TO: Jason^2, Manager

FROM: Jason, Manager

DATE: September 20, 2018

SUBJECT: Project Plan for Santorini

Part 1: Parts

Santorini will need a Menu representation. This is required to handle starting tournaments, handling the starting of each game in a tournament, recording the result of each game, and then recording or returning the result of the tournament.The game will require a representation of a game board. This board will consist of a six by six matrix of tiles. These tiles must be able to be empty or contain buildings. Additionally, a tile must be able to contain a worker. The game will require a representation of a player and the ability to interact with one other player within the game representation. As it should be autonomous, the player representation should be able to make decisions based on the game, and express these decisions to the game manager. The game must be able to move objects and maintain a current state. There must also be a way to express the current state to a user. The game must also have a set of rules. It must be able to verify that these rules are being followed and enforce them in the case that rules are broken by a player. Rules must exist to determine who starts the game, whose turn it is, if the game is over and if a move is valid, checking for characteristics like that the destination of a move exists and that building height requirements are respected.

Part 2: Potential Implementations

The game's board will be represented by a two dimensional grid of slice of slices of cells. These cells are represented by the Tile struct. This struct will have properties for tracking the number of floors that are on that tile and if a certain and what worker currently is located there. There should be a struct representing a worker. The game will also have a variable for tracking which player's turn it is and once a new turn has been started, which of the player's two workers is the 'active' one. The game will have a Move() function that will take one of a player's workers and the desired destination tile. It will also take an additional tile whose floor count will be increased by one. The function will confirm that the worker is currently the active worker and is a neighbor of the tile. It will also confirm that this addition will not surpass the maximum amount of allowed floors. This function will call a function on the worker to determine its current location. For all of these functions, the game will have to verify that any actions on workers are only taken by the respective player. The game must also have a function for returning a representation of the current game state so that a view implementation can retrieve all of the relevant information it needs to display: This will include a slice of slices of tile structs, the players and their workers (including ways to differentiate the two) and current scores. We propose to implement the player as simply a function from a game's board to a triplet of worker, move position, and add position, together representing a move. This code can be dynamically loaded, if need be, to facilitate ease of user experience. We propose to implement the Menu as a function with a number of stages - this will likely be the *main()* of our implementation. In the first stage, it will gather players for the tournament, however they are passed to the program (via network, STDIN, hard coded, etc), and store them in a slice of players. The second phase will then begin, in which we iterate through this array in two nested for loops, initiating a game between each player with every other (with cases to exclude self matches). After each game is performed, the winner is determined, most likely returned by the game function, and used to increment a map from players to an integer representing their wins. After all games have been played, this is returned to the user, and the function ends.