I decided to choose the Uniform-Cost Search Algorithm. It seemed like the most obvious choice, especially after we did the Travel-in-Romania example, where you had a start, a goal, and each move had a cost. The algorithm is said to be both optimal and complete (as long as cost>0). The concept was simple enough, choose the least-cost unexpanded node, but implementing a priority queue (in racket) was not. I decided to implement the basic idea where I would pick the least cost square and move to it, if there were more then 1 option, I would pick the square closest to my destination. I thought iterative deepening or even Depth-Limited search because you know which depth limit to go down to (since you know the coordinates of the goal).