CS505: Homework #6

Due on March 26, 2016

Dr. Ravi

Austin Schwartz

Problem 1

- My algorithm is relatively simple. Once the process is initialized, it checks to see if its ID is the highest in the hostfile. If it is, then it declares itself the leader and starts to send messages to all other processes until it crashes or is exited. If the process doesn't have the highest ID, then it waits a certain number of seconds until a message from the highest ID process is receieved. If this message isn't received, then an election is called by notifying any processes with higher IDs than it. During the election, the algorithm attemps to determine which non-crashed process has the highest ID, and attempts to make that process the leader. Overall, the implementation is very similar to the Bully leader election algorithm, which satisfies all consensus properties.
- Unfortunately I had a lot of problems with my implementation. The algorithm still satisfies the consensus properties, but it will occasionally spam output during leader elections. I also have an edge case where all processes agree on say, process 4. Then when 4 crashes and comes back a few seconds later, the other processes will have agreed on process 3, but 4 believes itself to be the leader for a second or two. I believe I could have fixed this small edge case but I didn't have enough time. Something simple would be an additional few seconds as soon as a process starts, to receieve acks from all other processes, to see who is alive and who the agreed upon leader is. At this time, a process just goes off of the highest priority process sending it messages.b