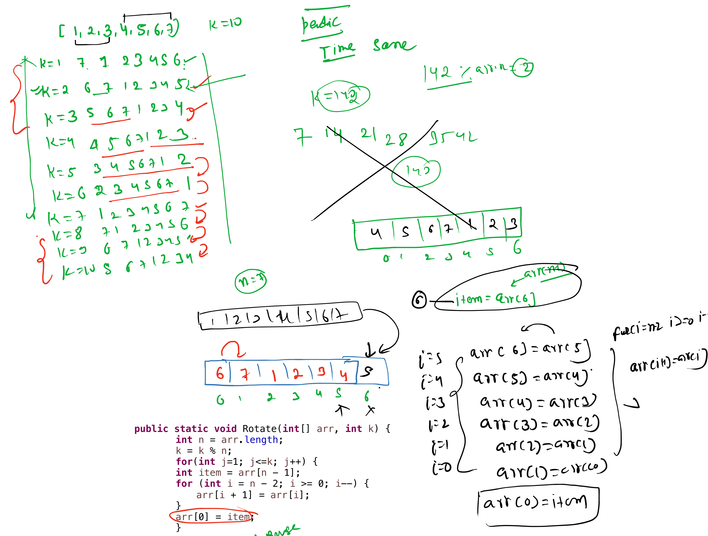
Lec-9

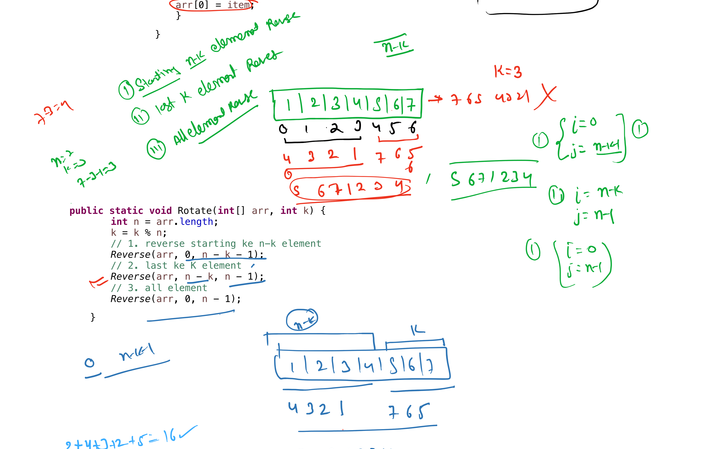
13/10/24 12:48 PM

Given an integer array `nums`, rotate the array to the right by `k` steps, where `k` is non-negative.

Example 1:

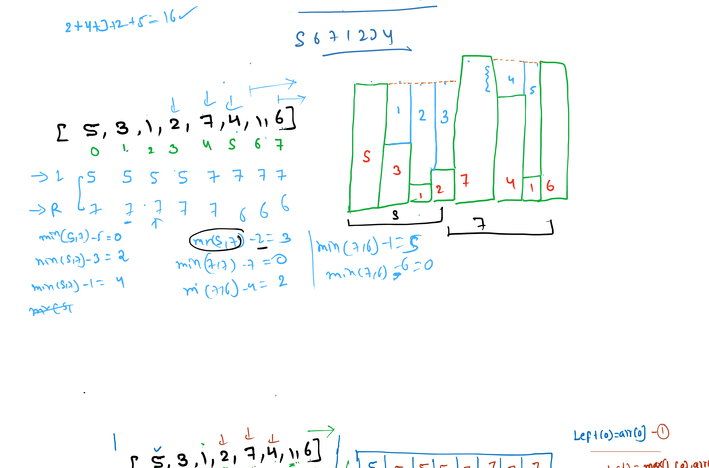
```
Input: nums = [1,2,3,4,5,6,7], k = 3
Output: [5,6,7,1,2,3,4]
Explanation:
rotate 1 steps to the right: [7,1,2,3,4,5,6]
rotate 2 steps to the right: [6,7,1,2,3,4,5]
rotate 3 steps to the right: [5,6,7,1,2,3,4]
```





```
public static void Rotate(int[] arr, int k) {
        int n = arr.length;
        k = k % n;
        for(int j=1; j<=k; j++) {
        int item = arr[n - 1];
        for (int i = n - 2; i >= 0; i--) {
                arr[i + 1] = arr[i];
        }
        arr[0] = item;
        }

    }
```



```
public static void Rotate(int[] arr, int k) {
        int n = arr.length;
        k = k % n;
        // 1. reverse starting ke n-k element
        Reverse(arr, 0, n - k - 1);
        // 2. last ke K element
        Reverse(arr, n - k, n - 1);
        // 3. all element
        Reverse(arr, 0, n - 1);

    }
```


Given an integer array `nums`, return *an array* `answer` *such that* `answer[i]` *is equal to the product of all the elements of* `nums` *except* `nums[i]`.

The product of any prefix or suffix of `nums` is **guaranteed** to fit in a **32-bit** integer.

You must write an algorithm that runs in `O(n)` time and without using the division operation.