```
#    Practice Interview
## Objective: The partner assignment aims to provide students with the opportunity to practice coding in # an interview context.
# You will analyze your partner's Assignment 1. Moreover, code reviews are
# common practice in a software development team. This assignment should give you a taste of the code # review process.
## Group Size: Each group should have 2 people. You will be assigned a partner
## Outline
### Part 1: You and your partner should send to each other your Assignment 1 submission.
### 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.
# The problem wants us to create a function. And this function will take a list of integers, from 0 to n.
# The function is required to find and return the missing numbers inside the range. Incase of no missing
# number, it should return -1. Keep in mind that the integers in the input list may not be unique.

# 2 CREATE 1 NEW EXAMPLE THAT DEMONSTRATES YOU UNDERSTAND THE
# PROBLEM. TRACE/WALKTHROUGH 1 EXAMPLE THAT YOUR PARTNER MADE AND
# EXPLAIN IT.
# Code which includes new example and explanation

from typing import List

def missing_num(nums: List[int]) -> List[int]:
    # The max value in the list
    max_num = max(nums)

    # Missing numbers list
    missing_numbers = []

    # The range from 0 to the max value
    for i in range(max_num + 1):
        # If the current number is missing from the list
        if i not in nums:
            missing_numbers.append(i)

    #  If exist any missing numbers
    if len(missing_numbers) == 0:
        return -1
    else:
        return missing_numbers


# Given examples
print(missing_num([0, 2]))
print(missing_num([5, 0, 1]))
print(missing_num([6, 8, 2, 3, 5, 7, 0, 1, 10]))
```

```
[1]
[2, 3, 4]
[4, 9]
```

```
# My New Example
print(missing_num([1, 2, 3, 3, 5, 6]))
# Missing numbers are [0, 4]

# Explanation for partner’s third example with the list [1, 2, 3, 4, 5, 10]
print(missing_num([1, 2, 3, 4, 5, 10]))
# The integer range is 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.
# 0 is missing; (1, 2, 3, 4, 5) are present; (6, 7, 8, 9) are missing; 10 is present.
# Missing numbers are [0, 6, 7, 8, 9].
```

```
[0, 4]
[0, 6, 7, 8, 9]
```

```
from typing import List

def missing_num(nums: List[int]) -> List[int]:
    miss_numbers = []
    max_value = max(nums)
    expected_set = list(range(0, max_value + 1))

    for i in expected_set:
        if i not in nums:
            miss_numbers.append(i)

    if not miss_numbers:
        return [-1]
    else:
        return miss_numbers


# 4 EXPLAIN WHY THEIR SOLUTION WORKS IN YOUR OWN WORDS.
# The solution works by finding the maximum value in the input list. This info helps determine the range of numbers.
# It considers through each number from 0 to the maximum value. For each number, it
# controls whether that number exists in the input list. If a number doesn't exist in the list, so it's missing,
# and it adds that number to a list of missing numbers. At the end, if there are any missing numbers found,
# it returns that list. If there no missing numbers, it returns -1.

#  5 EXPLAIN THE PROBLEM’S AND SPACE COMPLEXITY IN YOUR OWN WORDS.
# Time Complexity: The time complexity depends on the size of the input list and the maximum value in
# that list, which described as n. The time complexity can be considered as O(n), where n # is the size of the input list.
# Space Complexity: The space complexity depends on the size of the output list that stores the missing
#  numbers. If all numbers from 0 to n are missing in the input list, then the output list would have n+1
# numbers. So, the space complexity is O(n).

#  6 CRITIQUE YOUR PARTNER'S SOLUTION, INCLUDING EXPLANATION, IF THERE IS
# ANYTHING SHOULD BE ADJUSTED.
# More verbal explanations could be added in the code, in order to make it more easy to understand.

### 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
# In assignment 1, I learned how to define a binary tree node, and I learned the (‘is_symmetric”) function
# to find the closest duplicate value. I understood the left child node and right child node concept. I
# created a binary tree, and assigned it to the root. The “is_symmetric” function works, and the
# result is printed. If there is a duplicate, It prints the value, If not, it prints “no duplicate values
# found”.
# In assignment 2, I learned how to read someone else’s code. I learned how to find the maximum number
# in the input range, and then find the missing numbers.
# Both assignments helped me to understand the main concepts of algorithms / data structures,  how they
# work. And I realized once again that it requires a lot of study, practice.
```