Runtime analysis for Vector Data Structure

void parseFile(string csvPath) function:

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **OPEN csvFile** | 1 | 1 | 1 |
| **IF csvFile found && csvFile size is not zero** | 1 | 1 | 1 |
| **FOR LOOP all lines in the**  **csvFile** | 1 | n | n |
| **READ from input next**  **line of csvFile and**  **parse to file** | 1 | n | n |
| **IF line has less than**  **two parameters** | 1 | n | n |
| **DISPLAY message,**  **course cannot be**  **added** | 1 | n | n |
| **ELSE IF line is**  **greater than or equal**  **to two parameters and**  **course prerequisite is**  **already in file** | 1 | n | n |
| **CREATE new course**  **object** | 1 | n | n |
| **LOOP FOR**  **parameters in the**  **line** | 1 | n | n |
| **ASSIGN Course**  **attributes for**  **each parameter**  **in the line** | n | n | n |
| **ADD the new line**  **to the Course data**  **structure** | 1 | n | n |
| **ELSE csvFile size is zero** | 1 | 0 | 1 |
| **DISPLAY error message file**  **could not be opened** | 1 | 0 | 1 |
| **CLOSE csvFile** | 1 | 1 | 1 |
| **Total Cost** | | | n^2+7n+5 |
| **Runtime** | | | O(n^2) |

vector<Course> loadCoarses(string csvpath) function

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **DEFINE a vector data structure to hold a collection of courses** | 1 | 1 | 1 |
| **INITIALIZE the CSV Parser using the given path** | 1 | 1 | 1 |
| **TRY {** | 1 | 1 | 1 |
| **FOR all rows in the**  **csvFile** | 1 | n | n |
| **IF number of**  **parameters in row is**  **greater than two** | 1 | n | n |
| **CREATE a new**  **course data**  **structure** | 1 | n | n |
| **ASSIGN the new**  **course attributes**  **from the csv file** | 1 | n | n |
| **PUSHBACK the new**  **course to the**  **Courses data**  **structure** | 1 | n | n |
| **ELSE number of**  **parameters in row is**  **less than two** | 1 | n | n |
| **THROW error**  **message** | 1 | n | n |
| **CATCH (csv::Error& e)** | 1 | n | n |
| **DISPLAY any errors**  **encountered** | 1 | n | n |
| **RETURN the course data structure** | 1 | 1 | 1 |
| **Total Cost** | | | 9n + 4 |
| **Runtime** | | | O(n) |

void searchCourse(Vector<Course> courses, String courseNumber)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **FOR all courses** | 1 | n | n |
| **IF the course is the same**  **as courseNumber** | 1 | n | n |
| **PRINT out the course**  **information** | 1 | n | n |
| **FOR each**  **prerequisite of**  **the course** | 1 | n | n |
| **PRINT the**  **prerequisite**  **course information** | 1 | n | n |
| **RETURN course** | 1 | n | n |
| **CREATE a new course object** | 1 | 0 | 1 |
| **RETURN empty course** | 1 | 0 | 1 |
| **Total Cost** | | | n^2+4n+2 |
| **Runtime** | | | O(n^2) |

Runtime analysis for Hash Table Data Structure

void parseFile(string csvPath)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **OPEN csvFile** | 1 | 1 | 1 |
| **IF csvFile found && csvFile size is not zero** | 1 | 1 | 1 |
| **FOR LOOP all lines in the**  **csvFile** | 1 | n | n |
| **READ from input next**  **line of csvFile and**  **parse to file** | 1 | n | n |
| **IF line has less than**  **two parameters** | 1 | n | n |
| **DISPLAY message,**  **course cannot be**  **added** | 1 | n | n |
| **ELSE IF line is**  **greater than or equal**  **to two parameters and**  **course prerequisite is**  **already in file** | 1 | n | n |
| **CREATE new course**  **object** | 1 | n | n |
| **LOOP FOR**  **parameters in the**  **line** | 1 | n | n |
| **ASSIGN Course**  **attributes for**  **each parameter**  **in the line** | n | n | n |
| **ADD the new line**  **to the Course data**  **structure** | 1 | n | n |
| **ELSE csvFile size is zero** | 1 | 0 | 1 |
| **DISPLAY error message file**  **could not be opened** | 1 | 0 | 1 |
| **CLOSE csvFile** | 1 | 1 | 1 |
| **Total Cost** | | | n^2+7n+5 |
| **Runtime** | | | O(n^2) |

void loadCoarses(string csvPath, HashTable\* hashTable

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **INITIALIZE the CSV Parser using the given path** | 1 | 1 | 1 |
| **TRY {** | 1 | 1 | 1 |
| **FOR all rows in the**  **csvFile** | 1 | n | n |
| **IF number of**  **parameters in row is**  **greater than two** | 1 | n | n |
| **CREATE a new**  **course data**  **structure** | 1 | n | n |
| **ASSIGN the new**  **course attributes**  **from the csv file** | 1 | n | n |
| **INSERT the new**  **course to the**  **Courses data**  **Structure**  **(hashTable)** | 1 | n | n |
| **ELSE number of**  **parameters in row is**  **less than two** | 1 | n | n |
| **THROW error**  **message** | 1 | n | n |
| **CATCH (csv::Error& e)** | 1 | n | n |
| **DISPLAY any errors**  **encountered** | 1 | n | n |
| **Total Cost** | | | 9n + 2 |
| **Runtime** | | | O(n) |

