Evaluation

Runtime Analysis:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | **Vector** | **Hash Table** | **Binary Search Tree** |
| Open File | O(1) | O(1) | O(1) |
| Read File (n lines) | O(n) | O(n) | O(n) |
| Parse Line | O(n) | O(n) | O(n) |
| Insert Course | O(n) | O(n2) (worst) | O(n2) (worst) |
| **Total Runtime** | O(n) | O(n2) | O(n2) |

While the vector has a lower worst-case runtime for inserting and loading the courses (O(n)), the downside is that searching for a specific course or printing an alphanumeric list will require O(n) operations each time. This linear search can become costly, especially with large datasets, since there’s no inherent structure to improve search times. However, vectors are straightforward to implement, requiring less memory overhead, making them a good option for small or moderately sized datasets.

The binary search tree (BST) offers more efficient searching and insertion operations, which makes it a good candidate for larger datasets. The primary benefit of the BST is that it maintains the data in sorted order, which aligns perfectly with the program’s need to print an alphanumeric list. However, in the worst case the performance degrades to O(n), and the complexity of implementing a balanced tree increases.

The hash table offers the fastest lookups (O(1) on average), making it an excellent choice for fast searching. However, hash tables don’t maintain any inherent order, so printing a sorted list becomes challenging without extra steps. Additionally, collisions can degrade performance to O(n), and hash tables tend to use more memory.

I believe using the vector would be best for this program. Although the BST provides better search performance in theory, its worst-case performance of O(n) can’t be ignored, and implementing a balanced tree is complex. For small to moderate datasets, the vector’s simplicity and predictable runtime make it the most practical choice. While searching and printing will take O(n) time, this is acceptable given the straightforward nature of the dataset and use case. If the program were to handle significantly larger datasets, the BST might be reconsidered, but for now, the vector offers the best balance between ease of use and performance.