Evaluation

The run time for each of the data structures to read the file the, parse each line, and create a course object would differ depending on how it searches, organizes, and traverse the data. Utilizing the vector data structure would produce O(n) as the worst-case scenario, since it would take O(n) to read the file, split each line correctly, create the course object, and insert it into the vector. Which would equal O(n^2) since there can be up to n amount of prerequisites for the course. The worst-case scenario run time for the hash table would be O(n^2) since there is a chance for collisions. With a total run time of O(n). Finally, the binary search tree would also have O(n^2) as a worst-case scenario for its run time. Binary Search Tree has a total runtime of O(nlog).

Vector Run Time

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **for all courses** | 1 | n | n |
| **if the course is the same as courseNumber** | 1 | n | n |
| **search for course** | 1 | n | n2 |
| **for each prerequisite of the course** | 1 | n | n |
| **print the prerequisite course information** | 1 | n | n |
| **Total Cost** | | | n2+4n |
| **Runtime** | | | O(n2) |

Hash Table Run Time

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **for all courses** | 1 | n | n |
| **if the course is the same as courseNumber** | 1 | n | n |
| **Search for course** | 1 | n | n |
| **for each prerequisite of the course** | 1 | n | n |
| **print the prerequisite course information** | 1 | n | n |
| **Total Cost** | | | 5n |
| **Runtime** | | | O(n) |

Binary Search Tree Run Time

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **for all courses** | 1 | n | n |
| **if the course is the same as courseNumber** | 1 | n | n |
| **Search for course** | logn | n | nlogn |
| **for each prerequisite of the course** | 1 | n | n |
| **print the prerequisite course information** | 1 | n | n |
| **Total Cost** | | | 3n+nlogn |
| **Runtime** | | | O(nlogn) |

Every data structure has its pros and cons. Some advantages that vectors have would be that it preserves the order, lower memory usage, faster than binary search trees, and works better with smaller data. Some disadvantages of vectors would be that it takes longer to search, insert, and delete elements due to the search time being linear. Some advantages of a hash table would be that it is extremely fast to search, insert and delete data, scales well with a large set of data, and handles collisions with chaining. Some disadvantages of the hash table would be that it does not maintain order unless there is an ordered hash. It usually uses more memory for more buckets. Finally, if the hash function is not good, it can potentially lead to more collisions. The binary search tree advantages is that it has a sorted order, it is extremely fast with searching, inserting, and deleting data, and can handle a large amount of data. Some disadvantages of the binary search tree are that it can take more memory than a vector but less than a hash table. If the tree is unbalanced, it can take longer to traverse the tree. Finally, there is no direct access to a node in the tree.

The recommended data structure to store all the different courses would be a binary search tree for the academic advisors in the Computer Science department at ABCU. This data structure’s advantages outweighed the other data structure’s advantages. Although it is not the fastest, it can handle a very large amount of data, and it is easier to modify the data compared to other data structures. As the school grows and evolves, different classes will always be added and deleted which makes it imperative for the data structure to be able to handle a lot of data. Although the hash table is the fastest with a runtime of O(1) on average, BST is more efficient since it has a runtime of O(n) while being sorted and in order. Since the data must be ordered, the hash table would become inefficient when dealing with sorted data, while a binary search tree is much more capable of handling large amounts of ordered data.