Adam Johnson
Assignment 3 A*

The heuristic function used in the first A* program was simple, h(node) = g(node) + d(node), where g(node) was the actual distance from the starting point to said node and d(node) was the estimated cost to the Iron Hills from said node.

The second heuristic function takes the same approach except g(node) is no longer the actual path length from the start to the node, g(node) instead is sum_of_road_condition/sum_of_danger * .2 * total_path_length. This g(node) works interestingly in that increasing the road condition score makes increases the value of the heuristic function while increasing danger decreases the heuristic function meaning the function favors shorter paths that have worse roads and are more dangerous. The .2 was added to ensure the algorithm did not double back on itself.

The third heuristic function takes the same approach except g(node) is no longer the actual path length from the start to the node, g(node) instead is sum_of_danger/sum_of_road_condition * .2 * total_path_length. This g(node) works interestingly in that increasing the danger score makes increases the value of the heuristic function while increasing road condition decreases the heuristic function meaning the function favors shorter paths that have better roads and are less dangerous. Again .2 is used as a constant to prevent the algorithm from going back to a city it was just at.

Solutions From Blue Mountains to Iron Hills:

The first A* Algorithm Path:

1255.0
Blue Mountains
Lake Evendim
Fornost
Rivendell
Caradhras
Carrock
Esgaroth
Iron Hills

The second A* path: (Danger and poor roads)
83.44471153846153 : (Path Length) 1305
Blue Mountains
Lake Evendim
Fornost
Rivendell
North Pass
Carrock
Esgaroth
Iron Hills

The third A* path: (Safe, good roads, and fast)
255.86206896551727 : (Path Length) 1325
Blue Mountains
Michel Delving
Brandy Hall
Bree
Weathertop
Rivendell
North Pass
Carrock
Esgaroth
Iron Hills

The three solutions obviously found three different paths, while the first algorithm only cared about distance and found the fastest solution, the second algorithm had a thirst for poor roads and danger, the function gave us a wildly different path as it chose to go through the North Pass one of the worst roads and most dangerous locations. Even though the path was slightly longer, the heuristic function averaged lower than the thirst algorithm which favored the exact opposite. So the third algorithm focused on finding a safer path with better roads. Ideally the fastest path from the first original solution would be chosen but that went against my expectations, turns out the safer better path is slightly longer that the other two although the safer method still chooses to go through the North Pass which I would have expected it to avoid. As it turns out though the algorithm still puts our travelers in danger and hopefully we can develop an algorithm to avoid the North Pass.