked lists. The advantage of using a linked list is that it is faster and more flexible than arrays in that it can be implemented with as much data as you want and can fully delete locations. The disadvantage of arrays is that it is much slower than linked lists. The disadvantage of linked lists is the significant amount of code needed to implement them, they are more prone to errors, memory leaks, along with the general difficulty to use them. To traverse the linked list, you must go forward, and the data is not randomly accessible.

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

For the linked list, this would not speed up the operation as it must be traversed in order. However, for the array this could potentially speed it up as there are many more searches available. For example, the binary search is available which is almost always as fast as the linear search.

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

Yes, keeping the list in alphabetical order would add the difficulty of always having to insert the city in the appropriate location. For an array this would be very expensive as the rest of the array would have to be adjusted.