Jessica Li

April 2nd 2018

OPIM 3223

Yahtzee Reflective Paper

Working on the project taught me that having a good and logical structure, and wholly understanding your project before starting it is really important. I was not familiar with the game Yahtzee when I first started the project so I decided to work through the project and code parts of the game I understood first. At first, most of the problems I had were trying to figure out the logic behind the “check” functions. I knew I could hard code every category, but I wanted to make the program as efficient and non-redundant as possible (this later becomes a problem for tracking). I knew that for many of the checks it would be easier to reorder the hand in ascending order. I used stack overflow to find the sort array function and how to roll random dies. I had the most difficulties figuring out the logic for kinds and straights, but through many test runs, I eventually figured it out.

To test my code, I would print out the “count” variable and insert print statements between each section to see if it was working. I also realized that the game required a lot of player input and if they type invalid answers, the program wouldn’t work correctly. Therefore, I had to create several while loops that checked for invalid answers. I had to look up how to restrict user input to a number because an invalid answer would crash the program. Stack overflow taught me to use TryParse. After a few days, I thought I had finished my project. Everything seemed to work until I reread the instructions to the game Yahtzee.

There were two main things that I didn’t account for in my program: you can only choose one category once and the bonuses for uppers and Yahtzee’s. These issues were essentially the same issue: I have to keep track of the categories and store scores from those categories. This became the biggest problem for my code that I was unable to solve. I knew that because I tried to make my code efficient (by grouping many categories into one function, especially the uppers) that it would be hard for me to keep track of the turns and categories that were used. I thought of using global Boolean values and many if statements, but I couldn’t get it to work because of the structure of my program. I spent a lot of time trying to figure out how to fix this issue without rewriting my program, but in the end, I realized that the only simple solution is to have a “check” function for all 13 categories and build my program around that. Therefore, in my game, you can choose the same category multiple times and it does not calculate the bonuses. I know that if I had kept this mechanic in mind before I had started my project, it wouldn’t be a problem.

I also had to rewrite my “reroll” function because I misread the Yahtzee rules. At first, I thought you only rerolled one die per turn, so my “reroll” function asked for only a single die to reroll. Later I realized that the player could reroll some or all of their dies each turn, so I had to go back and account for that. At first, I was trying to use an array or list that took in their responses then rerolled those corresponding dies, but I found it too difficult to do. Instead I used a clunkier, but easier way by asking them if they wanted to reroll each of their die.

My biggest problem was that I started my project with no general structure and hoped to piece things together later when I understood the game more. However, when I understood the game more, I was “in too deep” to go back and change everything. Thus, I learned that you should understand your project fully and have an overall outline of what to do before you start your project. I also learned that sometimes having the most “efficient” code can sometimes make your project more complicated later on. For example, lumping many of the check functions made it difficult to track their outputs individually later on. Although it would be redundant and longer, having 13 check functions would have been easier to track and score for the game.

Lastly, I didn’t completely understand the web scraping part of the assignment, so I just included some text about the game and its price after the game ended. If I were to web scrape, I would create a function that creates a data table that would include the price of game and the description. I would add a row named “Yahtzee”. I would use the HTMLAgility package to parse the website information. I would copy the URL and use web.load to access the website. Then I would hit “f12” on the website, click “inspect nodes” and copy the XPaths to figure out what nodes connect to the price and description. I would call this function at the end of the game.

Sources:

<https://stackoverflow.com/questions/1019793/how-can-i-convert-string-to-int>

<https://msdn.microsoft.com/en-us/library/e78f86at(v=vs.110).aspx>

<http://grail.sourceforge.net/demo/yahtzee/rules.html>

<https://stackoverflow.com/questions/5359540/random-numbers-for-dice-game>

<https://stackoverflow.com/questions/5395630/how-to-validate-user-input-for-whether-its-an-integer>

<https://www.youtube.com/watch?v=4cPPD-MFadQ>

<https://www.codeproject.com/Articles/1041115/Webscraping-with-Csharp>