Contest Discussion

1. string should have → atleast 1 vowel & 1 consonant

   S = "BCW" X If length = 1 → 0 is ans

   'A' X

   'ABCE' ✓

* Print all valid strings of length n?

Use Recursion

1. We will go to each index and by brute force will make all strings and in last we will check if it is valid or not (1 vowel + 1 consonant). If yes → Print string.

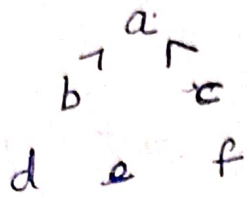
Optimal :-

All consonant → 21^N
All vowels → 5^N
} Not all consonants & not all vowels needed.

$\therefore$ ans = $26^n - (21^N + 5^N)$

Whenever constraints is greater than 10^9 → use maths

2.



$a < b$ && $a < c$

1) Sort the array given.
2) Now place the smallest element at starting and then keep the rest at b, c, d, e and f pos^n.
3) Now check the cond^n.

3.

Size = 3
Operation, M = 5

[1, 2, 3, 2, 1, 4]

[~~2~~ ~~3~~ ~~4~~ 2 1 4]

[2 2 3 ~~3~~ 2 ~~5~~]

[3 4 5 3 2 5]

[3 4 5 4 3 6]

[4 5 6 4 3 6]

Max − Min value in array = ?

If not 4 ~~if (4)~~ we can make ← 4 → If greater than it (4) we can make in right side

— — 1 2 3 (4) — —

$\therefore$ high = mid − 1 // go left

low = mid + 1 // go to right

Check if can bring (3)

[1 2 3 2 1 4] (+2 +2 +2) ⇒ [3 4 5 2 1 4]

[3 4 5 2 1 4] ((4)(+1)(+1)(+1)) ⇒ [3 4 5 3 2 5]

(4>3) (5>3)

A [3 4 5 3 2 5] ⇒ [3 4 5 3 3 6] (6>3)
(+1 on the 2, +1 on the 5)

Pseudocode :-

T.C : $O(n^2)$

```
check (x) {
 int step = 0
 for (i=0; i<n; i++) {
   if (a[i] < x) {
    int add = x - a[i], step += add;
    for (j=i; j<i+s; j++) a[j] += add;
   }
 }
 if (step <= M) return 1;
 return 0;
}
```

Optimal Approach : Using Prefix sum

Instead of updating the array a directly at every step, it updates a range indirectly by making increments at start and decrements ~~aft~~ at the end.

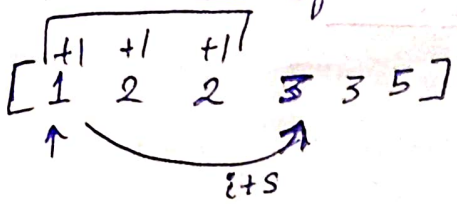
Code :-

```
int n;
vector<int>s;
int check (int x) {
  vector <int> partial (n,0);
  for (int i=0; i<n; i++){
    if (i>0) partial[i] += partial[i-1];
    a[i] += partial[i];
    if (a[i] < x){
      int add = x - a[i];
      partial[i] += add;
      if (i+m < n) partial[i+m] -= add;
      a[i] += add;
    }
  }
}
```

6. [1 2 3 3 2 5]

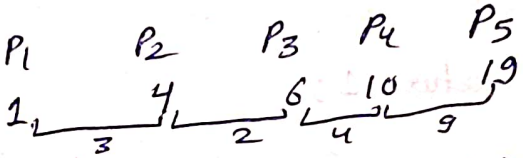
Subarray → Two pointers, sliding window

Subset ⇒ If i will sort → then would i will get ans



7. [1 - 4]
[6 - 12]
[19 - 20]

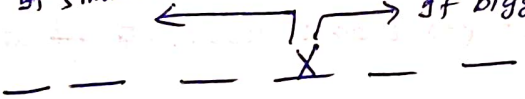
5 people we have



Z = 2 (~~min~~ closest pair value)

We have to maximize it

Binary search:



Check (5)

P1	P2	P3	P4	P5
1	6	11	19	X

] If min dist (Z) = 5 → Not psbl.