Design Decisions for Milestone #1:

* Initially we had planned for an abstract class called ‘Obstacle’ that described the pieces that a rabbit is able to jump over (such that it is easier to determine if the path is clear for the rabbit because we would simply check if the object is an instance of obstacles). We had 3 classes extending Obstacle: Fox, Rabbit, and Mushroom. However, we realized that there isn’t much in common between the animals and the mushroom. Further, the mushroom did not have much functionality and it was not defined by anything besides its location. Therefore, we decided to divide it such that the fox and rabbit are both described by a class called ‘Animal’. This class would store information about whether the animal is a fox or rabbit as well as the animal’s position. We used enumerations to determine exactly which fox or rabbit it is so that it is easy to identify them in the game logic.
* We began to implement the game logic in board which the idea that all game operations are occurring on the board. However, while trying to follow responsibility driven development, we realized that this was giving too much responsibility to the board and the code was not cohesive. As a result, we decided to have a GameEngine class that would combine the pieces and the board to create the game and implement its logic.
* Validating the animals’ moves was done in the GameEngine because it relied on the specifications of the board and the other objects on it.
* Enumerations were used for determining the direction of the move, for the type of square, and for the type of animal because their comparing them was simpler than strings and they were convenient for the toString() methods in animal. Further, they provide a more convenient way of getting the desired animal from the player because we could simply prompt for a number that corresponds to the ordinality of the animal object’s type.