Part 2:

Create a Jupyter Notebook, create 6 of the following headings, and complete the following about the your partner's assignment 1:

1) Paraphrase the problem in your own words.

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

3) Copy the solution your partner wrote.

4) Explain why their solution works in your own words.

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

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

My partner is Ben Ho. He worked on question 3 (given below).

The linke to solution he provided is: https://github.com/SplitInf/algo1/blob/main/algorithm_assignment1.pdf

Question Three: Missing Number in Range You are given a list containing n integers in the range [0, n]. Return a list of numbes that are missing from the range [0, n] of the array. If there is no missing number, return -1. Note, all the integers in the list may not be unique.

Examples Example 1 Input: lst = [0, 2]

Output: [1]

Example 2 Input: lst = [5, 0, 1]

Output: [2, 3, 4]

Example 3 Input: lst = [6, 8, 2, 3, 5, 7, 0, 1, 10]

Output: [4, 9]

Question 1) Paraphrase the problem in your own words.

Answer: for the given list of integer as a input, create a range of integers from 0 to the maximum number in list. Now from this range of integers, provide all the missing integers as output. list may also contain duplicates. if there are no missing integers in the range, provide -1 as output.

Question 2) Create 1 new example that demonstrates you understand the problem. Trace/walkthrough 1 example that your partner made and explain it.

Answer:

Example 1:

Input: lst = [1, 2, 2, 2, 4, 6]

expected output: [0,3,5]

Example made by partner Ben Ho

Input: `lst = [1, 2, 3, 4]`

Output: -1

Explanation: I think Ben Ho made some error here. The list contains values from 1 to 4. However, It does not contain '0'. so output should be '0'

Question 3) Copy the solution your partner wrote.

Original solution provided by Ben Ho.

```
def missing_num(nums: lst) -> int:
  #create sorted list
  lst_sorted=sorted(lst) #O(n^2)
  #create full range of numbers with first and last object
  full_range=range(lst_sorted[0], lst_sorted[-1], 1)

  #create new list to store output
  lst_missing=[]

  #iterate though full_range and return items if it is not in input list to new list
  for i in full_range: #O(n)
    if i not in lst:
      lst_missing.append(i)

  #return new list is size is not 0, else returns -1
  if len(lst_missing) > 0:
    return(lst_missing)
  else:
    return(-1)



#test 1
lst = [5, 0, 4, 12, 9]
missing_num(lst)
```

```
[1, 2, 3, 6, 7, 8, 10, 11]
```

```
#test 2
lst = [1, 2, 3, 4]
missing_num(lst)
```

```
-1
```

Question 4) Explain why their solution works in your own words.

Answer: first we sort the list. --> than we creat a range of list with lowest number in list to highest number (the error here is we are missing '0' as default lowest value). --> than we creat a enpty list as lst-missing --> now we are running for loop to identify missing integers in given list and storing them in lst_missing, to creat a list of all missing values. --> now we are providing missing value as output. --> except in scenario when list is empty.

Question 5) Explain the problem's time and space complexity in your own words.

Answer:

Time Complexity: there are 3 factor that affacts time complexity:

1) sorted function: this used Timsort algorith which have time complexity of O(n log n) (Source: https://stackoverflow.com/questions/14434490/what-is-the-complexity-of-the-sorted-function)

2) range function which have time complexity of O(m)

3) for loop which have time complexity of O(m)

However, in worst case, the tome complexity depends on sorted function hence over all time complexity will be O(n log n)

Space xomplexity: there are two factors which affects space complexity

1) sorted finction which depends on number of total input. so space complexity will be O(n)

2) range function which depends on only highest value in list, since default lowest is 0. so space complexity will be O(1)

So in worst case, space complexity depends on sorted function. Hence over all space complexity will be O(n)

Question 6) Critique your partner's solution, including explanation, if there is anything should be adjusted.

Answer: Ben Ho did great job explaining code and provided simplified version of it. However there are some improvements that can make code better.

1) first of all time complexity for sorted function is incorrect. it would be O(n log n) instead of #O(n^2). (Source: https://stackoverflow.com/questions/14434490/what-is-the-complexity-of-the-sorted-function)

2) the code does not assume 0 as lowest value in range. but it assumes 1 as lowest value in given range. So the range of given list is [1,n] instead of [0,n]

Alternate solution provided by Ben Ho:

alternative solution using set difference

1. given input integer 'lst', create sorted range of numbers
2. get min max from sorted range and create 'full_range'
3. apply difference() of full_range on lst

I have corrected error where missiong '0' is also included in output

```
def find_missing_numbers(nums):
    # creating a set of input numbers
    nums_set = set(nums)

    # getting maximum and minimum numbers
    min_num, max_num = min(nums), max(nums)

    # creating reange from 0 to max number
    full_range = set(range(0, max_num + 1))

    # finding missing numbers
    missing_numbers = [num for num in full_range if num not in nums_set]

    # returning missing numbers if any or returning -1 if no missing numbers
    return missing_numbers if missing_numbers else -1

# testing Example 1
lst = [1, 2, 2, 2, 4, 6]
# expected output [0,3,5]
result = find_missing_numbers(lst)
print(result)
```

```
[0, 3, 5]
```

Part 3:

Question: 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

Answer:

I found all 3 questions for assignment-1 a bit challenging. Especially question related to data structures. On the way to finding solutions, I ended up learning more.

For my assignment-1, I was trying to find the solution for Q1 which is related to finding duplicates in data structure. I think that my code was perfect until I received feedback from Tina. Where I realize that there is error in my code. Which can identify duplicates in the data tree. But cannot tell if the duplicate is closest to the root. Definitely I felt disappointed. But most importantly, I learnt that I forgot to test my code in different scenarios. Now I understand important of unit tests and quality control in catching buggy codes.

Than I started working on my partner's code Ben Ho. Ben Ho was working on question 3 which is related to finding the missing integers from the given list. My partner understood the assignment very well and Ben Ho found very simple and effective solution. My partner provided proper comments to explain each line of codes and also provided alternate solution. However there was small error in code because the code assumes the list [1,n] and not [0,n] which means if 0 is not provided in the given list, the output does not contain '0'.

I think this is small part that was missed and I was able to improvise it using alternate solution.

Overall, it was great brain exercise!