**Runtime Analysis**

| **Vector Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| Declare new Course object | 1 | 1 | 1 |
| Declare integer courseNumber | 1 | 1 | 1 |
| Declare string courseTitle | 1 | 1 | 1 |
| Declare vector <string> prerequistes | 1 | 1 | 1 |
| WHILE EOF is not reached  READ in courseNumber  READ in courseTitle  READ in prerequisites if any exist | 4 | n | 4n |
| Store courseNumber, courseTitle, and any prerequisites in course object | 1 | n | n |
| PUSH course object onto courses vector | 1 | n | n |
| **Total Cost** | | | 6n + 4 |
| **Runtime** | | | O(N) |

| **Hash Table** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| Declare new Hash Table object | 1 | 1 | 1 |
| Declare new Course object | 1 | 1 | 1 |
| Declare integer courseNumber | 1 | 1 | 1 |
| Declare string courseTitle | 1 | 1 | 1 |
| Declare vector <string> prerequistes | 1 | 1 | 1 |
| WHILE EOF is not reached  READ in courseNumber  READ in courseTitle  READ in prerequisites if any exist | 4 | n | 4n |
| Store courseNumber, courseTitle, and any prerequisites in course object | 1 | n | n |
| Insert Course object into Hash Table | 1 | n | n |
| **Total Cost** | | | 6n + 5 |
| **Runtime** | | | O(n) |

| **Binary Search Tree** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| Declare new Binary Search Tree object bst | 1 | 1 | 1 |
| Declare root equal to nullptr | 1 | 1 | 1 |
| Declare new Course object course | 1 | 1 | 1 |
| Declare integer courseNumber | 1 | 1 | 1 |
| Declare string courseTitle | 1 | 1 | 1 |
| Declare vector <string> prerequistes | 1 | 1 | 1 |
| WHILE EOF is not reached  READ in courseNumber  READ in courseTitle  READ in prerequisites if any exist | 4 | n | 4n |
| Store courseNumber, courseTitle, and any prerequisites in course object | 1 | n | n |
| Call Insert(course) | 1 | n | n |
| **Total Cost** | | | 6n+ 6 |
| **Runtime** | | | O(n) |

**Advantages and Disadvantages**

Vector

Advantages:

* The size of vectors are dynamic, therefore they can be increased.
* Vectors contain several member functions that allow for insertion and deletion.

Disadvantages:

* Searching for a particular element in a vector is O(n) due to potentially having to iterate through the entire vector.
* Insertion and deletion in the middle of a vector is very insufficient due to having to reallocate data.

Hash Table

Advantages:

* Hash tables are constant when searching, inserting, and deleting a particular element.

Disadvantages:

* Collisions can cause hash tables to become inefficient.

Binary Tree

Advantages:

* Binary trees can be implemented to maintain order after insertion so it’s easy to maintain a sorted list of the elements.
* Binary trees are O(log n) when searching, inserting, and deleting elements.

Disadvantages:

* Binary trees are more complex and require more memory.

**Recommendation**

Based on the Big O analysis and my analysis of each data structure, I would recommend using the hash table for this assignment. I recommend using a hash table, because hash tables are efficient for searching, inserting, and deleting elements. Each of these operations’ runtime is constant due to being able to hash a key value and locate a particular bucket in the hash table.