Advent of Code 2022 Day 2

Sareh Jalalizad

Spring Term 2023

Introduction

For this assignment, we were supposed to solve a puzzle of the Advent of Code 2022. This report provides a solution for the Advent of Code Day 2 challenge. On Day 2, the challenge involves calculating the score value based on a set of input data. The input data contains two columns of characters, the first column is what the opponent is going to play which is "A", "B", and "C", while the second column is "X", "Y", and "Z". The problem requires two parts to be solved. In the first part, the total score is calculated by adding the score for the shape selected (1 for Rock, 2 for Paper, and 3 for Scissors) and the score for the outcome of the round (0 if lost, 3 if the round was a draw, and 6 if won), while in the second part, the total score involves adding a different set of numbers to the outcome of the round based on a different set of rules for the second column.

Method

The solution to Day 2 involves writing a program in Elixir that reads the input data from a file, processes it to calculate the total score value, and then output those value to the console. The program is divided into several functions, including a solution for part 1, part 2, and getInput function to read the data from a file.

The partOne function takes in a list of inputs and maps over each element in the list, splitting each into a list of two elements. It then maps over and performs a case statement on each element to determine a value to be associated with it. For each case, a value is assigned based on the strategy guide for calculation, and if neither of the cases match, a default value of 0 is assigned. All values are then summed up and returned as the output of the function.

The partTwo function in the code is very similar to partOne, but with a different strategy for calculating the total score. In this part the second

column says how the round needs to end: X means need to lose, Y means need to end the round in a draw, and Z means need to win. So here, we need to figure out what shape to choose so the round ends. For example for "A","X", the opponent will choose Rock (A), and we choose Scissors (C) to lose (X), so the score is $3 + 0 = 3$.

```
def partTwo(input) do
    input
    |> Enum.map(&String.split/1)
    |> Enum.map(
        &case &1 do
          ["A", "X"] -> 3
          ["A", "Y"] -> 4
          ["A", "Z"] -> 8
          ["B", "X"] -> 1
          ["B", "Y"] -> 5
          ["B", "Z"] -> 9
          ["C", "X"] -> 2
          ["C", "Y"] -> 6
          ["C", "Z"] -> 7
          _ -> 0
    end )

    |> Enum.sum()
end
```

After running the program, the total score for part 1 and part 2 are output to the console, which are 10994, and 12526 respectively.

Conslusion

Overall, this assignment was interesting. This problem highlights the importance of data structures and pattern matching in Elixir and demonstrates how these features can be used to solve problems.