Jason Bonvie

2/16/18

Sparse matrix Determinant project

Brief Description

The implementation of my sparse matrix is a singly linked list. I used this data structure because using the given methods I could traverse and manipulate the elements in the list as needed to get the determinant. I created the linked list from scratch with a default size of 5x5. I then used my addElement method to input the desired data in order from top left to bottom right (of the matrix) given the coordinates. The removeElement, getElementand, minor all traverse the list and get the data needed. The minor method includes the creation of a new linked list that the original matrix is projected into. This list shrinks each time minor is called until it achieves its base case of 2x2 or 1 in the determinant method and the determinant value is returned. When an index that is out of bounds of the scope of the matrix is presented to the methods that contain inputs an error is thrown and the program will exit. (I asked 3 different TA's as to how I should throw and exception 2 of them said the program should throw and exit the program one of them said that I should use a try catch and continue the program. So I chose to use the throw and exit based on this.). 0's are never able to be stored in any linked list.

Computational complexity of methods:

addElement(row, col, data) = O(n)

toString() = O(n^2) (has to traverse all the elements in the list twice)

removeElement(row, col) = O(n)

getElement(row, col) = O(n)

determinant() = O(n!)

minor(row, col) = O(n)

getSize() = O(1)

setSize(size) = O(1)

clear() = O(1)