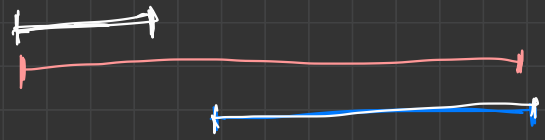
```
for( i=0; i<n ) {
    for( j=i; j<n) {
        sum=0;
        for ( k=i ; k<=j){
            sum += a[k];
        }
        if( sum%n ==0) cnt++;
    }
}
```

sum = csum[j] - csum[i-1]



1 5 4 6 8 9 10

csum

1	6	10	16	24	33	43
0	1	2	3	4	5	6

→ at index i, stores sum of all elements from 0 to i

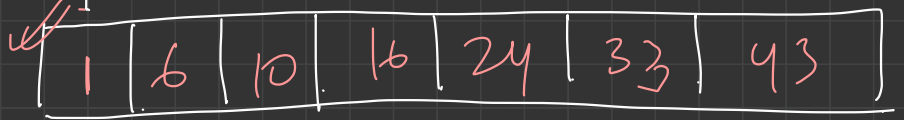
Prefix Sum

can you find sum b/w any two indices efficiently

i → 2 → 5

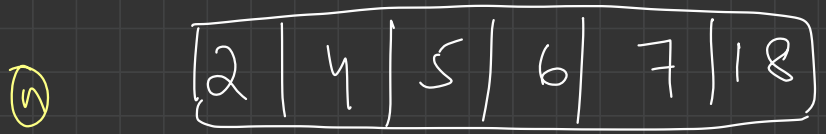
(csum[5] − csum[2−1]) / csum[1] } sum of elements b/w 2 and 5

1 5 4 6 8 9 10

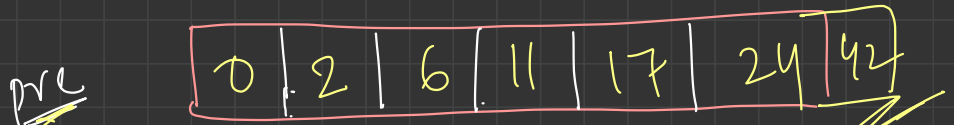


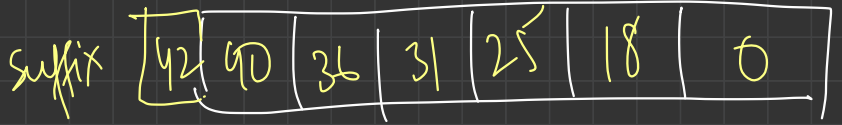
a

$csum[i] = csum[i-1] + a[i]$



n = 6





csum : sum from $0 \rightarrow i$

prefix sum : sum from $0 \rightarrow i-1$

suffix sum : sum from $i+1 \rightarrow n-1$

2	4	5	6	7	12

n = 6

csum

2	6	11	17	24	36

remainder of csum

2	0	5	5	0	0

csum[3] − csum[2]

