Naive Solution

Make a frequency vector of size n to store frequency of elements at fq[elem]

Traverse through the fq vector
elem with 0 freq → Missing elem
elem with 2 freq → Repeating elem

T.C → O(n) S.C → O(n)

Optimised Solution

→ Use the tortoise-hare method to find out the repeating element

ref :: (2_4_duplicate_in_n+1_integer_array)

→ We know $Sum(n) = \frac{n(n+1)}{2}$

if there were no missing elem, sum of array would be

original_sum = n(n+1) / 2

→ Find current_sum of array

This current_sum contains repeated elem, remove it

curr_sum = curr_sum - repeated_elem

→ Now, Missing element will be.

missing_element = original_sum - curr_sum

Dry Run :

[4, 3, 6, 2, 1, 1]

repeated_elem using slow-fast p/rs

repeated_elem = 1

original_sum = n(n+1)/2 = (6 × 7)/2 = 21

curr_sum = 4 + 3 + 6 + 2 + 1 + 1 = 17

remove the repeated element

curr_sum = 17 − repeated_elem = 16

Missing_elem = original_sum − curr_sum

= 21 − 16

= 5

5 is the missing element.

T.C → O(n) S.C → O(1)