Rush Hour Writeup, JJ Sandhu ----- I have declared failure.

**What I did accomplish is as follows:**

1. Established User Input
   1. I created a function that when called, would prompt the user to enter in their board row by row with certain expectations noted in terms of formatting. This would then go into a dictionary of coordinates with corresponding board values. I represented the Target car as X’s, blanks as – ‘s, and other cars as other letters.
2. Created Goal Test
   1. I made a function that checked if the board was in a state of solvency, i.e. was the target car in the correct position.
3. Implemented Board Output
   1. I created a function that simply printed out the state of the board in a readable fashion.
4. Implemented BFS Algorithm
   1. Relatively straightforward, I simply took my BFS from sliding puzzles 2 and changed it a bit in order to give the proper outputs for the Rush Hour problem.
5. Semi-Completed Getting all Children States of a Given Board
   1. Here is where I would say my reason for declaring failure occurs. I did end up finishing a get children function, however I had coded it all in one enormous function and my code was not giving me the proper output.
   2. My approach to getting the children is as follows:
      1. Traverse the Board
      2. Stop at any Car Found
         1. Check for Horizontal/Vertical Alignment
         2. Append any Valid Children to a Set of Children
            1. Horizontal/Vertical decides the constraints of valid children, and the actual moves that a car can make (vertical up/down, horizontal left/right)
6. Created Main Function
   1. This Function Prompts the User for Input, Formats the Input, Runs BFS on the State, and Returns the Path and the Length of the Solution

**Why did I Fail:**

1. In terms of code:
   1. I had not developed my get\_children function with incremental development, which made it that much harder to debug. I wasn’t really thinking long term when I started the function in class, and simply decided to roll with it as I went on since I did not want to have to change my current work. The result of this was a function that did a lot all at once, there were a lot of moving parts that individually could have been throwing off my solution. This was undoubtedly where my code contained its errors. Now actually debugging this would have taken a lot of time, and that would be fine given different circumstances. This leads me to my second reason for failure.
2. Time
   1. The past two weeks have been quite hectic, as teachers all put tests towards the very end of the quarter, resulting in a far greater than normal workload. Aside from this, I have had debate tournaments and MUN conferences occupying my weekends, all in all making for quite a time crunch for everything. I think I really could have finished this assignment as I had gotten a good amount of it done but given little time, I decided the risk of declaring success and not being able to finish in time was not worth it. I declared failure as it wasn’t as risky, and I can finish this assignment on my own time just for fun. I really liked this assignment as it was, I guess grounded in reality (a game I really had played a long time ago), and so completing it would be that much more rewarding. Imagine telling someone who doesn’t really know/care about CS, hey I learned to find the shortest path via Breadth First Search! They wouldn’t really be too intrigued. Saying I can solve any puzzle to this game we used to play in an absurdly little amount of time, although it isn’t the most fascinating thing in the world still would be a lot more interesting.

All in all, I had fun on this assignment and hope to finish it for my own benefit/fun later when I have plenty of time.