div 5

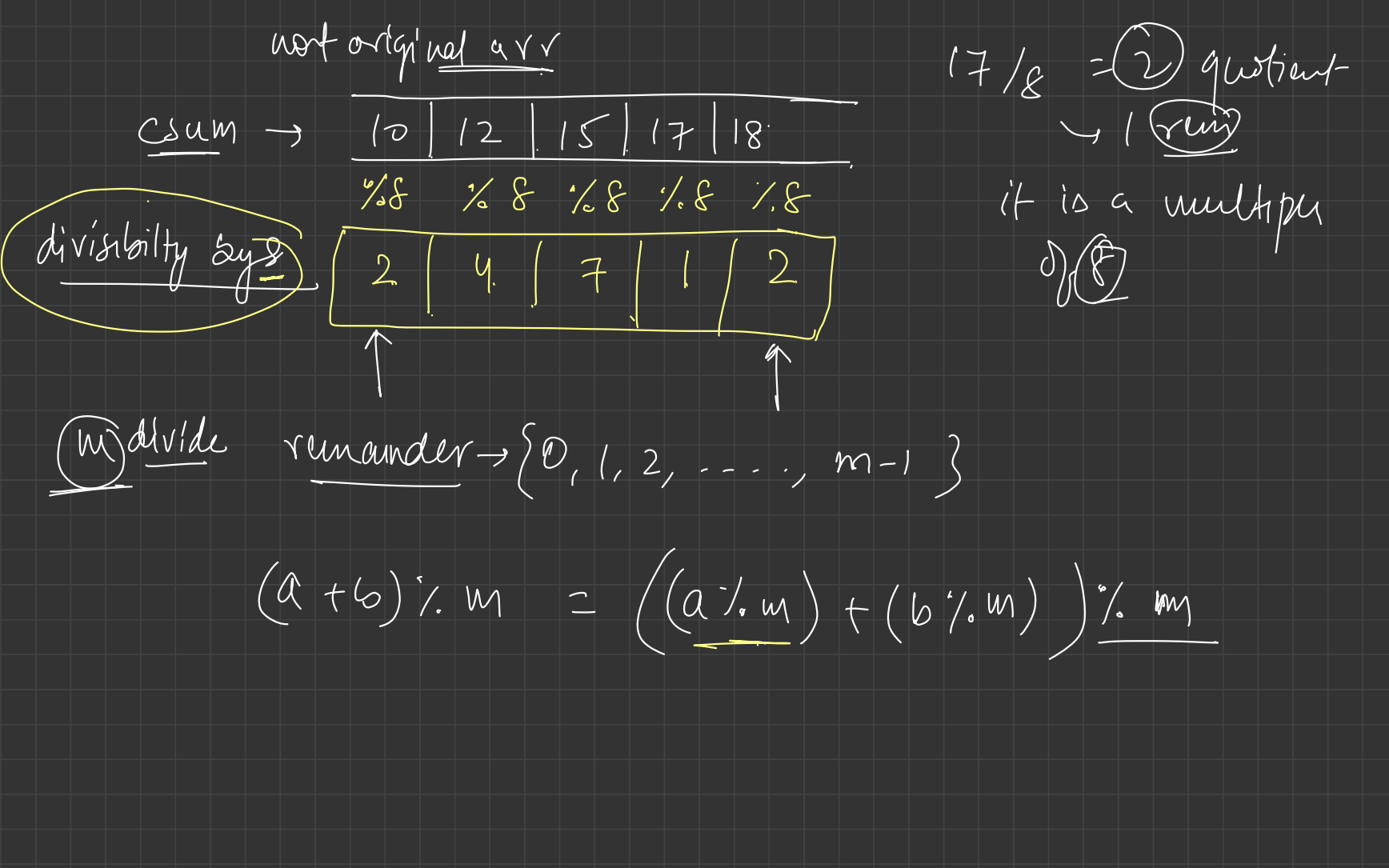
these sum upto a good arrays

rem	index
0	1, 4, 5
1	—
2	0
3	—
4	—
5	2, 3

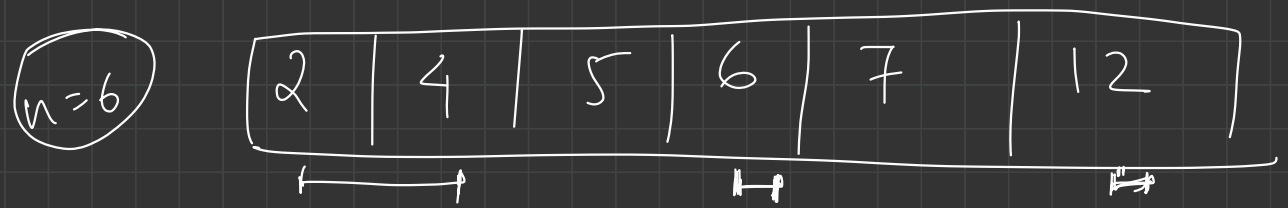
2 → 4 ⇒ 5+6+7 = 18

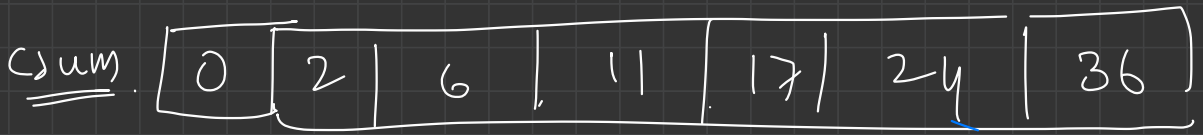
2 → 5 ⇒ 5+6+7+12 = 30

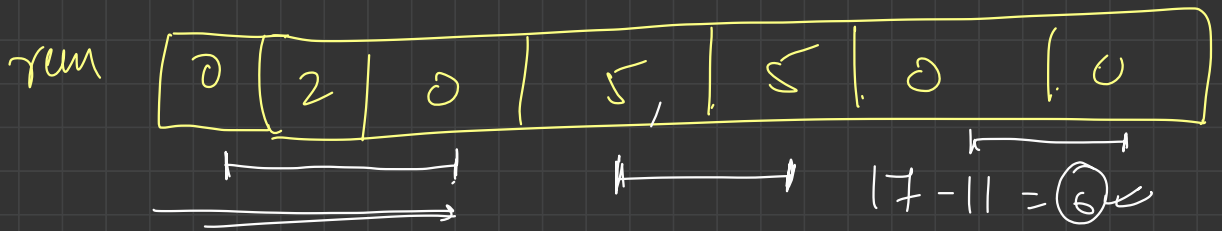
5 → 5 ⇒ 12



$$(a+b)\%\,m = \big((a\%\,m) + (b\%\,m)\big)\%\,m$$







5+6+7+12
= 30

36
-6
30

6 % 6 = 0

24 % 6 = 0

5+6+7 = 18

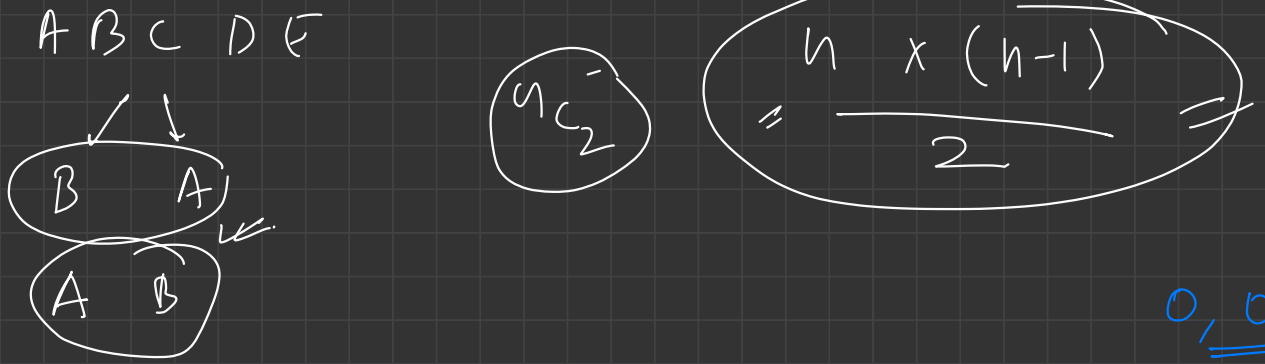
24 - 6 = 18 % 6 = 0

→ we simply need to count the frequency of remainders in csum[]

→ then we pick any two of them and it would give a required subarray

