**Networks Project Report**

Fraser Loneragan 22243455  
Clayton Herbst enter *student number here*

How would you scale up your solution to handle many more clients?

* Implement a queue. Process each client individually one at a time.
* Multiple servers, all on different hardware.

How could you deal with identical messages arriving simultaneously on the same socket:

* Timeout once first message has arrived
* Identical message check -> inform client of duplicate message and bounce it back

With reference to your project, what are some of the key differences between designing network programs, and other programs you have developed?

* Clients and no clients.
* Multiple processes occurring at once (process id of the different clients plus server).
* **(error checking)** Majority of code is dealing with various situations which arise from your clients. E.g. not enough people joining, cheating, duplicate messages, incorrect messages, client going afk, clients trying to join game when game is already in progress. Only a small proportion of the code is tied to the logic of the game. Unlike other programs I’ve developed where most of the code is logic tied to the goals of the project and a smaller proportion is dealing with errors

What are the limitations of your current implementation (e.g. scale, performance, complexity?

* No graphical interface
* Done on local machine, simulating a network, rather than connections from different hardware in different geographical locations
* Little player choice. No room for player improvement – luck based game.
* Relatively uninteresting
* *We should see what happens when we have 50 people connect, see if it affects performance*
* Can have unlimited scale but will probably be limited by performance of our local computers.

Contribution of each team member