SWEN225 Assignment 1 Reflection

For my reflection I will be splitting the analysis into four parts, as per the assignment brief: assumptions, CRC cards, game logic, and my contributions to the code.

Assumptions

After reading the project brief, and discussing the implementation, there were several assumptions my group decided to make.

- The first assumption was that each player has a physical card on which to keep track of the results of their guesses

This was decided as the implementation of this was optional, but we felt it would confuse the text output and make it unnecessarily long.

- Secondly, we assumed that even though the brief states that a player has the choice on which card to dispute an accusation with, they would not want this as when going off the text-based output, a question such as “Which card did you want to use?” would tell the opponents that the player had more than one of the suggested cards, giving the guesser an unfair advantage.

- We also assumed that players could not move through rooms on their turn to save moves, i.e. entering one door and leaving through a door on the other side in just 2 moves.

- Another assumption was that two players could not be standing on the same tile as each other, so if a person was standing outside a door then a person in the room could not leave through that door.

CRC Cards

I found the CRC cards to be of less use than the UML diagram, but they were still useful nonetheless. They allowed us to get a better appreciation for the links between classes, and this was especially useful during the start of the coding phase, as there were several of us working in parallel on the code. By looking at the CRC cards we had made, it was easier to break the initial coding work into chunks that could be done in parallel without disrupting the other people, for example, I worked on getting the output looking nice, and the initialisation of the board Tile array, whilst Oscar worked on the logic of the game. These two areas had little overlap, and the CRC cards made that easy to see. This strategy didn’t work so well later on, as the CRC cards evolved, and our classes became increasingly interdependent.

They were also used to check the UML diagram design was complete, as the CRC cards more clearly demonstrated the specific connections between classes, so made obvious some connections that were needed in the UML diagram. It was also helpful in identifying redundancies in our code, as there may be several similar methods which became clear to see on the CRC cards.

Daniel Marshall SWEN225 Assignment 1 Reflection 6/08/20
300440531

Game Logic

The game logic for our code was fairly simple, as Cluedo is not a particularly complicated game. We put most of the actual logic about moving, making suggestions and accusations, and winning inside the Game class, as this felt the most appropriate place for it. All of the initialisation, and output was handled mainly by the board class, and through respective toString methods in the tile class. We split the board up into an array of tiles 24 by 25 and split the squares into four types: Hall, Room, Blocked, and Door tiles. As is shown in the UML diagram, these classes are all subclasses of a greater class Tile, which held whether anything was being stored on it, which was crucial for several game points as well as for the output, for which we used a single ASCII character to represent each person and weapon. We also had Items as well as Tiles, which were the things that would appear on Cards, namely the Rooms, Weapons, and Persons. We later refined this to put Persons and Weapons in a subclass moveableItem, as this made the teleporting methods easier. We then had a player class, which contained the hand of the player, their name, and the Person object they were playing as. Each moveable item stored its position in its object, and this was used to find out where a person or weapon was on the board at any time. We also felt it was unfair to give some players more cards, opting to show the extra cards to everyone instead.

Contributions

My contributions to the project were largely in the code, and I helped to come up with the initial UML diagrams as well. My area in the code was the output and initialisation. For the output, I decided I wanted to do a board using ASCII art, and then me and my teammate Felix designed the board in a text document. From there, I then set about coding it. The first problem I encountered was how to represent walls that went horizontally across the board, as opposed to vertically, and for this, I added several lines to the output that were independent of the contents of the board array. I then simply interwove them with the board lines and had an output string to which I appended the first horizontal wall, then the first row of the board array, and so forth. The next challenge was to distinguish between the different rooms and to do that I used an if statement, that put a '@' between any hall tile and room tile, or hall tile and blocked tile, and had nothing put in between two rooms or two blocked tiles, and then had '|' put between hall tiles. The next issue I had was initialising the board, and for that, we created a Text string, and read it in and assigned each tile in the array a new DoorTile, or HallTile, etc depending on what the input character was, and this took a while to figure out as well.

In conclusion, I was very satisfied with my team's work, and think my group produced a very good version of Cluedo. I think we worked well together and am pleased with the final product.