Linus P Go (300345571)
Partner: Casey Xi Yi Huang (300316824)

Brief

This current Assignment involved designing and implementing a specification of Cluedo – the board game, from scratch. It is text-based, which means it relies on keyboard commands from the user to manipulate the games state. I worked with my partner, Casey Huang (ID: 300316284) to implement the game. Our implementation has the work spread over 18 classes, and we have extensively tested our program with 20 Junit tests. It has five packages, which are organized by the contents inside the package.

Design choices

Packages

When coming up with the Initial design, we decided to come up with four packages that the classes should fall under.

Arguments holds classes relating to the Suggesting and Accusing part of the game.

Assets holds miscellaneous parts of the game, such as the Player, Weapon, Room, Position classes.

Cards holds the Card abstract class, the Envelope data structure, and the three sub-card classes.

Main holds the main classes related to the direct execution of the game, such as the CluedoGame and TextClient classes.

I felt like this was necessary, because our amount of classes grew as development went on, and we needed to split it into subsections.

Class Design

We programmed a TextClient class, that we delegated the responsibility of handling user input to the board. The CluedoGame would ask for user input, and would fetch and parse that data from the TextClient.

The Board class holds the textual representation of the Cluedo Board, and has methods relating to drawing the various room, and the game environment. The board class was also used to determine if a player made a valid move.

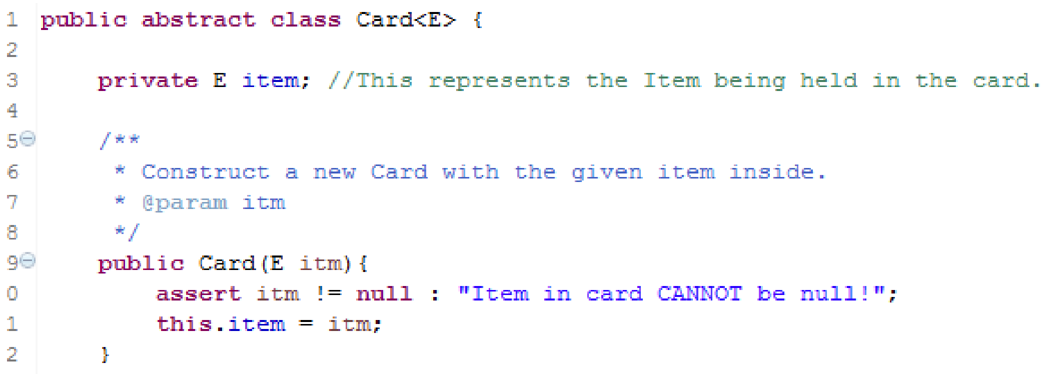
The Initializer class was used to initialize all of the data for the game, such as filling the rooms, initializing the cards, and setting up the players onto the game board.

The CluedoGame is the main game class, which is what is required to run the game. It relies on the TextClient to retrieve data from the user, the Board class to manipulate the state the board, and the initializer to initialize the data that is required. It is a very important class, because it provides the inner logic to the game.

Argument is a superclass that Accusation and Suggestion extend. It holds a quadruple of data, which contain a WeaponCard, RoomCard, CharacterCard, and the current player who is making the accusation.

The Envelope (or the solution card) is a data structure which is backed by an array. I made methods with accessing the data in the envelope, and filling it up.

With the Cards and their objects, we decided to make them separate entities. For example, we have an Abstract class called Card, which holds an Item of generic type E. The three subclasses, RoomCard, WeaponCard and CharacterCard called the superclass Card when created, but was placed in to make the code easier to understand for us.



```
public abstract class Card<E> {

    private E item; //This represents the Item being held in the card.

    /**
     * Construct a new Card with the given item inside.
     * @param itm
     */
    public Card(E itm){
        assert itm != null : "Item in card CANNOT be null!";
        this.item = itm;
    }
```

It allowed statements like this to be made:

Card murderWeapon = new WeaponCard(new Weapon("Dagger"));

That way, it made the hierarchical structure of the cards to be more clear, and that a WeaponCard was a subclass of a Card.

Room for improvement

Casey and I spent a lot of time and effort working on this assignment, I feel like we could have improved a lot of things with it.

Looking back on the design, I personally feel that there are too many classes. While it is good to have low coupling and high cohesion, I feel like my classes are spread too thinly. At one stage, we had 20 classes, because we even included classes like a Dice and a CornerRoom class, when they were just simply small minor thing.

I feel like having the objects and their cards as separate entities made object creation very tedious. My other friends implemented their designs by having only the Cards classes, and they did not bother with separating them.

On the other hand, I feel like this will be helpful for Assignment 2 – because we will be implementing a graphical user interface in Swing.

Finally, I wanted to mention that our partner and I used Git, and we learnt the value of co-ordinating with each other on pulling and pushing commits.