

Problem Definition

給定一大小為 $n+1$ 的整數陣列 $nums$, 設其中所有數皆在 $[1, n]$ 範圍中, 有其中一數為重複出現的, 欲找到此重複數。

Example: $nums = [1, 2, 3, 2, 4]$

$\Rightarrow output = 2$

Solution:

I. Hash Table 方法

利用 - hash table 儲存所有見過的元素。
遍歷 num 查找，若有，則為 duplicated element
若無，則插入 hash table。

Time Complexity, $O(n)$

Space Complexity: $O(n)$

II. 排序法 + brute-force search

將 $nums$ 先排序後, 從頭 traverse 陣列,
即可找到 duplicated element.

Time Complexity : $O(n \lg n) + O(n)$

Space Complexity : $O(\lg n)$, 設使用 quick sort

IV. 排序法 + Binary Search

將 $nums$ 先排序後, 利用 Binary Search 找到 duplicated element.

Binary Search 怎樣設計?

$nums = [1, 1, 2, 3, 4, 5, 6]$

設以第 n 次出現的 index 為 output index

可發現: 在解右方包括解, $nums[idx] = idx$
左方, $nums[idx] > idx$

Variation: 設重複數不止重複一次要如何設計?

nums = [1 1 1 2 3 4 5]

設以第^二次出現的 index 為 output index

可發現：在解右方包括解， $nums[idx] \leq idx$

左方， $nums[idx] > idx$

但該算法會 modify nums, 如何不 modify nums

且 $O(1)$ space 下做到。

Time: $O(n \lg n) + O(\lg n)$

Space: $O(\lg n)$.

IV. 解法解法

設共 $n+1$ 個數且所有 $1 \sim n$ 數皆出現至少一次，

則可用： $\sum_{i=1}^{n+1} \text{num}[i] - \frac{n(n+1)}{2}$ 得到

Time: $O(n)$

Space: $O(1)$

V. 標記法

∵ 已知所有數皆為正且共有 $n+1$ 數且範圍為 $[1, n]$

∴ 可將 $nums$ 中的值當做 $index$ 來看。

Algorithm: 遍歷 $nums$, 在 iteration i 時, 將 $nums[nums[i]]$ 轉為

負號, 若檢查 $nums[nums[i]]$ 為負, 則表示

$nums[i]$ 為 duplicated \neq

Example: $nums = [1, 1, 2, 3]$

Iter 0 : $nums[0] = 1$

$\Rightarrow nums = [1, -1, 2, 3]$

Iter 1 : $nums[1] = 1$

$\Rightarrow nums[1] = -1 \quad \therefore 1$ 為 duplicated \neq

Time : $O(n)$

Space : $O(1)$

但該 method 會改變原 num, t_2

VI. Array as Hash Table

Intuition: 設 - 無 duplicated num 應為 $[-1, 1, 2, 3, \dots, n]$

即: $num[i] = i, \forall i = 1, \dots, n$

\therefore 目標要將 duplicated element 移至 $num[0]$, 若有
 $num[num[0]] = num[0]$ 時, 則為 duplicated.

Example, $[3, 1, 2, 3]$, $\because num[num[0]] = 3 = num[0]$

$\therefore 3$ 為 duplicated.

$[1, 3, 2, 3]$, $\because num[num[0]] = 3 \neq num[0]$

$\therefore 1$ 不為 duplicated.

Algorithm: 只要 $num[0] : num[num[0]]$

則交換 $num[0]$ 与 $num[num[0]]$

Example: $nums = [1, 2, 3, 4, 5, 6, 6]$

①. $[2, 1, 3, 4, 5, 6, 6]$

②. $[3, 1, 2, 4, 5, 6, 6]$

③. $[4, 1, 2, 3, 5, 6, 6]$

④. $[5, 1, 2, 3, 4, 6, 6]$

⑤. $[6, 1, 2, 3, 4, 5, 6]$

⑥. $\Rightarrow 6$ #.

Time: $O(n)$

Space: $O(1)$

但 該 method 會 改變 原 $nums$ #

VII. Another Binary Search Method

Intuition: 設無 duplicate 的 num_1 下, $\text{num}_1 = [1, 2, 3, 4, 5]$

其中, number n 共有 n 個數在 num_1 中 \leq 於 n , 但在有 duplicated 的情況下, 則在 duplicated number 以上的數, 不滿足該性質。

\therefore 滿足在 n 個數中共有多於 n 個數 \leq 於 n 的
最小的數, 即為 duplicated number.

e.g. $\text{num}_1 = [4, 6, 4, 2, 1, 4, 3, 5]$

$\Rightarrow 1$: 共有 1 個數 \leq 於 1

2 : 共有 2 個數 \leq 於 2

3 : 共有 3 個數 \leq 於 3

4 : 共有 6 個數 \leq 於 4

5 : 共有 7 個數 \leq 於 5

6 : 共有 8 個數 \leq 於 6

∴ 可利用該性質做 Binary Search, 找 duplicated number

Binary Search 如何設計?

