Angelo E.
CS 220
Due 4/8/25

# Milestone 5 Paper

**Intro:**
For this milestone I implemented all the previous milestones. Then evaluated the three data structures: Linked list, Binary search tree, and the Hash table. The purpose was to find which structure was the most efficient to handle operations such as insertion, search and deletion. These are core operations for data management. Optimizing those operations are important when you have to work with very big data sets.

**Setup:**

DATASETS

For me I had to set it up by copying the classes and placing them all into one sort of master class. This was due to being unorganized while doing the other classes and I also ran into the problem of me making different inserts for all the classes and giving them different names. This led to me rewriting the inserts to get them to match and work with the data tester class. I used two data sets one with all the Peoria schools from Illinois then the other being all the schools in Illinois. Each dataset had basic information such as the school name, address, city, state and county.

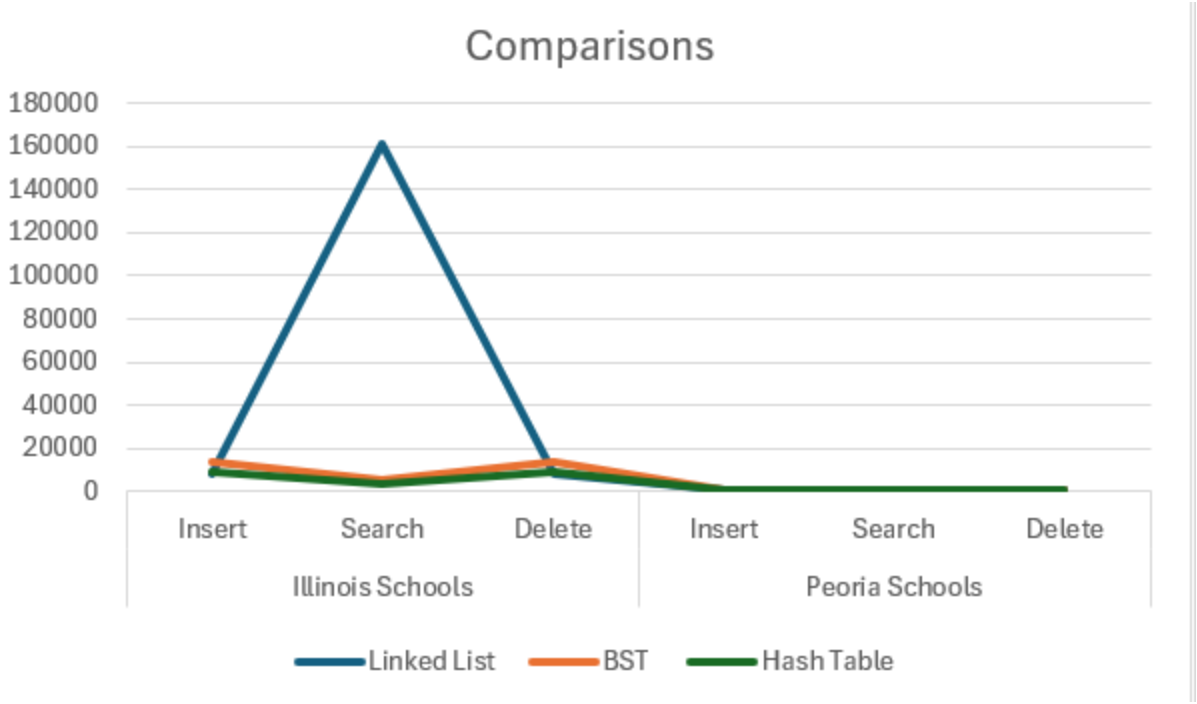
TOOLS AND METHODOLOGY

Everyone wrote the test code in C++ and instead of using the Timer.h I used the chrono library for the timing. At first I tried to time it in milliseconds but the operations were executed so fast that it kept showing 0ms. To fix this issue I switched it to microseconds to make it more accurate. The class used for testing I named it “SchoolDataTester” this handled basically everything such as reading names from the CSV, using each operation on all the data structures, and also recording the time taken for insert, search, and delete. After getting the data I used Excel to make a table comparing the two datasets then created a line graph based off the table to help visualize the differences.

Results

Here is a screenshot showing the excel sheet and line graph:

| Micro Sec. | Illinois Schools | | | Peoria Schools | | |
|---|---|---|---|---|---|---|
| | Insert | Search | Delete | Insert | Search | Delete |
| Linked List | 7808 | 161222 | 7808 | 309 | 74 | 309 |
| BST | 13459 | 4849 | 13459 | 323 | 45 | 323 |
| Hash Table | 9044 | 3744 | 9044 | 195 | 160 | 195 |




The first thing to take note of is that the Hash table was the fastest in all ways. Then in second was BST. And in last was the linked list since it went through the list linearly.

**Conclusion**

In the end the Hash table performed the best. It was the fastest for each operation this means that it is ideal for real time systems where performance is important. The BST wasn't very far behind. And the worst but easier to implement in my opinion was the Linked list. This project showed me how important it is to use the correct data structure when dealing with larger datasets.