→ we know frequency of each remainder, and want to count no of ways we can pick 2 of those

n items and i need to pick two of them



we can pick any two same values in rem[] and they will represent a required array.

given a freq of a remainder, we need to know no. of unique ways to choose any two of them.

$^{n}C_2 \rightarrow \frac{n \times (n-1)}{2}$

4 items A B C D

n n-1

1st 4 2nd 3

$\frac{4 \times 3}{2} = 6$ (2)

A B ✓

(-)

B A ✓

A B ✓ B A ✗

A C ✓ B C ✓

A D ✓ B D ✓

✗ A ✗ D A

✗ B ✗ D B

C D ✓ ✗ C ✗ D C

generate all substrings and check

↓

TLE

p → anagrams

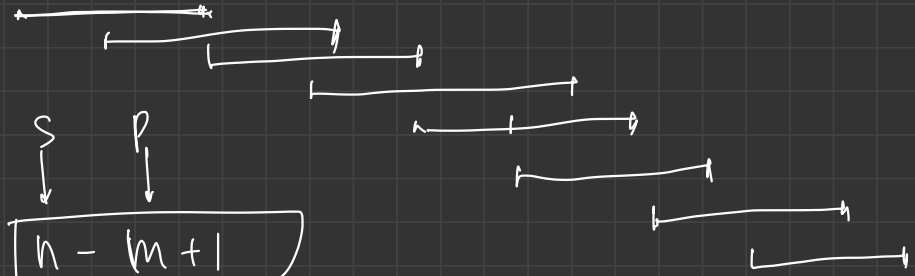
anagram : substring (s) → len (sub (s)) = len(p).

