Challenge B: Most Frequent Number

Problem Domain:
In this challenge, I will declare a function that takes an array and returns the number that appears the most times in the array.

Algorithm:
Step 1:
Create an empty dictionary to count how many times each number appears.Challenge A: Array Reversal:

1. maxCount ⟶ to store the highest count found during the loop.
2. mostFreq ⟶ to store the most frequent number in the array.

Step 2:
Loop through each number in the array:

- If the number is already in the dictionary, increase its count by 1.
- If it's not in the dictionary, set its count to 1.

After updating the count, compare it with maxCount.

- If the current number's count is greater than maxCount, update maxCount and store the number in mostFreq.
-

Visualization:
Let's say the input array is:[4, 2, 4, 3, 4, 2, 1]
Step 1:
Initialize:
counts = {}
maxCount = 0
mostFreq = null
Step 2: Looping through the array

4 ⟶ not in counts, set counts[4] = 1
⟶ maxCount = 1, mostFreq = 4

2 ⟶ not in counts, set counts[2] = 1
⟶ no change to mostFreq

4 ⟶ already in counts, increment to counts[4] = 2
⟶ maxCount = 2, mostFreq = 4

3 ⟶ not in counts, set counts[3] = 1
⟶ no change

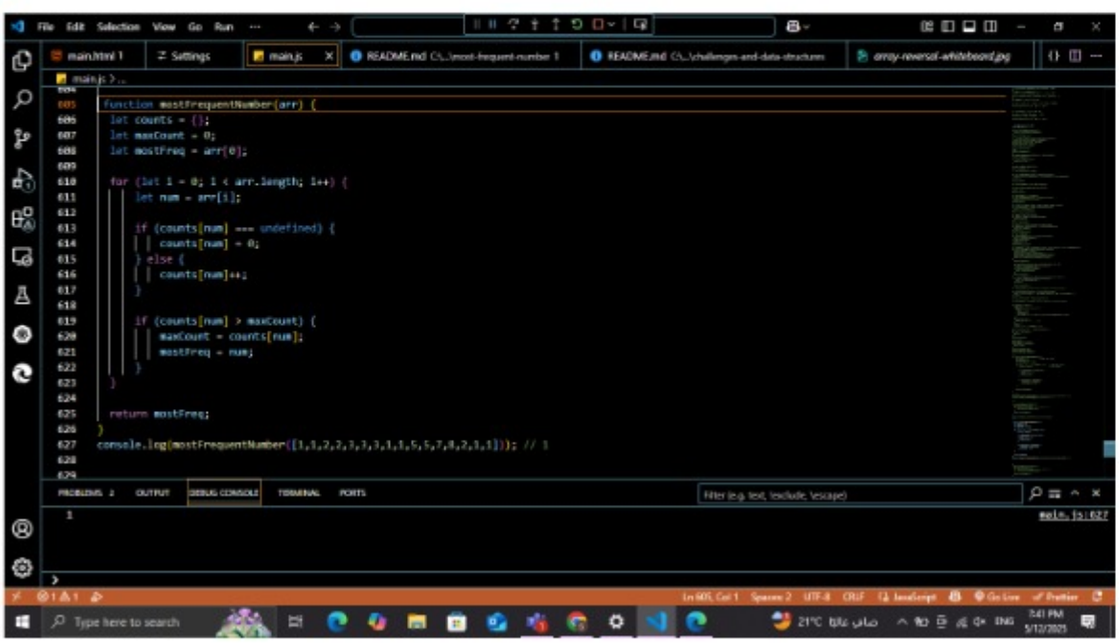
4 ⟶ counts[4] = 3
⟶ maxCount = 3, mostFreq = 4

2 ⟶ counts[2] = 2
⟶ no change

1 ⟶ not in counts, set counts[1] = 1
⟶ no change

Final Output:
mostFreq = 4

Code:



Time Complexty =O(n)