∴ 已知, num 範圍必在 $[1, n]$ 間,

∴ 計算 $1 \sim n$ 中, i 時在 num 中共有幾個數比 i 小

Ex: num = [4 6 4 2 1 4 3 5]

count = [1 2 3 6 7 8 8]

∴ count 陣列為 monotonic, 故可以在 count 中做 Binary Search

找滿足性質最大的數。

Time: $O(n)$ $O(n \lg n)$

Space: $O(n)$ $O(1)$

其中: ①. 預先計算並儲存 count.

②. 在 Binary Search 時, 要用到再計算 count.

VIII. Bit Operation Method

Intuition: 又觀察 - 無 duplicate 的 num 的性質:

Example: ①. $num = [1, 2, 3, 4, 5]$

其中用 bit 表示: $1 = 001$

$2 = 010$

$3 = 011$

$4 = 100$

$5 = 101$

將每個表示式中, 各個 bit 和為: 223

再來看 - 有 duplicated element 的例子:

$num = [2, 1, 2, 2, 3, 4, 5]$

其中用 bit 表示: $2: 010$

$1: 001$

$2: 010$

$2: 010$

$3: 011$

$4: 100$

$5: 101$

将每个表示汉中, 各个 bit 和 差: 243

$\therefore 243 - 223 = 020$, 可知只有 bit 1 变动

\therefore duplicated number 必为 $010 = 2$

② $nums = [3, 1, 3, 3, 3]$

其中用 bit 表示: 3: 011

1: 001

3: 011

3: 011

3: 011

将每个表示汉中, 各个 bit 和 差: 045

而原 $nums = [1, 2, 3, 4]$

将每个表示汉中, 各个 bit 和 差: 122

$\therefore [0, 4, 5] - [1, 2, 2] = [-1, 2, 3]$

\therefore Duplicated element 只会增加, \therefore 考虑对 $[0, 2, 3]$

的貢獻，即為 $011 \Rightarrow 3$ 。

Algorithm:

- ①. 計算 $1 \sim n$ 中各 bit 和 為 x
- ②. 計算當前各 bit 和 為 y
- ③. $y-x$ 中, positive bit 設為 1, 反之為 0
- ④. 最終的 $y-x$ 即為 duplicated number

Example:

[2, 1, 2, 2, 3, 4, 5]

- ①. 計算: [1, 2, 2, 4, 5, 6] 各 bit 和:

$$\begin{array}{r} 001 \\ +010 \\ +011 \\ +100 \\ +101 \\ +110 \\ \hline 333 \end{array}$$

$$x = 333$$

①. 計算 $[2, 1, 2, 2, 3, 4, 5]$ 各 bit 40.

$$\begin{array}{r}
 010 \\
 +001 \\
 +010 \\
 +010 \\
 +011 \\
 +100 \\
 +101 \\
 \hline
 243
 \end{array}$$

$$\therefore y = 243$$

②. 計算 $y - x = -110$

可知，結果為 $010 = 2$ 也，

Implementation: 不可能有 333 in binary representation,

必须得 Iterate over k 次, 每次计算在 $y-x$ 的第 k 个 bit 是否 positive.

Time: $O(n \cdot k) = O(n \cdot \lg n)$

Space: $O(1)$

VIII. Cycle Detection

Intuition: 該問題可 reduce 到 cycle detection 問題

觀察 - 無 duplicated number 的 $num = [2, 3, 1, 4, 5]$

設計 - traverse 方式為:

$$x \rightarrow f(x) \rightarrow f(f(x)) \rightarrow \dots \rightarrow f(f(f(\dots f(x)\dots)))$$

其中: $f(x) = num[x]$

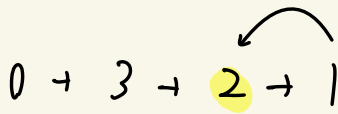
則:

$$\begin{aligned} 0 \rightarrow num[0] = 2 \rightarrow num[2] = 1 \rightarrow num[1] = 3 \\ \rightarrow num[3] = 4 \rightarrow num[4] = 5 \rightarrow 5 \end{aligned}$$

但假設存在 duplicated number 時,

$$num = [3, 2, 1, 2, 4, 5]$$

0 → 3 → 2 → 1

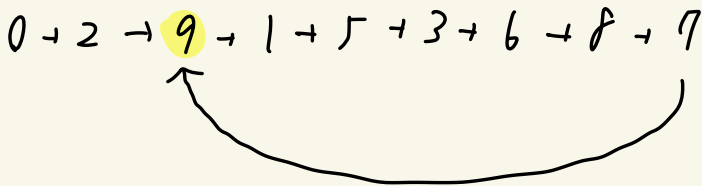


可發現，該 traverse 中存在 - cycle

Cycle 起點即為 duplicated number

Example: num = [2, 5, 9, 6, 4, 3, 8, 9, 7, 1]

0 → 2 → 9 → 1 → 5 → 3 → 6 → 8 → 7



故又要利用 Find Cycle 演算法，找到 Cycle entrance

即可解該問題