Comp4004 – Assignment 3

Andrew Dodge – 100938015

My Findings

**1) What you can test with Selenium**

With respect to the course:

The framework Selenium is used for the automated testing of web applications. With Selenium, it is possible to automate every task in your web browser as if a human being was controlling the task. The interface that is used to send commands to the browser is called a Selenium WebDriver.

With respect to the assignment:

**ConnectionBasicsTest:**

* You can test if you can connect to the app
* You can test if you can open the lobby: *missing code 1*
* You can test if you can connect two players: *missing code 2, 3*
* You can test if you can connect multiple players: *missing code 5, 6, 7*
* You can test if you can start the game: *missing code 8*
* You can test if you can disconnect from the game: *missing code 4*

**GameTest:**

* You can test if you can have the AI make their choices: *missing code 9*
* You can test if you can use the stay button: *missing code 10*
* You can test if you can use the hit button: *missing code 11*
* You can test if you can improve cards: *missing code 12*

**RankingTest:**

* You can test if you can rig the game to give the players specific cards: *missing code 13 – 29*

The *italicized* portions refer to the portions of the code where this is tested.

**Strategy1:**

* You can test that the AI holds when they are given a straight or higher
* Possible to manually rig the game to force the else portion of strategy 1

**Strategy2:**

* You can test that the AI holds when they are given a straight or higher
* Possible to manually rig the game to force the else if and else portion of strategy 2

Mainly Selenium is used to test the proper responsiveness of all the buttons in the applications, as well as connecting to the actual application itself.

**2) What you can test with Selenium and Junit**

With respect to the course:

When Selenium and Junit are used together, they can determine if the correct paths were taken in accordance to the logic tests, i.e. its used in accordance to a black and white box testing. Selenium used independently will allow you to predetermine which paths the code execution will take. Though this may be different than the expected outcome. However, when used together it is possible to compare, using asserts from Junit, that the outcome is the same as the expected.

With respect to the assignment:

**ConnectionBasicsTest:**

* Junit is used to test the proper console output is printed for all tests mentioned above in “What you can test with Selenium”
* Used to test if the proper number of human players are connected *missing code 2, 3, 5, 6, 7*

**GameTest:**

* Junit is used to test the proper console output is printed for all tests mentioned above in “What you can test with Selenium”

**RankingTest:**

* Junit is used to test the proper console output is printed for all tests mentioned above in “What you can test with Selenium”

**Strategy 1:**

* Junit is used to test the proper console output is printed for all tests mentioned above in “What you can test with Selenium”

**3) What you cannot test with Selenium by itself**

With respect to the course:

If a webpage has dynamically loaded components that are hidden at first would not be possible to test. As in the webpage is user dependent and displays different content based on which user is currently using the application, this is not possible to test. When a locator cannot identify a new element present Selenium WebDriver comes with a built in Explicit Wait method where it is possible to specify an amount of time to wait before executing a command. If this is the case that should give the web page enough time to load the element in question so it is possible to interact with.

With respect to the assignment:

**ConnectionBasicsTest:**

* If the proper number of human players are connected

**4) What you cannot test with Selenium combined with Junit**

With respect to the course:

Since Junit is a light-weight and easy to use testing framework that helps the user execute and check their tests. With Junit, it is possible to manage user sessions and assert expectation, which allows you to do comprehensive UI tests using Selenium.

With respect to the assignment:

**Strategy 1:**

The only way to test the else statement of strategy one, shown here: “else this AI player attempts to get a full house by exchanging everything that is not a pair or 3 of a kind”, is by hard coding the cards. This does not test the strategy properly as we must use a rigged game to set up this scenario. If we were to use a regular game, there is no way for us to be certain that we will get the hands that are needed to test these scenarios.

When using a rigged game, you give it the starting hands, but you also choose which cards it exchanges. The fact that you choose which cards the AI discards makes the strategy testing irrelevant, as you are doing the work for it. The AI still decides when to HIT or STAY but the actual swapping of cards in done in the written code. Making it impossible to test fully.

**Strategy 2:**

This strategy is not possible with the provided code base, meaning it is not possible to test. The strategy requires the first AI to always follow strategy 1, which as stated above cannot be fully tested. The else if of the strategy posted here: “any player before you has 3 visible cards of the same kind: keep your pair(s) if any, and exchange other cards”. There is no way to test this, as it is not possible to get the AI to see the other players cards and then respond appropriately. For the same reason that is specified above in the strategy 1 area. Finally, the else follows strategy 1, which again cannot be tested properly.

There is also the problem that to even begin to check if they strategy is being followed properly, we would have to rig the game. Rigging the game does not allow for proper testing as we are telling the game what to do and not testing if the AI does it all themselves.

**All AI Game:**

It is not possible to test an all AI game with the code base that we were provided. The code base always requires at least one human player. If there is no human player then there is no admin in the game, meaning the start button will never be pressed and the game will never start.

**5) How you’d go about using Selenium and other testing tools to test your game’s interface AND your game’s logic in a systematic way**

Selenium and Junit do a very good job at testing most aspects of the game and the interface as is. It is simply because the game is coded is a way that prevents full testing that makes it so not everything can be fully tested. Selenium allows for proper running through of the web application, testing all buttons and going from a non-opened game, to connecting to a game, to opening the lobby, to starting the game. Selenium is also used to give all the cards to all the players. Selenium sets up the entire game, and then the code runs the strategies from there. Junit is used to make sure all the console output matches up with the desired result we want from each test. So together they can basically test every aspect. The only things that can’t be tested fully (the strategies) is because the way the code works forces us to rig the game, making us do the strategies for the computer, which in turn does not test them properly.