Test Schedule

The system was tested via the use of the Junit test framework. The code has been designed for testability by separating the game rules into the RuleProvider class which makes all game decisions. This allows the rules to be tested individually without the need to cater for UI dependencies. The table below includes the unit tests that were included to check game rules. Note that these do not cover the entire set of rules due to time constraints

|  |  |  |
| --- | --- | --- |
| **Test Description**  **RuleProviderUnitTests** | **Inputs**  **All tests start with the initialised GameModel** | **Expected Outputs** |
| anyStage\_userBackrupt\_Test  Tests whether the system can detect a player bankruptcy. | Sets balance to negative value and checks that no properties are owned | True |
| anyStage\_negativeBalance\_Test  Tests how the system responds to a player whose balance becomes negative during the processing of a turn. | Sets balance to negative value and presets property ownership including houses | Player option set that includes SellHouse, SellProperty, MortgageProperty and LeaveGame |
| diceThrown\_double\_Test  Tests that setting equal dice values at the DiceThrown game stage and executing the associated instructions causes the doubleThrown flag to be set and an extra turn to be awarded | Sets dice values to be equal. | Checks that the doubleFlag is set and that an extra turn is awarded |
| diceThrown\_double\_double\_Test  Tests that a subsequesnt double throw sends the player to jail | Sets dice values to be equal and sets the doubleFlag | Checks that the current player space is the Jail space |
| movedToNewSpace\_cardSpace\_Test1  Checks that an opportunity card to pay a fine is executed correctly | Places a pay-fine card at the top of the opportunity card stack, puts player on the Opportunity Knocks space and sets the gameStage to MovedToNewSpace | Checks that the first option generated is to pay the fine |
| movedToNewSpace\_cardSpace\_Test2  Checks that an opportunity card to move back 3 spaces is executed correctly | Places a move-back-3-spaces card at the top of the opportunity card stack, puts player on the Opportunity Knocks space and sets the gameStage to MovedToNewSpace | Checks that the player has been moved back 3 spaces |
| movedToNewSpace\_propertySpace\_payRent\_Test1  Checks that landing on an owned property with one house causes the correct rent to be calculated | Moves player to an improved property space owned by another player | Checks that the PayRent option is displayed with the correct amount |
|  |  |  |

Additional tests were added to test the operation of the Game Controller as follows:

|  |  |  |
| --- | --- | --- |
| **Test Description**  **GameControllerUnitTests** | **Inputs**  **All tests except the first start with the initialised GameModel** | **Expected Outputs** |
| loadStaticDataTest  Checks that the game static data is loaded correctly | Calls the LoadStatcData method | Checks that the loaded model contains correct values sample |
| startGameTest  Checks that the StartGame method sets the correct number of users and states | Calls The StartGame method | Checks that the RuleProvider responds with ThrowDice for the first user |
| processUserResponseTest  Checks the operation of the processUserResponse method at the start of a turn | Sets the player to fetch a card and calls processUserResponse | Checks that the correct instruction is contained in the reponse |
| processUserResponseTest2  Checks the operation of the processUserResponse method | Sets the player to fetch a card and calls processUserResponse | Checks that the correct instruction is contained in the reponse |
| processUserResponse\_abridgedGame\_Test  Checks the operation of the processUserResponse method for the abridged game | Sets the abridged game flag and calls processUserResponse | Check for correct reponses |
| getAbridgedGameOverTest  Checks the timeout mechanism for the abridges game | Sets the abridged game flag and calls processUserResponse | Checks that the game times out correctly |
| saveLoadGameTest  Checks the ability to accurately save and load a partially completed game | Sets values in the GameModel | Checks that the values persist after save and load |

Tests were also included to check the operation of the executeInstruction method in the GameModel. A sample test was run but these need to be implemented for every instruction. This was not completed due to time constraints

Tests were implemented to check the basic working of the Agent.

|  |  |  |
| --- | --- | --- |
| **Test Description**  **AgentUnitTests** | **Inputs**  **All tests except the first start with the initialised GameModel** | **Expected Outputs** |
| makeChoiceTest\_ThowDice  Checks that the Agent is able to make random choices from the set of options but always avoids selecting the LeaveGame option | Calls the MakeChoice method | Checks the resulting option chosen |
| makeChoiceTest\_BuyProperty  Checks that the Agent can make a sensible choice about buying property based on having sufficient funds in reserve | Calls the MakeChoice method | Checks that the Agent only buys a property if it has more than £500 |