Title: Sorry! Board Game

Group Members: Brian Noble, Sidhant Puntambekar, Isaac Pyle

Description Paragraph:

For our group project, we will be remaking the board game “Sorry!”. In “Sorry!”, the rules of the game are as follows: There are 2-4 players that attempt to move all of their pieces out of the start space around the board clockwise, and into their color’s home space. Opponents can knock their pieces back into home if they slide through on a slide, land on their exact tile, or draw a ‘Sorry!’ card. The rest of the cards are numbers 1-12, and the player moves the corresponding number that they draw depending on the card, with a few exceptions:

- A ‘Sorry!’ card allows the player to take one of their pieces from home, and move it to an opponent's piece, replacing it on the board and sending that piece back to the opponents home.
- A 1 or a 2 can get a piece out of the player’s ‘Home’ space.
- A 2 lets the player draw another card.
- A 4 requires the player to move a piece backward instead of forward.
- A 7 can be split among the player’s pieces instead of 1 piece going all 7 spaces.
- A 10 can either move a piece forward 10 spaces or backward 1 space.
- An 11 can either move a piece 11 spaces, or swap the player’s piece with an opponent.

We will be creating the game using Java, with JavaFX being the primary library used for graphics. There may be some simplifications made depending on timing, but for now the idea is to implement all of the functionality of the game, with 2 - 4 users taking turns playing the game.

Patterns:
We plan to use the model view controller pattern for interactions between the graphical user interface and the underlying object-oriented backend code. We also plan to use a factory pattern to create the playing cards necessary to move each user’s pawns. We then plan on using a singleton pattern and observer pattern in a similar fashion to the FNMS logger and print output of the game to the stdout. This logger will be eagerly instantiated. If time permits, we will try to use the decorator and strategy patterns respectively in order to create additional piece behavior and specific user behavior.

Language Choice: We plan to use the Java programming language incorporated with the JavaFX graphical user interface (GUI) designer.

Functional Elements:

- Graphics / UI (JavaFX) - Brian
- Player/piece Objects (Using strategy pattern) (generated using factory) - Isaac
- Cards (numbers 1-12 with varying modifications, and sorry card) (generated using factory) - Isaac
- Deck (For the cards) - Isaac

- Board (Containing start zones, safe zones, slides) - Brian
- Turn manager / players - Brian
- Logger (Using observer and singleton pattern) - Sid
- Model View Controller - Sid