Assignment 2 - Eliezer Molina Mello

Part 1

Kamal was my partner assigned. Kamal's Repository.

Part 2

Paraphrase the problem in your own words.

The problem revolves around finding any missing numbers from a list of integers that can range from `0` to `n`. In order to do so we need to write a function that can properly identify them. Also, if all numbers are present, the function must return `-1`. As a heads up, a list may contain *duplicate* integers.

Create 1 new example that demonstrates you understand the problem. Trace/walkthrough 1 example that your partner made and explain it.

New example:

Input: `lst = [1, 7, 0, 3]`

Output: `[2, 4, 5, 6]`

Kamal's example and walkthrough:

Input = `[8,9]`

Output = `[0,1,2,3,4,5,6,7]`

The example Kamal presented was interesting as he only listed the last two numbers of the list. The function should then be able to undertand that the missing numbers are all integers from `0` to `7`.

Copy the solution your partner wrote.

```
import numpy as np
import pandas as pd


def missing_num2(lst) -> int:
    if lst:
        rlst = np.arange(0,max(lst)+1,1)    #O(n)
        lst = np.array(lst)                 #O(n)

        diff = np.setdiff1d(rlst,lst).tolist() # O(nlog(n)) becuase setdiff1d needs to sort

        if not diff:
            return -1

        return diff  #O(m) m<n

    else:
        raise AssertionError('input list is empty')
```

Explain why their solution works in your own words.

```
# Output the time it takes to run the below code
%time

# Testing Kamal's solution using the example I provided
l1 = [1, 7, 0, 3]
missing_num2(l1)
```

```
     CPU times: user 3 µs, sys: 0 ns, total: 3 µs
     Wall time: 7.39 µs
     [2, 4, 5, 6]
```

Kamal's solution works great as he used a straightforward, yet very effective, code.

First, he created a range of number from `0` to the `max` number found in the list. Then, he converted the input list into a numpy array. A very important step he used was `numpysetdiff1` which finds the difference between the complete range and the input list, that is how it identifies missing numbers.

Lastly, if there are missing numbers, it returns them, otherwise, it returns `-1` if no numbers are missing. Also, he added an `AssertionError` to display an error message if an empty list was provided.

Explain the problem's time and space complexity in your own words.

In terms of time complexity:

- The `np.arange` functions has a time complexity of `O(n)` as it creates an array from `0` to the maximum number in the list where `n` is the length of the range.
- Now, `np.setdiff1` is more complex as it needs to sort the array and then retrieve the difference/missing numbers. This implies a `O(nlog(n))` complexity due to sorting. However, since both lists, the complete list `m` and the input list `n` need to be sorted, then the **overall time complexity** would be `O(mlog(m) + nlog(n))`

In terms of space complexity:

- Creating an array and converting the input list requires space proportional to the number of values in each, therefore, `O(m + n)` would be the appropriate **space complexity** as it considers storage needed to store data structures temporarily.

Critique your partner's solution, including explanation, if there is anything that should be adjusted.

There are couple suggestions I have based on everything learned in class:

- Kamal's code and alternative solution heavily depends on Numpy. Salaar mentioned that for more complex or heavier problems we should look for other sources or packages that can effectively handle large quantities of data.
- Following my first point, using `np.arange` to generate a complete range from 0 the maximum number in the list could be memory intensive if the maximum number is very large. This can be optimized by analizing the list more efficiently without generating a full range. This can also affect the space complexity negatively due memory consumption.
- Another related suggestion is to find a way to optimize the algorithm by not having to sort the list every time we run the code. For instance, using data structures like `hash sets` to track numbers can potentially be a good option as if would find missing numbers without generating a full range.
- Regarding Kamal's explanation and reasoning behing his code and logic, while his code is relatively straighforward, I'd suggest adding more comments to more, or even each, line of code. For someone new or unfamiliar to algorithms and/or numpy having few comments makes the code harder to understand. However, Kamal provided quite a comprehensive explanation to both his actual and alternative solution.

Part 3

Please write a 200 word reflection documenting your studying process from assignment 1, and your presentation and reviewing experience with your partner at the bottom of the Juypter Notebook under a new heading "Reflection." Again, export this Notebook as pdf.

Reflection

My studying process in order to understand and complete assignment 1 was as follows. First, I had to re-watch some of the class recordings, review the lessons' slides, and read more definitions and explanations on the web. Then I started writing some test code and the part where I spent most of my time was creating the `if` statements as I have never coded anything similar before. What also helped me was to search for additional help on `stackoverflow` and other forums. Then, I started testing my code with my own examples to verify whether it gave me the correct output. This took quite a few trials and errors but in the end I was able to effectively generate the correct output.

My reviewing experience was not as time-consuming as my studying process for assignment 1 as I had already some idea of the topics and code Kamal, my partner, used. Still, I had to review some class content as well as get help from the web to further understand Numpy arrays and particularly the `setdiff1d` function which I had never seen before. Thanks to my partner's great explanation I also learned more about arrays, and solidified my understanding of algorithms as well as time and space complexity.

Overall it was a very enjoyable experience both completing my assignment and reviewing my partner's solution.

End of asignment 2.