CS 300: Data Structures and Algorithms

Professor: David Ostrowski

Student: Phong Nguyen

phong.nguyen1@snhu.edu

Southern New Hampshire University

**Run time Analysis**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Data Structure** | **Loading Data (Big O)** | **Parsing & Creating Objects (Big O)** | **Find Course (Big O)** | **Insert Course (Big O)** | **Space Complexity** | **Advantages** | **Disadvantages** |
| **Vector** | O(n) | O(n) | O(n) | O(1) | O(n) | Simple, sequential access, dynamic resizing | Slow search, costly middle insertions/deletions |
| **Hash Table** | O(n) | O(n) | O(1) | O(1) | O(n + k) (k = buckets) | Fast look-up, efficient for large datasets | High memory usage, complex implementation |
| **Binary Search Tree** | O(n) | O(nlogn) | O(logn) | O(logn) | O(n) | Efficient for insertion, deletion, and sorting | Higher memory usage, balanced tree maintenance |

**Vector**

* Advantages:
  + Efficient for sequential access.
  + Dynamic resizing to handle varying data sizes.
* Disadvantages:
  + Inefficient for searching (O(n)).
  + Costly insertions and deletions in the middle (O(n)).

**Hash Table**

* Advantages:
  + Fast lookups and insertions (O(1) on average).
  + Efficient handling of large datasets.
* Disadvantages:
  + Higher memory usage due to additional structures for handling collisions.
  + More complex implementation and maintenance.

**Binary Search Tree**

* Advantages:
  + Efficient for searching, insertion, and deletion (O(log n)).
  + Naturally maintains order, making sorting operations straightforward.
* Disadvantages:
  + Requires balancing to maintain performance (which can be complex).
  + Higher memory usage due to tree structure.

**Verdict:**

Based on the analysis, the Hash Table is recommended for managing the course database. Here’s why:

1. **Efficiency in Lookups and Insertions:** The average time complexity for these operations is O(1), making it highly efficient for accessing course data quickly.
2. **Handling Large Datasets:** Hash tables handle large datasets more effectively, ensuring that operations remain fast even as the number of courses grows.
3. **Unique Identifiers:** Courses typically have unique identifiers (course numbers), which fit well with the hash table’s key-value storage mechanism.

While vectors offer simplicity and ease of use for smaller datasets, their performance degrades with larger datasets. Binary Search Trees provide a good balance but require more complex maintenance to keep the tree balanced. Given the typical requirements of a course management system (frequent lookups and insertions), the Hash Table offers the best performance and scalability.