1.0 INTRODUCTION

We will be unit testing the code for our Blackjack project. We plan to develop a JUnit test suite to test all methods in every class of the Blackjack project.

2.0 OBJECTIVES AND TASKS

2.1 Objectives

Make sure each method in every class is operating as expected. This means that no constructor returns null, parameterized constructors don't return null and contain intended data, mutators & actuators don't return null and return expected data, and all other methods don't return null at any time.

2.2 Tasks

For this project, we will be testing the following classes

- Balance.java
- BlackJackDriver.java
- BlackJackGUI.java
- Card.java
- Dealer.java
- Deck.java
- Game.java
- GameServer.java
- Hand.java
- Player.java
- Rank.java
- Rules.java
- Suit.java
- User.java
- WaitTime.java

3.0 SCOPE

General: All methods listed in 2.2 will be tested.

Tactics: We will be making sure the methods work by simple use of making sure the methods pass a JUnit test and by using assertEquals where applicable.

4.0 TESTING STRATEGY

4.1 Unit Testing

Definition: Minimum requirement is to make sure the functions work. Recommended requirement is to make sure that they work as intended when passing in parameters when applicable and as a whole. This also includes not returning null and, when applicable, return the expected data.

Participants: Ivan Peric, Avin Tiletile, Yew Mun Loon.

Methodology: By use of common JUnit testing and applying a Suite to run multiple test classes at once. We will also look into making sure methods don't return null.

4.2 System and Integration Testing

Definition: System and integration testing is to make sure the program works for the intended systems it's for and it's able to work as intended with all classes involved.

Participants: Ivan Peric, Avin Tiletile, Yew Mun Loon.

Methodology: By use of common JUnit testing and applying a Suite to run multiple test classes at once. We will also look into making sure methods don't return null.

4.3 Performance and Stress Testing

Definition: Make sure that the tests run under 30ms for each method and, where applicable, under different parameters.

Participants: Ivan Peric, Avin Tiletile, Yew Mun Loon.

Methodology: By use of common JUnit testing and applying a Suite to run multiple test classes at once. Where applicable, apply different values when parameter passing. We will also look into making sure methods don’t return null.

5.0 HARDWARE REQUIREMENTS

A capable computer able to run code.

6.0 ENVIRONMENT REQUIREMENTS

6.1 Main Frame

Use Eclipse IDE with at least an Intel i5 quad core equivalent or better processor.

7.0 TEST SCHEDULE

To be finished by 12 May 2020.

8.0 CONTROL PROCEDURE

Problem Reporting

If a problem occurs, please look up how to fix the problem. Do not change your test to pass the code. Changing your code to properly execute and pass the tests is ideal. If problem persists, then please report to whoever is in charge of the section you worked on.

Change Requests

Document your changes and where exactly you made them, then report to whoever is in charge of the section you worked on.

9.0 FEATURES TO BE TESTED

All the methods listed in 2.2 are to be tested.

10.0 FEATURES NOT TO BE TESTED

All features not listed in 2.2 are not to be tested.

11.0 RESOURCES/ROLES & RESPONSIBILITIES

Ivan Peric, Avin Tiletile, and Yew Mun Loon are responsible for the successful carryout of this project.

12.0 SCHEDULES

Deliverables: A unique status report for all members, git logs, screenshots of code in operation, screenshots of test suite runs, project documents, all source code, all testing code, and compiled Java code are all to be delivered in a single ZIP file via Blackboard.

13.0 SIGNIFICANTLY IMPACTED DEPARTMENTS(SIDs)

Cal State East Bay’s Computer Science Department’s Christopher Smith is to be impacted because he’s going to have to grade this.

14.0 DEPENDENCIES

- Must finish by 12 May 2020.
- Project is running on a capable computer to compile.

15.0 RISKS/ASSUMPTIONS

We assume that the user is using Eclipse with correct knowledge of how to JUnit test.

16.0 TOOLS

Eclipse IDE.

17.0 APPROVALS

Must be approved by the following

- Ivan Peric
- Avin Tiletile
- Yew Mun Loon
- Christopher Smith