SNHU CS 300: Project One

By: Michael Duke

4/13/2025

**Runtime Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| **Vector Data Structure** | **Line Cost** | **# of Times Executed** | **Total Cost** |
| Create vector | **1** | **1** | **1** |
| For each line in file | **1** | **N** | **N** |
| Create vector course items | **1** | **N** | **N** |
| Create vector | **1** | **1** | **1** |
| While prerequisites exist: | **1** | **N** | **N** |
| Append prerequisites (while present) | **1** | **N** | **N** |
| Pushback course item | **1** | **N** | **N** |
|  |  | **Total Cost** | **5N + 1** |
|  |  | **Runtime** | **0(N)** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Hash Table** | **Line Cost** | **# of Times Executed** | **Total Cost** |
| Create hash table | 1 | 1 | 1 |
| Insert method | 0 | 0 | 0 |
| Create key for course | 1 | N | N |
| If no entry found for key | 1 | N | N |
| Assign node to key | 1 | N | N |
| Else | 1 | N | N |
| Assign old node key to UNIT\_MAX, set as key, set the old node to course | 4 | N | 4N |
| Else | 1 | N | N |
| Find the next open node | 1 | N | N |
| Add newNode to end | 1 | N | N |
| For each line in file | 1 | N | N |
| Create vector course item | 1 | N | N |
| While prerequisite exists | 1 | N | N |
| Append prerequisite | 1 | N | N |
| Insert course item | 1 | N | N |
|  |  | **Total Cost** | **16N + 1** |
|  |  | **Runtime** | **O(N)** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Tree Data Structure** | **Line Cost** | **# of Times Executed** | **Total Cost** |
| Create tree | 1 | 1 | 1 |
| Add node method | 0 | 0 | 0 |
| IF root is null, add root | 1 | 1 | 1 |
| IF node is less than root, add to left | 1 | N | N |
| IF no left node | 1 | N | N |
| IF node becomes left | 1 | N | N |
| IF node is greater than root, add right | 1 | N | 4N |
| IF no right node | 1 | N | N |
| This node becomes right | 1 | N | N |
| For each line in file | 1 | N | N |
| Create vector course item | 1 | N | N |
| While prerequisite exists | 1 | N | N |
| Append prerequisite | 1 | N | N |
| Insert course item | 1 | N | N |
|  |  | Total Cost | 11N + 2 |
|  |  | Runtime | 0(N) |

Vector:

+Easy to adjust/modify

+Scalable

+Best total cost runtime

+ Highly precise

- More complex data structure

- Not ideal for larger amounts of data

Hash Table:

+ Fast operator

+Efficient data storage

-Data isn’t in order

Binary Search Tree:

+Efficient memory storage

+More balanced

-more complex to modify

I would recommend using the Binary Search Tree since it keeps everything in proper order, unlike the hash table. Also, it offers better memory storage and is an overall more balanced method than the others, at least in my opinion. The Vector is a close second but since it isn’t ideal for larger amounts of data, I don’t think it would be as good of a fit for this project.