Timothy Ha
junkwan
1367917
junkwan@uw.edu
CSE 373
05.27.15
HW 5
README

1. For adjacentVertices the worst possible running time would be O(V), meaning if the passed in vertex created an edge with every other vertex. This CAN be slightly faster than O(E), because other vertices can still be connected with one another. edgeCost would be the same running time of O(V) in the case that (vertex a) is connected to every single other vertex. The worst runningtime for shortestPath is O(V log(V)). This is because the algorithm has to check over every single vertex to see which path is the shortest, and yes it checks vertices multiple times, but it doesn't check every vertex V amount of times. So instead of V^2 it is V*logV.
2. The majority of me testing my code was by putting println statements throughout my code to see if parts of my program were acting as they should. It's really one of my favorite methods of not only debugging, but seeing my program work step by step. And because our given files were pretty large in terms of number of V and E, I created two additional sets of vertices and edges so I could see what should happen on paper, and what the program did and compare. If the computer did not give the right output or path, then I knew I did something wrong, and with the many println statements I could usually tell pretty easily. I also checked to see in the beginning if I was getting the correct set of vertices and edges, which is very important for the rest of the program.
3. Not applicable
4. Not applicable