**Prolog1 Documentation**

My program called Prolog1.pl, upon loading it into the corresponding development environment the program can be initiated by typing at the command prompt.

?- citypath(X1, Y1, X2, Y2, [ [List1], [List2], … ], Path).

Where X1, Y1 represents the starting location in the maze and X2, Y2 represents the end location in the maze. Followed by [ [List1], [List2], … ], which is a list of 0’s and/or 1’s which represents the maze as input where 1 is a wall and 0 is an open space. Lastly, Path which represents the solution path as a list. Below this the program prints out the number of moves and followed by the list of the moves made.

\*\*When running, the program only makes one assertion per printing of the maze for some reason, resulting in one wall being placed each step. I could not figure out. Therefore, in order to get the finished maze, solution and all you will have to ; through to the end and the program will print out the correct solution(s). So, I have it working correctly just could not get my buildWall predicate to make all the assertions needed before moving into my solve predicate.\*\*

The maze representation I chose to use is a mapping of the maze that prints onto the screen. It states that the columns shown are the avenues and the rows are streets then it prints out the number of rows and columns (1, 2, 3, … ) and prints the maze where . is an empty space, x is a wall, and o is the path for the solution. The algorithm or predicate rather I went with is called solve, its parameters are Start, End, Path. Where Start is the start location in the maze in the form [X,Y] and End is the ending location also in the form [X,Y] and followed by the path variable. The solve predicate first checks to see if the starting and ending locations are valid and in the maze. It then jumps into pathFind predicate which accepts the Start and End locations and the Path as parameters and recursively finds a path to the solution. After this solve checks if there exists a shorter path and if not it proceeds to print the maze solution.

I learned honestly a ton about prolog but I still feel I am barely a novice at it because I cannot for the life of me figure out how to get the program to essentially “loop” and make all the assertions creating all the walls before finding and printing the solution. I feel like I have come fairly far but I have quite a way to go in prolog. It is a very different kind of programming that is for sure, it is tough to think about syntactically speaking sometimes.