# Questions

## Dynamic Programming

### Regular expression match

Given an input “**xaabyc**” and regular expression **“xa\*b.c”**

We need to check if the pattern matches the input. We need to use dynamic programming using tabulation to solve this approach.

## Array

### Multiplication without that position value

First loop over the array and calculate the total multiplication. Then again iterate over the array and divide the total multiplication with the positional element.

Time complexity will be **O (2n)** because two times iteration.

**Approach 2:**

If we asked to not use division?

We use one loop to find all the pre multiplication except that position.

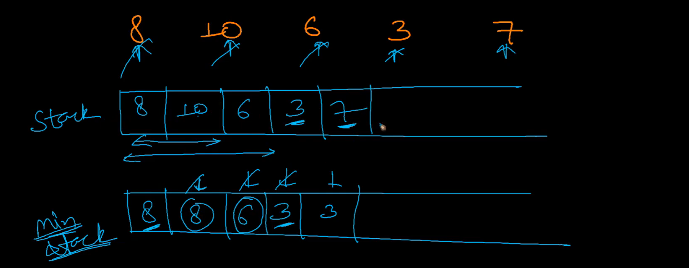
Other loop from last position to calculate after sum.

One more loop to multiply before and after sum.

## Special stack to get minimum element in O(1)

### Solution

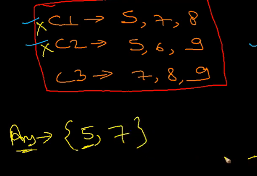
We need to maintain an auxiliary stack where we will keep the minimum element till that position.



## Lazy Bartender problem

### Problem

Given 3 customers and their favorite drink (3 arrays). Find minimum drinks the bartender server to satisfy the entire customer.



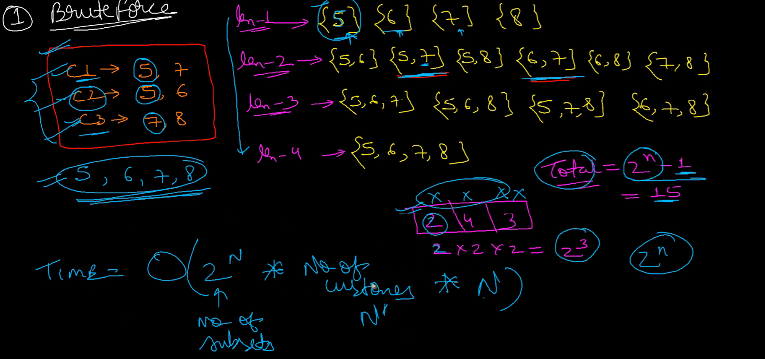
Here we need to find minimum common elements in arrays

### Brute force with subsets

Combine all the arrays to form a single array (unique) and find all possible subset. If an array has **n** elements number of subsets will be **2n.** As each element have 2 choices to be picked or not to be picked in subset.

Sort all the subset in ascending order of element count.

Pick each subset and compare if it’s present in all the other subsets and if present then that’s the minimum subset we are looking for.



### Greedy using Map (kind of graph can say)

Create a map key as drink and value is list of customer likes it. Sort the value based on size of the list (customer). We can use a max heap here.

Take the top element and add the drink number to result.

Iterate over the customer array (value of that drink) and remove all the keys from the map (as those customers are now served.

Repeat the above two steps till the map is empty.

