Work Allocation & Design Review Report

Implementing Yote (group 1)

Group 5

Taha Ansari
Devin Dagg
Erik Zorn - Wallentin
Vincent Yong

Work Allocations

Group Member	Classes Contributed	Additional Work Details
Taha	- Board - View - ViewBoard - Piece	- Created work allocation document - Helped Vincent with rules implementation - Worked with other to make various methods work from the following classes: Piece, Player and Cell
Devin	- Cell - Board - Player	- Helped contribute to design review section of documentation - Actively helped team members understand object oriented structures and design implementations - Worked with other to make various methods work including: View, Piece and Viewboard
Erik	- Driver - Player - Piece - ViewBoard - View - Board - Rules	- Erik started the implementation and made the base method and constructors for every class given to us - Helped contribute to design review section of documentation - Created the main method that puts everything together which includes error checking
Vincent	- Board - Rules	- Created github directories for software control - Helped contribute to design review section of documentation - Reviewed comments in all classes

Design Implementation Review

<u>General Issues</u>

Despite the document following good format and having properly made diagrams and session cards, the problem with the design was in it’s fundamental logic. The main method was not a part of the design at all so we did not have a class which we could use as the driver for the program. Another major concern lied in each class having no instance variables so the entire design was very vague and hard to implement without the details that were required to make the program work with all of its required functionality. Our team had a hard time understanding how to link the driver class to the objects that would run the game because it wasn’t specified very well in the design and this caused the game difficult to implement without completely changing the entire design.

<u>Board</u>

Review

The board class had a couple of issues. The lack of instance variables made it hard to understand where and how we could occupy the board with objects. The spec didn’t say anything about making a 2d array of cells but we assumed it and were forced to implement it in order to make all the functions work.

Fixed issues

- the class was called gameBoard initially but the constructor they assigned to the class was named Board so we changed it to just Board
- Had to created a 2d array of the cell objects in order to make all the methods work, was not in the spec but sort of implied with the way they were structuring their classes
- Had to make an object for rules in order to validate most of the methods, also not written in the specs

Unfixed issues

- Board doesn’t have any methods to control the contents of the cell, it just has check cell methods like the spec told us to make

Cell

Review

A simple class to implement. There were no problems with this class

Driver

Review

The main problem was that the design we were given had no way of telling us how the driver class was supposed to control the game. There were no indications of objects that needed to be called or instance variables that were necessary in order to implement the Driver to control the game.

Fixed issues

- Design never specifies how any action is to be inputted, we added an input prompt and basic input interface.
- No main method was in the design, we added it to driver as it made the most sense to put it there
- Document does not specify what Turn Enum values should be, we figured it should be playerOne and playerTwo
- Design doesn't tell us how to implement how to find a new player so we made our own turn checker in the main method

Unfixed issues

- Could not compile properly because methods use a public colour enum that the class does not have
- Driver has no instance variables for declaration of objects
- No game control methods other than players changing turn
- Driver has methods like checkCell which are already implemented in other classes so it was redundant to program it twice

Piece

Review
The piece class was missing a lot of fundamental variables that were required to use the class the way the system was designed.

Fixed issues
- Added private integers to keep track of a pieces x and y location
- Added setters and getter methods for the x and y locations

Unfixed issues
- Colour can't really be used to represent the pieces as we are not allowed to import graphic libraries, this stopped us from being able to display the pieces on the board

Player

Review
The player class was missing a few important variables and had a few logical errors that stopped it from functioning into what the design implied the player class to be used for.

Fixed issues
- There was no way of tracking the current number of pieces a player had on the board which affected gameplay because the indication of how the game would end was when a player runs out of pieces. Added a piece count.
- Made a new constructor for the player class that took in colour and a turn boolean for specifying which player was currently selected.
- fixed grammar syntax error for important method argument (piece toPlace should be Piece toPlace)
- Added setter and getter methods for the current player turn boolean

Unfixed issues
- Methods created but not utilized in the proper class for attacking and moving piece. The attacking and moving methods should be in the driver class or at least the board class

Rules

Review

The Rules class had the right methods and logic but the problem with rules was that there were many limitations from the parameters sent over from the Board class. Many rule checks that should have been done in the Rules class were not specified so there was a lot of error checking that had to be done in the boards class before the Board class could call rules to legitimately check the moves being made on the Board. All rules should have been in one Class and all limitations for the game should have been stated in the spec.

Fixed issues

- Made changes to method logics to work around limitations caused by parameters sent from Board class

Unfixed issues

- we can't check if the first piece selected is the same colour as the current players piece
- No way of really confirming what the status of the current position is in terms of who it belongs to since we can’t use the colour to identify the piece.

View

Review

A very vague class in it’s description. It passes in arguments without really telling us why or how we’re supposed to utilize them.

Fixed issues

- The method for display viewboard had a very vague description in how they wanted us to utilize the board variable being passed as an argument and actually do the displaying so we created our own loops for creating and printing the Board

Unfixed issues

- since we figured the viewBoard class was redundant in displaying the board, updateViewBoard became futile to implement.

ViewBoard

Review

The document was written to implement two board objects one being named viewBoard the other just as board. It would have been redundant to constantly have to update two boards against each other therefore it had to be removed because there was no connection between the two classes except through driver therefore the viewBoard class would have just been a copy of the board class to update and print out simultaneously.

Fixed issues

- did not implement because the class was redundant, we could just display the board using the 2d cells array that held all the piece locations anyways