Sam Berkson
CPSC 224
HW 2 Summary

Goal: The ultimate goal for this program is to run a game of Yahtzee with a variable number of dice with variable numbers of sides, and output all possible scores at the end.

Overview: I expanded very little beyond my original framework, only needing to adjust it slightly. Since I relied on constructors for all 6 of my classes before, I was able to overload them to take new parameters (number of sides, number of dice) and upload them into new private member variables, accessible through getter methods (and set appropriately throughout member functions). This made refactoring the original rules of Yahtzee die and hands pretty smooth. I ended using a pretty simple workaround to handle the scoring at this point, choosing to iterate over all the possible sides, printing it to denote it to the user and then iterating over the die in the hand to check if any match, adding up the score and printing as we go (line by line). I used my traditional methods for most of the lower scorecard handling, shifting more towards nested iterative loops as opposed to blocks of conditional statements.

Testing: One of my tests checks that my overloaded constructor and getDie() methods work as intended. It creates a new hand, with 6 die (each 1 sided). It then checks all 6 (1 past the default) indices in the hand, checking that they are all one. This ensures that the number of die is being correctly assigned, the number of sides per die is being assigned correctly, and that it is correctly setting the hand length to be 1 longer than it is by default (proving that the default constructor correctly handles the private member variables).

Issues: Whoo boy, there were a lot of them. Most of them arose from me getting ahead of myself and refactoring from the top down, instead of the bottom up. The biggest issue I had

was with refactoring or, more accurately, demolishing and rebuilding my scorecard class. The biggest conceptual challenge I had was with figuring out how to correctly display the number of lines according to the number of sides on the die. After a failed attempt at figuring out how to implement a lambda function for it, I ended up bashing my head against a wall for 2 days until I figured out that I could simply throw some nested loops together and it was smooth sailing from there on out.

Retrospective: I definitely need to spend some more time with my scorecard. It was more of a stepping stone for HW2, but I didn't have enough time to fully implement scorecard past printing and more in the realm of storage and accessibility. I will also need to take another look at my game loop, as there are a few fringe cases that seem to make my program very angry. I've got them itemized and I've got leads on all of them, but I still haven't quite figured out where I'm going wrong here. That's an issue for future me to deal with.