Advent Of Code

Aditya Gupta

February 2023

1 Introduction

In this assignment, I will solve the questions associated with Day 3 of the Advent of Code challenge.

2 Part 1

2.1 Description

In this part, we want to find the character which is common between the first and the second half of each line of input text. For example, consider the first line in the input text:

wgqJtbJMqZVTwWPZZT

First Half: *wgqJtbJMq*
Second Half: *ZVTwWPZZT*

Here we find that the character w is the common character in the 2 halves of the line. According to the question a priority has also been created for all characters in the following way:

- Lowercase item types a through z have priorities 1 through 26.
- Uppercase item types A through Z have priorities 27 through 52.

This means that for the common letter we obtain for this line has a priority of 23. For this task, we are required to find the sum of the priorities for all of the lines we see in the input text.

2.2 Implementation

The approach to solving this question is quite simple. We start by reading the input. We then go through all of it, line by line and split each line into 2 halves. These halves are then split into a list of character strings which we put into

MapSets. Then we take the intersection of them to find the common character. We convert the intersection to a list which we convert into just a charlist later with the help of Enum.map and List.flatten(). We then take the integer value from the charlist and check its value to determine if it's upper or lower case and then return the priority based on the value number. We do this recursively using the loop1() function and using the accumulator argument in it we get the

```
def loop1([], priority) do priority end
def loop1([head|tail], priority) do
    listParts = Tuple.to_list(String.split_at(head,trunc(String.length(head)/2)))
    loop1(tail, priority + extractPriority(listParts))
end

def extractPriority(listParts) do
    part1 = MapSet.new(String.graphemes(Enum.at(listParts,0,nil)))
    part2 = MapSet.new(String.graphemes(Enum.at(listParts,1,nil)))
    map = MapSet.to_list(MapSet.intersection(part1,part2))
    [value] = map |> Enum.map(&String.to_charlist/1) |> List.flatten
    cond do
      65 <= value and value <= 90 -> value - 38
      97 <= value and value <= 122 -> value - 96
    end
end
```

3 Part 2

3.1 Description

In this part, we want to find the character which is common between groups of 3 lines from the input text. For example, consider the first 3 lines in the input text:

wgqJtbJMqZVTwWPZZT
LHcTGHQhzrTzBsZFPHFZWFFs
RnLRClzGzRGLGLGCNRjTMjJfgmffSffMqNgp

Here we find that the character T is the common character in the 3 lines under consideration. We follow the same priority rules as before. This time the priority of the common character is given by 46. We are basically supposed to sum up the priorities the same way we did in the previous part but this time for the groups of 3 lines. We would then repeat this entire process with the set of the next 3 lines of input and so on.

3.2 Implementation

The approach is not very different from before. This time the 3 lines of text that we look at are put into a MapSet, after which we take their intersection in order to find the common character. We then extract the priority for the character using the same approach as before, and we sum them up by recursively calling the loop2() function.

```
def loop2([], priority) do priority end
def loop2([h1,h2,h3|rest], priority) do
    part1 = MapSet.new(String.graphemes(h1))
    part2 = MapSet.new(String.graphemes(h2))
    part3 = MapSet.new(String.graphemes(h3))
    intersection = MapSet.intersection(part1,part2)
    intersection = MapSet.intersection(intersection,part3) |> MapSet.to_list
    loop2(rest, priority + priorityFromIntersection(intersection))
end

def priorityFromIntersection(intersection) do
    intersection = intersection |> Enum.map(&String.to_charlist/1) |> List.flatten
    [value] = intersection
    cond do
      65 <= value and value <= 90 -> value - 38
      97 <= value and value <= 122 -> value - 96
    end
end
```

4 Results

These are the results that I get for the input that I got from the Advent of Code website:

- Part 1 Sum: 7727
- Part 2 Sum: 2609


Advent of Code [About] [Events] [Shop] [Settings] [Log Out] Aditya Gupta 12*
λy.2022 [Calendar] [AoC++] [Sponsors] [Leaderboard] [Stats]
25
24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6 **
5 **
4 **
3 **
2 **
1 **