Maximum Swap

solved by	Senan
Platform	LeetCode
difficulty	Medium
# Serial	670
tags	Vector
language	C++
solved on	@17/10/2024
link	https://leetcode.com/problems/maximum-swap/
Completion	☑

Intuition

The goal is to get the largest possible number by swapping at most two digits of the given number. To achieve this, we should swap the smallest digit (starting from the left) with the largest digit (starting from the right) that occurs later in the number.

Approach

1. **Convert the number to a string** to easily manipulate individual digits.
2. **Iterate backwards** through the digits while maintaining the largest digit (`currMax`) and its index (`maxInd`).
3. For each digit, check if it is smaller than the largest digit found after it (`currMax`). If so, mark this position for a potential swap.
4. **Perform the swap**: After identifying two digits to swap (if any), perform the swap to maximize the number.
5. **Return the resulting number** after converting the string back to an integer.

Complexity

Time Complexity:

- Converting the number to a string takes **O(d)**, where **d** is the number of digits.
- The single loop through the string takes **O(d)**.
- Swapping two characters and converting back to an integer are **O(d)** operations.
- Overall, the time complexity is **O(d)**.

Space Complexity:

- We use extra space for the string representation of the number, which requires **O(d)** space.
- Thus, the space complexity is **O(d)**.

Code

```
class Solution {
public:
    int maximumSwap(int num) {
        string myNum = to_string(num);

        char currMax = '0';
        int maxInd = -1;
        int swap1 = -1, swap2 = -1;

        for (int i = myNum.size() - 1; i >= 0; i--) {
            if (myNum[i] > currMax) {
                currMax = myNum[i];
                maxInd = i;
            }
            if (myNum[i] < currMax) {
                swap1 = i;
                swap2 = maxInd;
            }
        }
        if (swap1 >= 0 && swap2 >= 0)
            swap(myNum[swap1], myNum[swap2]);

        return stoi(myNum);
    }
};
```