**Runtime Analysis**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **readFile Function** | **Line Cost** | **Number of Times Executed** | **Total Cost** | |
| FUNCTION readFile(File f, lines) | 1 | 1 | 1 | |
| DECLARE courseNumber, courseTitle, prerequisite, line: String | 1 | 1 | 1 | |
| DECLARE I = 0, j = 0: Int | 1 | 1 | 1 | |
| DECLARE bool = TRUE: Bool | 1 | n | n | |
| WHILE (not at end of file) | 1 | n | n | |
| courseInfo = SPLIT (READLINE(f, line),  DELIMETER = ,) | 1 | n | n | |
| APPEND line to lines | 1 | n | n | |
| IF length of courseInfo is less than 2 | 1 | n | n | |
| bool = FALSE | 1 | 1 | 1 | |
| BREAK | 1 | 1 | 1 | |
| courseNumber[i] = courseInfo[0] | 1 | n | n | |
| courseTitle[i] = courseInfo[1] | 1 | n | n | |
| INCREMENT i | 1 | n | n | |
| IF length of courseInfo is equal to or greater  than 2 | 1 | n | n | |
| FOR k = 2 for courseInfo length | 1 | n | n | |
| Prerequisite[j] = courseInfo[k] | 1 | n | n | |
| INCREMENT j | 1 | n | n | |
| IF bool is equal to TRUE | 1 | 1 | 1 | |
| FOR each p in prerequisite | 1 | n | n | |
| IF p is NOT in courseNumber | 1 | n | n | |
| bool is equal to FALSE | 1 | 1 | 1 | |
| BREAK | 1 | 1 | 1 | |
| RETURN | 1 | 1 | 1 | |
| Total Cost | | | | 14n+9 |
| Runtime | | | | O(n) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Vector to Create Course Objects** | **Line Cost** | **Number of Times Executed** | **Total Cost** | |
| FUNCTION for createObject (Courses <Course>, File f) | 1 | 1 | 1 | |
| Lines[] = “ “ | 1 | 1 | 1 | |
| IF readFile (f, Lines) = TRUE | 1 | n | n | |
| APPEND new course (line) TO Courses | 1 | n | n | |
| ELSE PRINT “Error file cannot be read” | 1 | 1 | 1 | |
|  |  |  |  | |
| FUNCTION for dataStructure | 1 | 1 | 1 | |
| INITIATE CLASS Course | 1 | 1 | 1 | |
| DECLARE number, title, prerequisites: String | 1 | 1 | 1 | |
| number = SPLIT (line, DELIMETER = ,) [0] | 1 | 1 | 1 | |
| title = SPLIT (line, DELIMETER = ,) [1] | 1 | 1 | 1 | |
| IF LENGTH of the SPLIT is greater than 2 | 1 | 1 | 1 | |
| Prerequisite = SPLIT (line)[2 to LENGTH of SPLIT (line,  DELIMETER = ,)] | 1 | 1 | 1 | |
| Total Cost | | | | 2n+11 |
| Runtime | | | | O(n) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hash Table to Create Course Objects** | **Line Cost** | **Number of Times Executed** | **Total Cost** | |
| FUNCTION for createObject (Courses <Course>, File f) | 1 | 1 | 1 | |
| Lines[] = “ “ | 1 | 1 | 1 | |
| IF readFile (f, Lines) = TRUE | 1 | n | n | |
| APPEND new course (line) TO Courses | 1 | n | n | |
| ELSE PRINT “Error file cannot be read” | 1 | 1 | 1 | |
|  |  |  |  | |
| Function for dataStructure | 1 | 1 | 1 | |
| INITITATE CLASS Course | 1 | 1 | 1 | |
| DECLARE Number, Title, Prerequisite: String | 1 | 1 | 1 | |
| Number = SPLIT(line, DELIMETER = ,) [0] | 1 | 1 | 1 | |
| Title = SPLIT(line, DELIMETER = ,) [1] | 1 | 1 | 1 | |
| IF (LENGTH of SPLIT(line, DELIMETER = ,) > 2 | 1 | 1 | 1 | |
| Prerequisites = SPLIT(line)[ 3 to LENGTH of SPLIT(line,  DELIMETER = ,)] | 1 | 1 | 1 | |
| Total Cost | | | | 2n+10 |
| Runtime | | | | O(n) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Binary Search Tree to Create Course Objects** | **Line Cost** | **Number of Times Executed** | **Total Cost** | |
| Struct Course | 1 | 1 | 1 | |
| DECLARE courseName, courseNumber: String | 1 | 1 | 1 | |
| DECLARE Vector<String> Prerequisite | 1 | 1 | 1 | |
| Struct Node | 1 | 1 | 1 | |
| SET Course course | 1 | 1 | 1 | |
| SET key for course | 1 | 1 | 1 | |
| IF (element exists for the given key) | 1 | 1 | 1 | |
| RETURN node-> course | 1 | 1 | 1 | |
| IF (no element exists for the given key) | 1 | 1 | 1 | |
| RETURN course | 1 | 1 | 1 | |
| INTIATE CLASS Tree | 1 | 1 | 1 | |
| SET Node root | 1 | 1 | 1 | |
| Void Insert(Course course) | 1 | 1 | 1 | |
| IF (root is null) | 1 | 1 | 1 | |
| root equals newNode(course) | 1 | 1 | 1 | |
| ELSE this->addNode(root, course) | 4 | n | 4n | |
| | addNode(Node\* node, Course course) | | 1 | 1 |  | |
| | IF (node->bidId > 0) | | 1 | 1 |  | |
| | node->right equals new Node(course) | | 1 | 1 |  | |
| | ELSE this->addNode(node->right, course) | | 1 | 1 |  | |
| Total Cost | | | | 4n+15 |
| Runtime | | | | O(n) |