Rotate And Delete

solved by	Senan
Platform	GeeksForGeeks
difficulty	Medium
tags	Logic Vector
language	C++
solved on	@October 2, 2024
link	https://www.geeksforgeeks.org/problems/rotate-and-delete-1587115621/1
Completion	☑

Intuition

The problem involves repeatedly rotating an array and deleting elements based on a changing index `k`. By rotating the array and progressively deleting elements, we are left with one final element. The intuition is to rotate the array in such a way that we can mimic the behavior of deleting the last few elements and gradually reduce the array size.

Approach

1. Start with a counter `k` initialized to 1.
2. Rotate the array such that the last element moves to the front (right rotation).
3. Calculate the position of the element to be deleted: `arr.size() - k`. This dynamically adjusts with each step.
4. If the calculated deletion index is negative (when `k` exceeds the size of the array), reset it to 0 to avoid out-of-bounds access.
5. Delete the element at the calculated position.
6. Increment `k` and repeat the process until only one element is left.
7. Return the remaining element.

Complexity

Time Complexity:

$$O(n^2)$$

- Rotating the array takes O(N) time in each iteration, and deleting an element takes O(N) time as well.
- The loop runs `N` times, making the total time complexity O(N^2).

Space Complexity:

$$O(1)$$

- No additional space is used apart from the input array itself, so the space complexity is O(1).

Code

```
class Solution {
  public:
    int rotateDelete(vector<int> &arr) {
        int k = 1;
        while(arr.size()>1)
        {
            rotate(arr.begin(), arr.begin() + arr.size() - 1, arr.end());

            int del = arr.size() - k;
            if(del<0) del = 0;

            arr.erase(arr.begin()+del);
            k++;
        }
        return arr[0];
    }
};
```