Run-time analysis for data structures

Vector

Add data from file	Line Cost	# Times Executes	Total Cost
`Define vector`	1	1	n
`Try`	1	1	1
`For each row from the file`	1	n	n
`Create course as Course object`	1	1	1
`Set course.courseCode = file[i][1]`	1	n	n
`Set course.courseName = file[i][0]`	1	n	n
`While (not end of the line`	1	n	n
`Set course.preres =file[i][8] to store and prerequisites the row may have`	1	n	n
`courses.pushback(course) to add data to the vector`	1	n	n
`Catch error`	1	1	1
`print error`	1	1	1
		Total Cost	8n + 1
		Runtime	O(n)

Print Sample Schedule and course data	Line Cost	# Times Executes	Total Cost
`void printSampleSchedule(Vector< Course> courses) {`	1	n	
`for (all course in courses)`	1	1	1
`print course`	1	n	n
`For each row from the file`	1	n	1

Print Sample Schedule and course data	**Line Cost**	**# Times Executes**	**Total Cost**
`if (course has prereq){`	1	n	n
`print prereq}`	1	n	n
`define printCourseData(Vector<Course>courses, courseCode){`	1	n	n
`for all courses in vector`	1	n	n
`if the course code matches`	1	n	n
`print course data`	1	1	1
`for each prerequisite in current course`	1	n	n
`print prerequisite data`	1	1	1
		Total Cost	4n + 1
		Runtime	O(n)

Hash Table

Add data from file	**Line Cost**	**# Times Executes**	**Total Cost**
`Define HashTable`	1	1	n
`Try`	1	1	1
`For each row from the file`	1	n	n
`Create course as Course object`	1	1	1
`Set course.courseCode = file[i][1]`	1	n	n
`Set course.courseName = file[i][0]`	1	n	n
`While (not end of the line`	1	n	n

Add data from file	Line Cost	# Times Executes	Total Cost
`Set course.preres =file[i][8] to store and prerequisites the row may have`	1	n	n
`courses.pushback(course) to add data to the hash table`	1	n	n
`Catch error`	1	1	1
`print error`	1	1	1
		Total Cost	8n + 1
		Runtime	O(n)

Print Sample Schedule and course data	Line Cost	# Times Executes	Total Cost
`Define void printSampleSchedule(Hashtable<Course> courses) {`	1	1	1
`for (all keys in courses)`	1	n	n
`Print courseCode for key`	1	1	1
`If key had prereq{`	1	n	n
`Print prereq}`	1	1	1
`Define void printCourseInformation(Hashtable<Course> courses, String courseNumber) {`	1	n	n
`Set key as course code`	1	n	n
`Set node to the location of the key`	1	n	n
`If the current node matches the key{`	1	n	n
`print course data`	1	1	1

Print Sample Schedule and course data	Line Cost	# Times Executes	Total Cost
`Otherwise{`	1	1	1
`While node exists{`	1	n	n
`If key matches coursecode {`	1	n	n
Print course information}	1	1	1
`Current nodes points to next node`	1	1	1
		Total Cost	5n + 1
		Runtime	O(n)

Binary Search Tree

Add data from file	Line Cost	# Times Executes	Total Cost
`Define Binary Search Tree`	1	1	n
`Try`	1	1	1
`For each row from the file`	1	n	n
`Create course as Course object`	1	1	1
`Set course.courseCode = file[i][1]`	1	n	n
`Set course.courseName = file[i][0]`	1	n	n
`While (not end of the line`	1	n	n
`Set course.preres =file[i][8] to store and prerequisites the row may have`	1	n	n
`courses.pushback(course) to add data to the BST`	1	n	n
`Catch error`	1	1	1

Add data from file	**Line Cost**	**# Times Executes**	**Total Cost**
`print error`	1	1	1
		Total Cost	1n + 1
		Runtime	O(n)

Print Sample Schedule and course data	**Line Cost**	**# Times Executes**	**Total Cost**
`Define void printSampleSchedule(Tree<Course> courses) {`	1	1	1
`While courses has courses to the left`	1	n	n
`Print course}}`	1	1	1
`oid printCourseInformation(Tree<Course> courses, String courseNumber) {`	1	1	1
`set the current node as the root`	1	1	1
`while the current node is not null{`	1	n	n
if the course code matches	1	n	n
`print course information`	1	1	1
`while (prerequisite exists){`	1	n	n
`print course information`	1	1	1
`while (prerequisite exists){`	1	n	n
`print prereq for the course}}`	1	1	1
`if the course code is smaller than the root{`	1	n	n
`set the current node to the left`	1	1	1

Print Sample Schedule and course data	**Line Cost**	**# Times Executes**	**Total Cost**
`otherwise set the current node to the right`	1	1	1
		Total Cost	1n + 1
		Runtime	O(n)