39. Find the Missing and Repeating Number

Tuesday, May 2, 2023 11:30 AM

arr = [4, 3, 6, 2, 1, 1] n = 6

n = 6

→ 1 to 6

[1 2 3 4 5 6]

✓ ✓ ✓ ✓ ✓

Repeating number = 1 , missing number = 5

1) 1st Approach

- loop from 1 to N.
- Check how many time the element is present

  if it appear more than 2 times then repeating,

  if it missing then return the missing.

Pseudo Code:

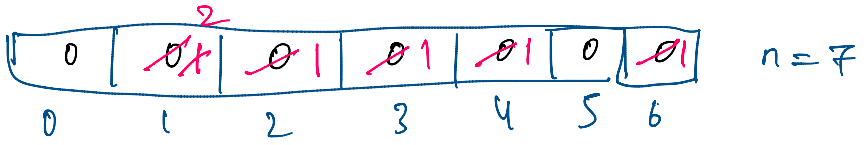
```
repeating, missing = -1;
for (i = 1 ⟶ n)
{
    cnt = 0
    for (j = 0 ⟶ n-1)
    {
        if (arr[j] == i)
            cnt++;
    }
    if (cnt == 2)
        repeating = i;
    else if (cnt == 0) missing = i;
    if (missing != -1 && repeating != -1)
        break;
}
```

T.C ⇒ $O(n^2)$ S.C ⇒ $O(1)$

2nd Approach - Hashing:

arr = [4 3 6 2 1 1] n = 6

- Create a hash array of size '7', as n=6 you need index of '6'.



- Hash array should be '0'. then iterate over the array and mark them in the hasharray.

- Now as the nos is from 1 → 6, start iterating from hasharray. if the cnt == 2, add it to the repeating number and if the cnt == 0 then it's a missing number

```cpp
vector<int> findMissingRepeatingNumbers(vector<int> a) {
  int n = a.size();
  int hasharray[n + 1] = {0};

  for (int i = 0; i < n; i++) {
    hasharray[a[i]]++;
  }

  int repeating = -1, missing = -1;
  for (int i = 0; i <= n; i++) {
    if (hasharray[i] == 2) {
      repeating = i;
    } else if (hasharray[i] == 0) {
      missing = i;
    }
    if(repeating != -1 && missing != -1){
        break;
    }
  }
  return {repeating, missing};
}
```

T.C ⇒ O(2N)

S.C ⇒ O(n)

3) Optimal Approach (Maths + XOR):

arr = [4 3 6 2 1 1] n = 6

Using Maths:

x → repeating

y → missing.

- Sum all the element $S = (4+3+6+2+1+1)$ and

even sum the 'n' $\Rightarrow 6 \Rightarrow S_n = (1+2+3+4+5+6)$

sum of n natural nos $\Rightarrow \frac{n \times (n+1)}{2}$

- Now substract,

$$S - S_n$$

$$= (4+3+6+2+1+1) - (1+2+3+4+5+6)$$

$$= 1 - 5$$

$$= -4$$

1 $\rightarrow$ it's actually 'x'.

5 $\rightarrow$ it's actually 'y'.

But while substracting you don't know it's 1 and -5.

you will simply get -4.

But on observing. we known it' is.

so we take it as eq ①

$$x - y = -4 \quad \text{---} \quad (1)$$

as we cannot solve eq(1), so we need eq(2)

So, for that we will sum up the square of S_2^2 and S_n^2 and the substract

S_n^2 — sum of first n natural nos

$$S_2^2 - S_n^2 \qquad \frac{n \times (n+1)(2n+1)}{6}$$

$$(1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2) - (1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2)$$

$$= i^2 - 5^2$$

$$= 1 - 25$$

$$= -24$$

$$x^2 - y^2 = -24$$

-4 → $(x-y)(x+y) = -24$

$$x + y = \frac{-24}{-4} = 6$$

$x + y = 6$ —— (2)

$x - y = -4$

$x + y = 6$

$+$ ——————

$2x = 2$

$x = 1$ repeating

$1 - y = -4$

$y = 5$ missing

```cpp
vector<int> findMissingRepeatingNumbers(vector<int> a) {
  long long n = a.size();
  // S - Sn = x - y
  // S2 - S2N

  long long SN = (n * (n + 1)) / 2;
  long long S2N = (n * (n + 1) * (2 * n + 1)) / 6;

  long long S = 0, S2 = 0;
  for (int i = 0; i < n; i++) {
    S += a[i];
    S2 += (long long)a[i] * (long long)a[i];
  }
  long long val1 = S - SN; // x - y
  long long val2 = S2 - S2N;
  val2 = val2 / val1; // x + y

  // find x
  long long x = (val1 + val2) / 2;
  //   find y
  long long y = x - val1;
  return {(int)x, (int)y};
}
```

T.C => O(n)

S.C => O(1)