Integration Testing for HW #4:

For our yahtzee project, we are trying to strategically use integration testing to focus on catching bugs that haven't been detected by the unit tests. The areas we are going to focus on using integration testing the most is the junction between our settings frame and our game play frame, and between our main frame and the end frame. There are a lot of setting features, such as the number of players, in the settings page that are vital to the function requirements for the rest of the code. For that junction, we might test to see if the number of selected players in the startWindow changes the number of player's names to be filled out in the nameWindow. We might also check that the MainWindow cycles through the appropriate number of players received in the startWindow when the user selects the number of players from drop down in the frame before. For the scorecard and main game play through junction, we will test to see that the scorecard has appropriately assembled itself in the endWindow, displaying the rankings of the players with the proper player scorecard data. We will be building more integration tests as we keep working.

When we work on code separately, everytime we try to push back to the repo in gitHub, we will ensure that the tests are passing in JUnit. Everytime we meet together in group meetings, we will also amply discuss bug issues and potential areas where there might be problems. In these meetings, we will also be able to talk through current bug problems. Overall, the most important thing we will do is add onto new code slowly and will test frequently with manually made data and tests. In these tests, the data sets we make to test our code will be inclusive of all test cases, such as testing for zeros, negative values, and other unordinary input data. After producing these data sets, we will then analyze the different structures of the two joint functions and determine the most vulnerable attributes. Afterwards, we will be able to better identify and remedy areas of our code that need to be strengthened.

We also expect that the individuals attempting to add features to the code should have an open dialogue with the person who's code they are attempting to change. Through text, meeting in person, or through git, the people involved in editing and pushing new code should also be communicating with the other owner to address any questions and problems, and to re-write code if it is preferred. We hope that by following the methods outlined above, we can avoid potential bugs, avoid fixing broken code, as well as avoid any frustration as we work together.