Previous (/articles/find-the-duplicate-number/) Next (/articles/pascals-triangle/)

# 179. Largest Number (/problems/largest-number/)

Dec. 12, 2017 | 30.5K views

Average Rating: 3.83 (30 votes)

Given a list of non negative integers, arrange them such that they form the largest number.

**Example 1:**

```
Input: [10,2]
Output: "210"
```

**Example 2:**

```
Input: [3,30,34,5,9]
Output: "9534330"
```

**Note:** The result may be very large, so you need to return a string instead of an integer.

## Approach #1 Sorting via Custom Comparator [Accepted]

### Intuition

To construct the largest number, we want to ensure that the most significant digits are occupied by the largest digits.

### Algorithm

First, we convert each integer to a string. Then, we sort the array of strings.

While it might be tempting to simply sort the numbers in descending order, this causes problems for sets of numbers with the same leading digit. For example, sorting the problem example in descending order would produce the number $9534303$, while the correct answer can be achieved by transposing the $3$ and the $30$. Therefore, for each pairwise comparison during the sort, we compare the numbers achieved by concatenating the pair in both orders. We can prove that this sorts into the proper order as follows:

Assume that (without loss of generality), for some pair of integers $a$ and $b$, our comparator dictates that $a$ should precede $b$ in sorted order. This means that $a \frown b > b \frown a$ (where $\frown$ represents concatenation). For the sort to produce an incorrect ordering, there must be some $c$ for which $b$ precedes $c$ and $c$ precedes $a$. This is a contradiction because $a \frown b > b \frown a$ and $b \frown c > c \frown b$ implies $a \frown c > c \frown a$. In other words, our custom comparator preserves transitivity, so the sort is correct.

Once the array is sorted, the most "signficant" number will be at the front. There is a minor edge case that comes up when the array consists of only zeroes, so if the most significant number is $0$, we can simply return $0$. Otherwise, we build a string out of the sorted array and return it.

Java | Python3 — Copy

```
class Solution {
    private class LargerNumberComparator implements Comparator<String> {
        @Override
        public int compare(String a, String b) {
            String order1 = a + b;
            String order2 = b + a;
           return order2.compareTo(order1);
        }
    }

    public String largestNumber(int[] nums) {
        // Get input integers as strings.
        String[] asStrs = new String[nums.length];
        for (int i = 0; i < nums.length; i++) {
            asStrs[i] = String.valueOf(nums[i]);
        }

        // Sort strings according to custom comparator.
        Arrays.sort(asStrs, new LargerNumberComparator());

        // If, after being sorted, the largest number is `0`, the entire number
        // is zero.
        if (asStrs[0].equals("0")) {
            return "0";
        }

        // Build largest number from sorted array.
```

### Complexity Analysis

- Time complexity : $\mathcal{O}(nlgn)$

  Although we are doing extra work in our comparator, it is only by a constant factor. Therefore, the overall runtime is dominated by the complexity of `sort`, which is $\mathcal{O}(nlgn)$ in Python and Java.

- Space complexity : $\mathcal{O}(n)$

  Here, we allocate $\mathcal{O}(n)$ additional space to store the copy of `nums`. Although we could do that work in place (if we decide that it is okay to modify `nums`), we must allocate $\mathcal{O}(n)$ space for the final return string. Therefore, the overall memory footprint is linear in the length of `nums`.

Analysis and solutions written by: @emptyset (https://leetcode.com/emptyset)

Rate this article:

Previous (/articles/find-the-duplicate-number/) Next (/articles/pascals-triangle/)

## Comments: 21

Sort By

Type comment here... (Markdown is supported)

Preview Post

plantvsdyf (plantvsdyf) ★ 5 October 15, 2018 1:29 AM

LOL. The "proof" is cute.

4 Share Reply

the0demiurge (the0demiurge) ★ 3 March 28, 2018 9:48 AM

@kounkou
It is not an O(N) solution because you used sort() and sort is O(N log(N))

3 Share Reply

SHOW 3 REPLIES

kalfon (kalfon) ★ 2 May 14, 2018 8:47 AM

I came up with a different compare function. I take the two strings representing the numbers to be compared, s1 and s2, and compare each character with wraparound. That is, I start at i = 0 up to len(s1)*len(s2) (I think lcm(len(s1), len(s2)) is actually enough) and compare s1[i % len(s1)] and s2[i % len(s2)]. The first non-equal

occurence tells me which string is 'bigger'.

**2** Share Reply

SHOW 2 REPLIES

victorxie2013 (victorxie2013) ★ 2 June 7, 2018 2:28 AM

I don't get why "a⁀b > b⁀a and b⁀c > c⁀b implies a⁀c > c⁀a ". It seems to me some proof is needed here. Can some elaborate how to prove it?

**2** Share Reply

SHOW 2 REPLIES

anmingyu11 (anmingyu11) ★ 106 February 22, 2019 8:28 AM

I cant imagine the a~b > b~a , this is so subtle, how can you figure it out ? i mean int math?or just like a idea com out of your mind ?

**2** Share Reply

sunsys (sunsys) ★ 11 October 17, 2018 3:57 AM

Why is the Custom Comparator descending order after sort, not ascending as the String's natural ascending order? It seems "return order2.compareTo(order1)" caused that descending order after sort. @emptyset

**1** Share Reply

SHOW 1 REPLY

haoyangfan (haoyangfan) ★ 202 January 27, 2019 3:57 PM

offer my javascript solution:

```
/**
 * @param {number[]} nums
 * @return {string}
```

Read More

**1** Share Reply

shodan (shodan) ★ 9 August 5, 2018 6:01 PM

The Java solution is accidentally O(n^2) because it is not using a StringBuilder, and instead repeatedly concatenating `String` s. This

```
        // Build largest number from sorted array.
        String largestNumberStr = new String();
```

Read More

**0** Share Reply

SHOW 2 REPLIES

kounkou (kounkou) ★ 19 March 17, 2018 11:21 AM

Hello all,

what about my solution ?
Can someone confirm this is a O(N) Time complexity solution ?

Read More

**0** Share Reply

jvjplus (jvjplus) ★ 0 December 29, 2017 7:38 AM

One of the most challenging problem solved as a novice by myself! great feat for me, but looking editorial made me sad as i wouldn't took advantage of inbuilt java functionalities!

**0** Share Reply

‹ 1 2 3 ›