Set Mismatch

The set `S` originally contains numbers from 1 to `n`. But unfortunately, due to the data error, one of the numbers in the set got duplicated to **another** number in the set, which results in repetition of one number and loss of another number.

Given an array `nums` representing the data status of this set after the error. Your task is to firstly find the number occurs twice and then find the number that is missing. Return them in the form of an array.

Example 1:

```
Input: nums = [1,2,2,4]
Output: [2,3]
```

Note:

1. The given array size will in the range [2, 10000].
2. The given array's numbers won't have any order.

Solution 1

Idea is to compute the sum mathematically first, and subtracting the elements from it.
Find the duplicate element, and add that to sum.

```java
public int[] findErrorNums(int[] nums) {
    Set<Integer> set = new HashSet<>();
    int duplicate = 0, n = nums.length;
    long sum = (n * (n+1)) / 2;
    for(int i : nums) {
        if(set.contains(i)) duplicate = i;
        sum -= i;
        set.add(i);
    }
    return new int[] {duplicate, (int)sum + duplicate};
}
```

written by jaqenhgar original link here

Solution 2

```java
public int[] findErrorNums(int[] nums) {
        int[] arr = new int[nums.length+1];
        int a=0,b=arr.length;
        for(int i: nums) arr[i]++;

        for(int j=1;j<arr.length;j++){
            if(arr[j]==2) a=j;
            if(arr[j]==0) b=j;
        }
        return new int[]{a,b};
    }
```

written by aayushgarg original link here

Solution 3

Count each element. We know the original elements must have been `1, 2, ..., len(A)`.
Once we have the counts, it is easy to scan through and see which element must have occurred twice, and which one never occurred.

In our implementation, we could also use `collections.Counter(A)`.

```
def findErrorNums(self, A):
    N = len(A)
    count = [0] * (N+1)
    for x in A:
      count[x] += 1
    for x in xrange(1, len(A)+1):
        if count[x] == 2:
            twice = x
        if count[x] == 0:
            never = x
    return twice, never
```

Bonus solution: Say `(x, y)` is the desired answer. We know `sum(A) - x + y = sum([1, 2, ..., N])`, and `sum(x*x for x in A) - x*x + y*y = sum([1*1, 2*2, ..., N*N])`. So we know `x-y` and `x*x-y*y`. Dividing the latter by `x-y`, we know `x+y`. Hence, we know `x` and `y`.

```
def findErrorNums(self, A):
    N = len(A)
    alpha = sum(A) - N*(N+1)/2
    beta = (sum(x*x for x in A) - N*(N+1)*(2*N+1)/6) / alpha
    return (alpha + beta) / 2, (beta - alpha) / 2
```

written by awice original link here

From Leetcoder.