Tyler Yasukochi

Gonzaga University

CPSC 224 - HW #1 Summary

Zag Farkle Rolling and Scoring

Program Goal: To develop a single round game of Farkle. The program will generate and roll dice, then add the values to the user's hand. The user will then be able to choose which dice they want to add to their meld, or if they want to end the round. The program will also automatically detect whether the user hits Farkle, which would then end the game.

This project was my first time coding in Java, so I started outputting a user friendly menu. This task was straightforward, and it helped me get familiar with the syntax.

Next, I created an array list for the hand, which would generate and roll the dice, assign the values, then sort and return the array. Once I was getting the proper output, I created a function to detect whether the user rolled a Farkle.

Then, I focused on implementing more of the menu functions. This is where an issue came up because when the program would call the hand function to print it, it would regenerate the dice. To avoid this, I created another array list called "currenthand" and made it equal to the hand function. Admittedly, even now this feels redundant as I could've just moved the dice generation into main and have it iterated through once at the beginning of the game.

At this point I created a function for the meld which would read the user's input and would run through switch cases that act accordingly. Ex. when the user inputs "c", it takes the value stored in hand(2), moves it to meld(2), and clears hand(2).

Here I needed to create another function called "toggledie", because using switch cases made it so I could only move the dice from hand to meld, and not from meld to hand. The toggle dice function was simply an if, else-if statement to account for this, which again, seems redundant and I probably could've figured out a more efficient way to get around this.

I then implemented a print function, that would iterate through the arrays with a for-loop and print the menu. I also included the score for later down the line.

Again, I had issues here because you can't print an empty/semi-empty array, and my code up to this point didn't account for this. I solved this by storing the value "-1" in the arrays, converting the arrays from integer to string in the print function, and replacing all instances of "-1" with a space, " ".

Now I implemented the scoring system by creating a score function, that would define the possible combinations and calculate the user's score based on their meld. I approached this by first creating and initializing multiple variables to track the occurrence of each value. Then I would iterate through the meld, and using switch cases, would increment the matching variable

Ex. If number in meld == 1, “int countone” would increment. I would use these values to calculate the user’s score.

Looking back, I recognize a lot of this program could’ve been optimized, had I thought of other solutions, instead of just sticking with a certain strategy and trying to make it work. I also hardcoded significant portions of this program that didn’t need to be. Overall, I feel I did well, especially since this was my first time ever coding in Java. It was a learning curve, filled with googling syntax. However, applying strategies and logic I learned from other classes made this project very doable and manageable.