SE3313A – Final Project Design Document

Sam Mallabone – 250844429

Andrew Black – 250834840

Our Goal: To create a game that simulates a game of tag. Each game of tag will accommodate multiple players in the game and synchronize the game play between all of them.

Our intended technology: As of right now, we are planning on completing this project using c++ and a JavaFX front-end. We believe that these two programming languages give us the best opportunity to completing the game with the quality we expect from ourselves. Our knowledge of c++, as developed through the labs, will allow us to use our socket programming knowledge, threading knowledge and general c++ to code the game.

Identified Steps in completing our game. We will not move on to the next step until the prior one has been completed.

1. Be able to connect multiple hosts using socket programming.
2. Synchronize the movements of both hosts characters
3. With the movements synchronized we can then implement the logic for creating a game of tag. Implementing one character to be "it" and upon them colliding with another character, transfer who is "it".
4. Add additional obstacles into the area that the hosts will be playing in. These obstacles will be rigid thus not allowing a character to go through them and they must go around them.
5. If time permits, start creating additional areas that the hosts will play in that could incorporate more hosts and different types of obstacles to avoid.

Due Date: December 8th.

Final comments: Upon each step being completed in our design process, we will add additional features while time permits. This will allow us to have a working prototype early in the design process and allows us a scalable design that will continue to get better as we complete the prior tasks. An additional goal of ours is to have a working game early on and to add to it to make it as polished as can be. This ensures that we have something that is technically sound but also able to provide a fun and inviting user interference.

Additional Questions:

Handling termination: upon the initial time limit for the game ending or all but one host exiting the game, all sockets will be gracefully closed and all remaining players will be invited to play another game. If a host chooses to play again they will open another socket that connects to another game of tag.

Shared Memory: we anticipate needing a shared memory object that will store the (x,y) co-ordinates of the hosts within the game. This shared memory object will synchronize the location of every host among all the UI's of all hosts.

Threads: we anticipate needing one thread to the run the game and one thread per each host playing the game. In each game, there is a maximum of four hosts so if additional hosts join, a new game thread will be spawned allowing the new hosts to spawn threads and connect to it.