

Lab 7 - Advent of code 2022 1

Weichen Chai

Spring 2023

Introduction

This paper seeks to answer the questions presented in "Advent of code 1" for ID1019. In this assignment, we will explore how the functional language Elixir can be utilized to process quizzes and tasks given by our teachers on a website named "adventofcode".

Tasks 1

In this task, we are asked to find which elf carries the most calories as well as the sum of the amount of calories they are carrying. It is visible that the calories carried by each elf is separated by a gap in the text shown below.

2000

3000

4000

In the case above, the first elf carries $2000+3000 = 5000$ calories in total, and the second elf carries 4000.

In order to approach this question, we first find a method to separate elves from one another, we know that they are separated by a gap, therefore we can use it to identify when one elf's inventory has been searched when head is equal to an empty string.

Another aspect we would want to consider is that we want to import the list given in this assignment to be used in our program. This can be done with the `file.read` function, which returns the result of things inside the file, however, with gaps in the middle replaced by sigils, therefore, we go with an approach discussed in class.

```
txt = File.read!("data.txt")  
//Running this function we realize that the numbers  
//are split by "\r\n", so we use a String.split
```

```
readable = String.split(txt, "\r\n")
//removes all sigils between numbers.
```

To make it computable, we need to read all the values into a list, running the functions in the terminal, we see that there is an empty string indicating change of elf, so we can create another list containing summed calories for each elf by:

```
def sumList([], acc, sum) do [acc | sum] end
// the first being used for gap
def sumList([head | tail], acc, sum) do
//acc for accumulator, which is used for sum for each elf.
```

The important thing to take note of is the integer function parse, which formats string to an integer directly, this is handy when we want to execute calculations on those numbers.

```
case head do
  _ ->
    //in the case when head contains something
    {head, _} = Integer.parse(head)
    // When the parse function is used, the result usually is
    //a tuple containing the intager and
    //the leftover non-integer content of the input, therefore
    //but in this case we only keep the intager.
    sumList(tail, acc + head, sum)
    //This indicates the addition of accumulator.

...

```

Another case is when an empty string is encountered, and the program simply isolates the accumulator and proceeds with sumList function with the tail.

In order to get the elf with the highest calories, there is a handy function for enumerating, which is : **Enum.max** This function simply returns the largest number in the list, which is : 69795.


Tasks 2

Moving on to task 2 section of the assignment, we are asked to find the top three elves carrying the most calories. This part isn't too difficult if we continue using the **Enumerable protocol**. For example, we can sort the list in descending order, then just pick the top three and add them up.

```
fin = Enum.sort(list, :desc)
//Sort the list in a decending order, from large to small
three = Enum.take(fin, 3)
//take the first three elements in the list
Enum.reduce(three, 0, fn x, acc-> x + acc end)
// Add all of them up with accumulator.
```

The result we receive in the end is 208437, which is the correct answer according to the assignment website.

The challenge is completed as indicated below with a screenshot of completion(The teacher said it was fine to include a screenshot)

A screenshot of a dark-themed interface showing a puzzle completion message. The text is displayed in a monospaced font. The first line says "Your puzzle answer was" followed by the number "208437" which is enclosed in a light-colored rectangular box. The second line says "Both parts of this puzzle are complete! They provide two gold stars: **".

Your puzzle answer was 208437.

Both parts of this puzzle are complete! They provide two gold stars: **