James Alford-Golojuch

CS463G

October 25, 2016

Prolog Assignment 1

For my program I used SWI for my compiler and programming environment. To simplify my code I used the lists module provided with SWI which is called in the first line of the code. I used the nth1, member, length and append predicates from the lists module. The nth1 predicate checks for a value at a specified position in the list which is what I used to determine whether a spot in the maze was an open space or a blocked space. The member predicate is used to check if a specified value is a member of a list which I used to check to see if a spot has been visited before. The length predicate is used to check if a list is the given length which I use to make sure that the new coordinate values aren’t out of range of the maze. Finally the append predicate checks to see if two lists concatenated are equal to the third list which I used to add on coordinates to the Path.

For my program it starts with a call to citypath() given starting and ending coordinates along with the maze itself which is a list of lists with values of 0 or 1 to signify whether that specific coordinate in the maze is traversable or not. It also has a Path value which is supposed to contain the Path value. I was unable to figure out how to get the Path value to hold the length of the entire path rather it only ends up holding the first value of X1 and Y1. This is a known bug which I believe has something to deal with the recursion in this program but after many attempts I was unable to find a direct solution. I am able to print out the actual path once a solution is found so I am not too sure why that value isn’t returned. Anyway once the citypath predicate is called it adds the first coordinate to Path and then proceeds to call citypath1 which involves the movement of the program where the given coordinate is moved one unit up, down, left or right and then it repeats that until a solution is found in that the coordinate is equal to the desired ending space. At that point the path is printed out along with the length of the path.

For citypath1 the x or y coordinate is increased or decreased depending on which predicate is used. It then checks whether the new coordinate is within the maze still and if so checks if the new coordinate is represented by an open space in the maze. It also checks if the new coordinate has been visited before. If not then it adds that new coordinate to the path. Then citypath1 is called again until a solution path has been found or until no solution is found in which it then returns false.

From this assignment I learned how to program in Prolog which has actually been a lot more fun than I initially anticipated. A weakness in my programming has always been recursion so all of the recursion involved with Prolog has helped strengthen my skill some. I am looking forward to learning more Prolog for the next assignment and maybe even using the language in the future past this class.