William Yan

15-112 Fundamentals of Programming

**Project Description**

*112.io* (don’t really have a name yet): A 2-d battle royale shooter similar to others found online. You start as a humble circle and collect weapons, gear, and armor to fight other players and Prof. Kosbie to be the last circle standing!

**Competitive Analysis**

Functionally, my game is very similar to other games in the ‘battle royale’ genre, all featuring a last-man-standing win condition with gatherable weapons and equipment. In particular, my game is heavily inspired by surviv.io, an online 2d battle royale game. Though 112.io will be very similar to surviv.io, I plan on adding and modifying many elements to turn the game into my own style. First, there will be a different weapon / inventory system than surviv.io that’ll make core gameplay considerably different. Furthermore, I’ll implement a game AI that can be used for single player / survival style gamemodes and add special 15-112 visual elements to the game.

**Structural Plan**

The final project will be organized into files based on objects and algorithmic type:

* Nature, Weapons, Items, and Player will each represent their respective objects on the game map, mostly extending a core ‘GameObject’ class that has basic drawing and initialization.
* Game.py is the primary file, which depends on Framework.py (copied from the Pygame presentation) and is run to play the game. It will have groups of sprites of all the previously mentioned classes, as well as contain most of the MVC.
* If multiplayer is implemented, all server-side commands will be part of Server.py.
* Game AI will mostly be on one or two files
* All images will be in a separate image folder

**Algorithmic Plan**

* Multiplayer: Multiplayer gameplay will be done using sockets, transferring information between players through a central server. I plan to approach this by having as much information be saved as possible, for example only receiving information about bullet movement and dropped items once before storing it locally. Also, to further reduce the amount of information being sent back and forth, the Player can only request/receive information from players within a certain range, as getting messages from players across the map is unnecessary.
* Game AI: Computer players will be trained using a neural network, most likely with an adversarial minimax algorithm. Elements like movement, shooting patterns, looting patterns, etc. all should be trained and optimized. Bots can be trained against each other to learn the optimal strategies or be manually trained by me.

**Timeline Plan**

* 11/20: TP1 - Basic obstacles, items, weapons, etc. implemented along with most UI stuff
* 11/23: Complete UI, add win condition (last one alive), add ability to use and drop items and customize the player character through skins
* 11/26: TP2 - Complete single player game AI
* 11/30: Implement multiplayer gameplay through sockets
* 12/5: TP3 - Design game map (or a way to randomly generate maps) and polish the UI, gameplay, etc.

**Version Control Plan**

After each session of programming, changes will be saved and also pushed to a Github repository as a backup:  
<https://github.com/wyan1103/surv112.io>

**Module List**

Pygame, Numpy, Sockets (maybe)

**TP2 Update**

Bots can also melee attack and learn through reinforcement learning rather than minimax or a neural network.

**TP3 Update**

* Game mode difficulties – number of bots, speed of bots, and learn rate of bots increases with the difficulty
* Added images to all obstacles and added home screens and death screens
* Added a scoring system
* Modified weapons to have more bullet spread
* Added ability to pause and restart the game