**Runtime Analysis Chart**

**Vector Data Structure Runtime as per Pseudocode**

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

**Hash Table Structure Runtime as per Pseudocode**

|  |  |  |  |
| --- | --- | --- | --- |
| Code | Line Cost | # Times Executes | Total Cost |
| Open file | 1 | 1 | 1 |
| For each line in file | 1 | n | n |
| Split line into parts | 1 | n | n |
| Create course object | 1 | n | n |
| Insert into hash table | 1 | n | n |
| **Total Cost** | | | **4n + 1** |
| **Runtime** | | | **O(n)** |

**Hash Table Structure Runtime as per Pseudocode**

|  |  |  |  |
| --- | --- | --- | --- |
| Code | Line Cost | # Times Executes | Total Cost |
| Open file | 1 | 1 | 1 |
| For each line in file | 1 | n | n |
| Split line into parts | 1 | n | n |
| Create course object | 1 | n | n |
| Insert into BST (balanced) | log n | n | n log n |
| **Total Cost** | | | **3n + n log n + 1** |
| **Runtime** | | | **O(n log n)** |

**Data Structure Advantages and Disadvantages**

|  |  |  |
| --- | --- | --- |
| **Data Structure** | **Advantage** | **Disadvantage** |
| Vector | Efficient in memory usage  Its simple to implement and use | Inorder to list courses, sorting is required.  Search for course info is linear |
| Hash Table | Insert and search is fast | Memory usage is moderate  Must extract and sort for listing. |
| Binary Search Tree (BST) | Sorting is automatic  Insert and search operations are balanced in trees | For efficiency, must remain balanced  Higher memory and is complex |