Test Plan
OCamlJack

For our test plan, we started off by testing the basic functionality of each of the modules. This included the tests for Deck, Game, Card, and Player modules. These were primarily OUnit tests, but QCheck tests were used for some of the Game module testing because property testing could be used to quickly test large numbers of games. In these modules, glass box testing was used to test the functionality of each function in the module. This allowed us to test edge cases for these functions and ensure they successfully worked regardless of the situation. The exception to this was the QCheck tests in the Game module, which were randomized property testing. Then, the main.ml function was tested manually. Instead of testing this using OUnit, we decided to test it manually because it had visual aspects that needed to be confirmed and all the functionality of the modules called in the main.ml file was already tested in other modules. We believe the test plan proves the correctness of our system because we first used glass box testing to validate the functionality of the functions within modules. This makes sure that the functions within modules work successfully. Then, QCheck testing was used to test the Game functions, which call on other functions in the other modules and QCheck testing validated that the interactions between these functions for a successful gameplay experience worked as intended. A Bisect coverage report was generated, and our test cases covered ~94% of the code over all compilation units, showing essentially every function had been tested. Finally, manual testing was done on the main.ml file, to ensure that the Game functions are being called correctly and the text based terminal interface is working as intended. Through this combination of testing, we believe that we have proved correctness of our game.