Problems on Arrays, Strings & Matrices

1. WAF to rotate an array of size `n` by `k in a clockwise direction`.

   Solution:
   ```
   reverse (ar, 0, n-1);
   reverse (ar, 0, k-1);
   reverse (ar, k, n-1);
   ```

2. WAF to answer multiple queries of the form `Sum[i,j]` for a given array.
   Extension: Queries are of the form `Product[i,j]`.

   Solution:
   a. Use PrefixSum: `PrefixSum[0] = ar[0]`
      `PrefixSum[i] = PrefixSum[i-1] + ar[i]`
   b. Answer Query: `Sum[i,j] = PrefixSum[j] - PrefixSum[i-1], or`
      `Sum[i,j] = PrefixSum[j] - PrefixSum[i] + ar[i]`

3. Given an array containing all `0`'s and Queries of the form (`i j x`) - add `x` to the sub-array `i` to `j`, WAF to find the final state of the array.

   Solution:
   a. Prefix Sum: To add `x` from index `i` to `j`, add `x` to `i` and subtract `x` from `j+1`. After processing all the queries in this manner, do a prefix sum.
   b. Use Segment Tree.
   c. Use Binary Indexed Tree [BIT].

4. Given an array containing heights of adjacent buildings, find the amount of water that will be trapped in the space between buildings.

   Solution: For each building `i`, find the height of the highest building on the left [L_{max}] and height of the highest building on the right [R_{max}], then the amount of water accommodated over current building (`height` $= H_i$) can be computed as:
   $Water_i = \max(0, \min(L_{max}, R_{max}) - H_i)$

5. Given an array, find the maximum `j-i` such that `arr[j] > arr[i]`.
   Solution: Take two auxiliary arrays L_{min} and R_{max} [Geeks4GeeksReference].

6. Given a matrix where every row is sorted in increasing order. Write a function that finds and returns a common element in all rows. If there is no common element, then the function returns `-1`.

   Solution: Start from right of all rows, if all are same, then bingo! Else, we find the minimum element and reduce indexes for all other rows. We can do the same thing starting from left and find the maximum element and increase indexes for all other rows.

7. Given a boolean two dimensional [2D] array, where each row is sorted, find the row with the maximum number of `1`'s.
   Solution Hint: Start from right-top or right-bottom of the matrix.

8. WAF to find the smallest missing integer in a sorted array, which contains only positive elements (`ar[i] >= 1`) and no duplicates.
   Solution Hint: Binary search based on `ar[i] == i+1`

9. Check if a given array contains duplicate elements within k distance from each other.
   Solution: Use `unorederd set` and keep removing (`i-k`)th element.

10. Given a set of time intervals in any order, merge all overlapping intervals into one and output the result which should have only mutually exclusive intervals.
    Solution: Sort by starting time, linear traversal to merge.

11. Given an unsorted integer array, find the first missing positive integer. Your algorithm should run in O(n) time and use constant space.
    Examples:
    ```
    1 2 0        ->    3
    3 4 -1 1     ->    2
    8 -7 -6      ->    1
    ```
    Solution: Use the same array for hashing elements of range [`1`, `n`] and iterate to find the missing first positive integers. Refer to class notes for code.

12. **Prime Numbers** - **Sieve of Eratosthenes** , **Sieve of Atkin** , **Geeks4GeeksReference**

13. **Game Theory** - **TopCoderReference** , **Additional Information on Game Theory**

14. **Meet-in-the-middle Algorithm** - **Reference Link**