Evaluation:

|  |  |
| --- | --- |
| Data Structure | Time Complexity |
| Vector | O(n^2) |
| Hash Table | O(n) |
| Binary Tree | O(n log n) |

For the vector data structure, the worst-case running time for reading the file and creating course objects is O(n \* m), where ‘n’ is the number of courses stored and ‘m’ os the average number of tokens per line. In comparison, the worst-case running time of the hash table and binary tree data structures in O(n). however, because all three data structures hold the same set of course objects, their memory use remains O(n). In terms of speed, if the average amount of tokens per line is minimal i comparison to the overall number of courses, the hash table or binary tree may be more efficient alternatives.

It is clear from the analysis that each data structure has advantages and disadvantages. Although searching for a specific course would take linear time and became less effective as the number of courses increased, vectors are easy to use and retain insertion order. On the other hand, hash tables provide constant time access for speedy lookups and effectively manage large datasets, although addressing collisions and managing hash functions might make the implementations more difficult. For balanced trees, binary trees offer quick access and quick searches, while unbalanced trees need balancing and might decline to O(n) time complexity.

The hash table is the most appropriate data structure, considering the particular requirements of the advisor’s advising program. It is great at delivering quick access times (O(1) average case) for getting course information using course numbers, which is vital for the program. Hash tables can scale for future course expansions since they can handle larger datasets efficiently. Although binary trees also perform well, hash tables are the favored option for this situation due to their ease of development and efficient collision management. As a result, the hash table is the best data structure to utilize in this program since it balances speed and implementation in comparison to binary trees and vectors.