[CS-300-H2996 DSA: Analysis and Design](https://learn.snhu.edu/d2l/home/1426308)

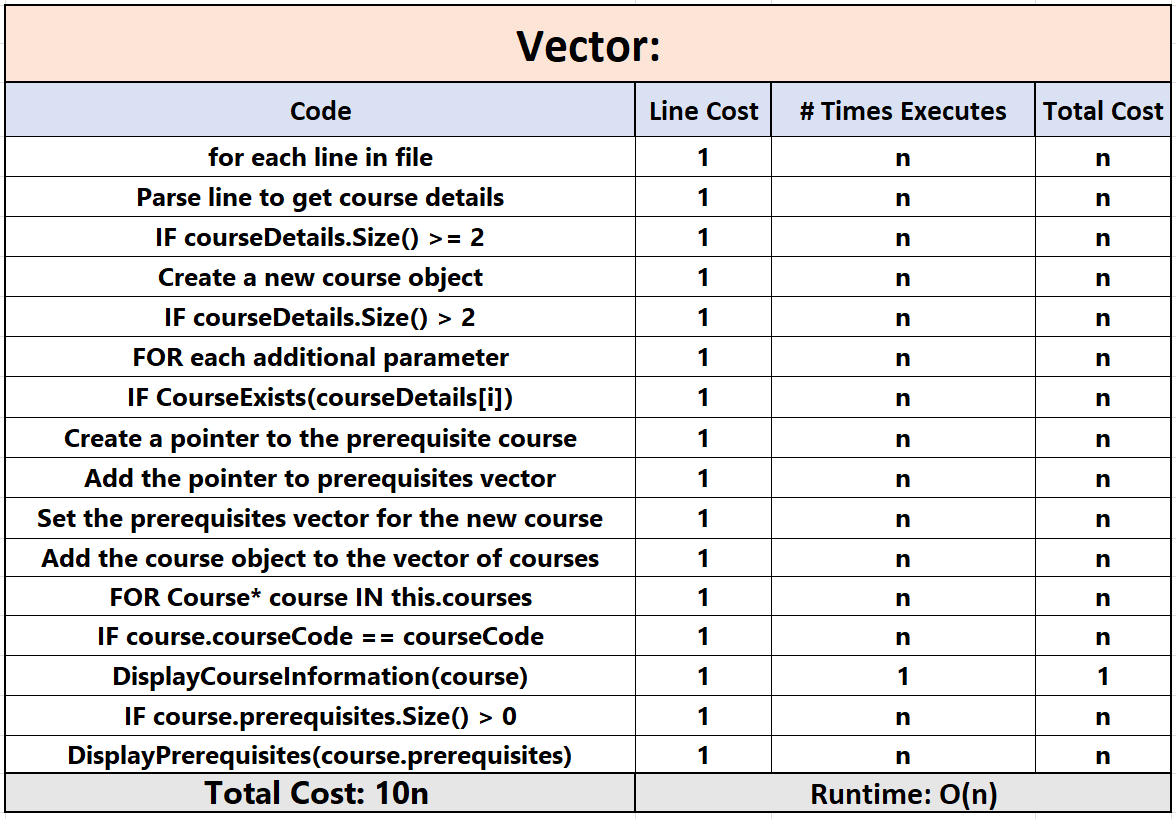
Adam Vosburg

Southern New Hampshire University

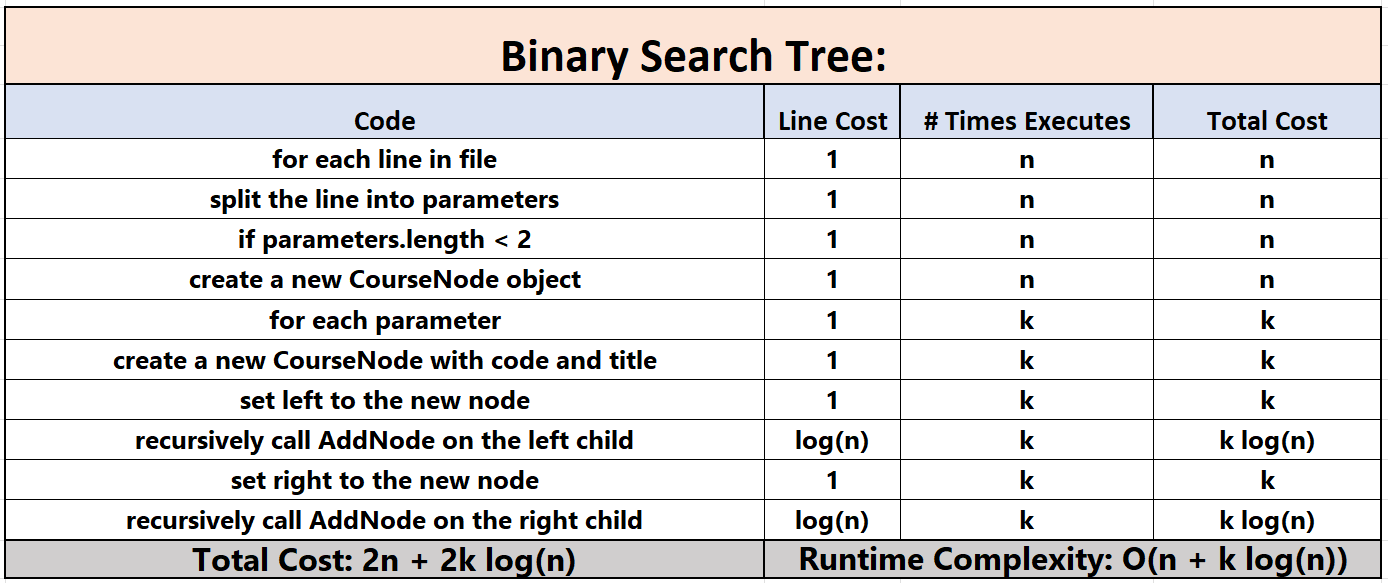
6-2 Project ABC University Analysis

11/28/2023

**6-2 Project ABC University Analysis**

****

****



**Run Time Analysis Conclusion:**

In the landscape of course management, three prominent data structures—Vector, Hash Table, and Binary Search Tree (BST)—each present distinct advantages and drawbacks. The Vector, lauded for its direct access to course data through indices, ensures simplicity and ease of use. Its straightforward implementation, marked by simplicity, facilitates easy understanding and management. However, as the dataset expands, the Vector encounters inefficiency in specific operations, especially with large datasets, where the linear search for courses becomes progressively less efficient. Additionally, operations like insertion or deletion can become computationally expensive, diminishing the Vector's appeal for dynamic course management needs.

The Hash Table emerges as a powerful contender, showcasing efficiency in swift searches, insertions, and deletions, making it particularly well-suited for dynamic course management scenarios. Its adaptability shines as it handles varying workloads effectively, providing resilience to changes in the course catalog. However, the Hash Table is not without its challenges. Collisions may occur, necessitating careful handling to maintain efficiency. Moreover, the hashing process introduces a slightly higher memory overhead, a trade-off for its computational advantages.

The Binary Search Tree (BST) introduces order to operations, offering an elegant solution for scenarios requiring an ordered structure, such as printing courses in a specific order. Its dynamic nature allows it to adapt well to changes in the course catalog, resembling the Hash Table in this aspect. Nevertheless, unbalanced trees can impact search and insertion performance, posing a significant drawback. Additionally, the management of pointers adds a layer of complexity to the implementation of Binary Search Trees.

In conclusion, each data structure has its strengths and weaknesses in the context of course management. The Hash Table emerges as the pragmatic choice due to its efficiency in handling frequent searches, insertions, and deletions. Its adaptability to varying workloads and dynamic resizing capabilities position the Hash Table as a fitting solution for practical course management needs. While the Vector and Binary Search Tree offer unique advantages, the Hash Table strikes a balance that aligns well with the demands of efficient and dynamic course data management.

Its resizing capability makes it a fitting solution for dynamic course management. While a balanced BST may carry allure, the Hash Table is the efficient solution aligned with practical needs.