void searchCourse(HashTable<Course>courses, String courseNumber)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **CREATE the key for the given course** | 1 | 1 | 1 |
| **ASSIGN node by trying to retrieve the node using the key** | 1 | 1 | 1 |
| **IF no entry exists for the key** | 1 | 1 | 1 |
| **RETURN the course** | 1 | 0 | 1 |
| **IF the entry is found for the key** | 1 | 1 | 1 |
| **DISPLAY course information** | 1 | 1 | 1 |
| **FOR each prerequisite of**  **the course** | 1 | n | n |
| **CREATE tempKey for the**  **prerequisite** | 1 | n | n |
| **ASSIGN tempNode by**  **trying to retrieve the**  **node using the tempKey** | 1 | n | n |
| **IF the tempNode is**  **found for the tempKey** | 1 | n | n |
| **DISPLAY course**  **information** | 1 | n | n |
| **Total Cost** | | | 5n+6 |
| **Runtime** | | | O(n) |

Runtime analysis for Binary Tree Data Structure

void parseFile(string csvpath)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
| **OPEN csvFile** | 1 | 1 | 1 |
| **IF csvFile found && csvFile size is not zero** | 1 | 1 | 1 |
| **FOR LOOP all lines in the**  **csvFile** | 1 | n | n |
| **READ from input next**  **line of csvFile and**  **parse to file** | 1 | n | n |
| **IF line has less than**  **two parameters** | 1 | n | n |
| **DISPLAY message,**  **course cannot be**  **added** | 1 | n | n |
| **ELSE IF line is**  **greater than or equal**  **to two parameters and**  **course prerequisite is**  **already in file** | 1 | n | n |
| **CREATE new course**  **object** | 1 | n | n |
| **LOOP FOR**  **parameters in the**  **line** | 1 | n | n |
| **ASSIGN Course**  **attributes for**  **each parameter**  **in the line** | n | n | n |
| **ADD the new line**  **to the Course data**  **structure** | 1 | n | n |
| **ELSE csvFile size is zero** | 1 | 0 | 1 |
| **DISPLAY error message file**  **could not be opened** | 1 | 0 | 1 |
| **CLOSE csvFile** | 1 | 1 | 1 |
| **Total Cost** | | | n^2+7n+5 |
| **Runtime** | | | O(n^2) |

void loadCourses(string csvpath, BinarySearchTree\* bst)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **INITIALIZE the CSV Parser using the given path** | 1 | 1 | 1 |
| **TRY {** | 1 | 1 | 1 |
| **FOR all rows in the**  **csvFile** | 1 | n | n |
| **IF number of**  **parameters in row is**  **greater than two** | 1 | n | n |
| **CREATE a new**  **course data**  **structure** | 1 | n | n |
| **ASSIGN the new**  **course attributes**  **from the csv file** | 1 | n | n |
| **INSERT the new**  **course to the**  **Courses data**  **Structure**  **(Binary Search**  **tree)** | 1 | n | n |
| **ELSE number of**  **parameters in row is**  **less than two** | 1 | n | n |
| **THROW error**  **message** | 1 | n | n |
| **CATCH (csv::Error& e)** | 1 | n | n |
| **DISPLAY any errors**  **encountered** | 1 | n | n |
| **Total Cost** | | | 9n + 2 |
| **Runtime** | | | O(n) |

void searchCourse(Tree<Course>course, String courseNumber)

| **Code** | **Line Cost** | **# Times Executes** | **Total Cost** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **ASSIGN current node equal to root node** | 1 | 1 | 1 |
| **WHILE current node is not nullptr** | 1 |  |  |
| **IF current node’s**  **course.courseNumber is the**  **same as the passed-in**  **courseNumber** | 1 | n | n |
| **DISPLAY course**  **information** | 1 | n | n |
| **FOR each prerequisite**  **of the course** | 1 | n | n |
| **DISPLAY course**  **number of the**  **prerequisite** | 1 | n | n |
| **RETURN current’s**  **course data** | 1 | n | n |
| **IF the passed in**  **courseNumber is smaller**  **than the current node’s**  **course.courseNumber** | 1 | n | n |
| **ASSIGN current node to**  **be current’s left node** | 1 | n | n |
| **ELSE larger so traverse**  **right side of BST** | 1 | n | n |
| **ASSIGN current node to**  **be current’s right**  **node** | 1 | n | n |
| **DEFINE course (type course)** | 1 | 0 | 1 |
| **RETURN course** | 1 | 0 | 1 |
| **Total Cost** | | |  |
| **Runtime** | | |  |