CS3050 - README

Contributors:

William Jack Boening

James Landy

George Gilmartin

How to run:

make

./maze < input\_file.txt >

Files Included:

Main.c

Solve.c

Maze.c

Maze.h

Solve.h

Makefile

Design Decision & Issues:

We ran into several issues, most came from reading in the file, it posed to be the most difficult aspect of the entire project. We tried multiple different things to try and find the most efficient way of reading in and saving the maze. We tried things such as, reading and finding the longest line (largest amount of characters in a line) and created a 2D array based on the number of lines X the longest line. This would have worked for small mazes, but as a maze grew, it would become more and more inefficient, so we decided to scrap that idea. After much thinking, we finally just decided to create a 2D array and at each row, stored an entire line of the file. This proved to be far more efficient than reading and storing each character one by one.

The graph searching algorithm, first we scan the entire maze to process all of the vertices in the graph as a preliminary action just to verify that there is infact an end point and a start point. Then, we begin to traverse over the 2d array, moving node by node, to visit every node in the 2d array. When we find the exit we keep searching if we have not visited every node in the graph which we check by putting in the nodes we have visited into another array. Once every node has been found we use the exit point we found while traversing the 2d array, the start point, and all the nodes to build the shortest path based on distances. Most of this code is in the solve.c file

As far as contributions go we all contributed equally as we had weekly sessions to meet up and work together along with decide who would do what parts on their own time. Jack came up and coded the best way to read in the file. James coded the search algorithm and all the error checking, and George coded the way to check to see if we had visited a node already. While these were the jobs we had we all contributed in each other’s part.

The code runs in (m\*n)log(n) time.