(len p = m)

c b a e b a b a c d

× × ×

abc



len

1 → n

2 → n-1 : n-2+1

3 → n-2 : n-3+1

m len

s p

n - m + 1

= n - (m - 1)

ans → 0, 6

p = abc

	freq P	freq subs S
a	1	1
b	1	0
c	1	1
d	0	1
e	0	0

S = c b a e b a b a cd

a = 1 2 3 4

b = 9 5 1 2

$f(0,3) = (a[0] \cdot b[0]) + (a[1] \cdot b[1]) + (a[2] \cdot b[2]) + (a[3] \cdot b[3])$

$= (1 \cdot 9) + (2 \cdot 5) + (3 \cdot 1) + (4 \cdot 2)$

$= 9 + 10 + 3 + 8 = 30$

can i reorder a[] and b[] so that f(0,3) is minimized

$$\begin{cases} a \to 1 & 2 & 3 & 4 & (\text{inc}) \\ b \to 9 & 5 & 2 & 1 & (\text{dec}) \end{cases}$$

$(3 \cdot 1) + (4 \cdot 2)$

$3 + 8 = 11$

$(3 \cdot 2) + (4 \cdot 1)$

$6 + 4 = 10$

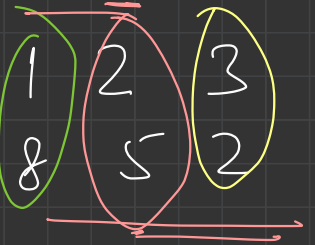
$f(0,3) = 29$

$f(0,0)$ $f(0,1)$ $f(0,2)$ $f(0,3)$

$f(1,1)$ $f(1,2)$ $f(1,3)$

$f(2,2)$ $f(2,3)$

$f(3,3)$



→ i 0 1 2

3 times 3 times

3 4 times 3

4

$f(0,0) \rightarrow 8$

$f(0,1) \rightarrow 8+10$

$f(0,2) \rightarrow 8+10+6$

$f(1,1) \rightarrow 10$

$f(1,2) \rightarrow 10+6$

$f(2,2) \rightarrow 6$

$(i+1)(n-i)$

i=0 : 1 × 5

i=1 : 2 × 4

i=2 : 3 × 3

i=3 : 4 × 2

i=4 : 5 × 1

5

$(i+1)(n-i)$

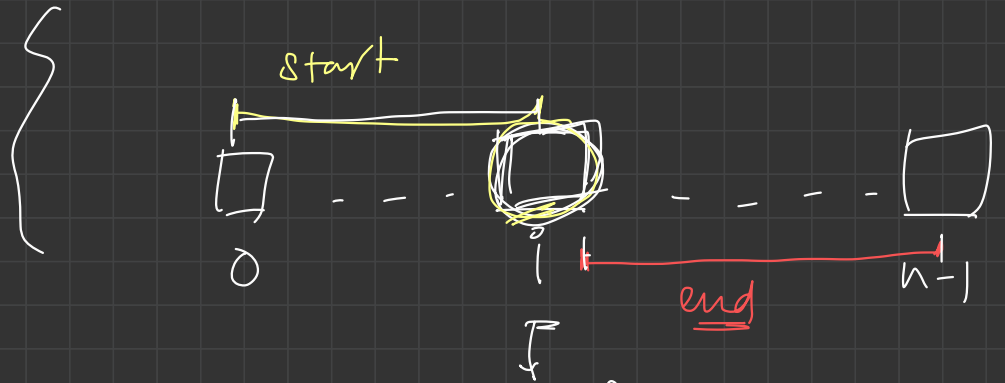
a → 1 8 7 2 4

b → 9 7 2 9 3

i: 0 1 2 3 4

a → 1 8 7 2 4

b → 9 7 2 9 3

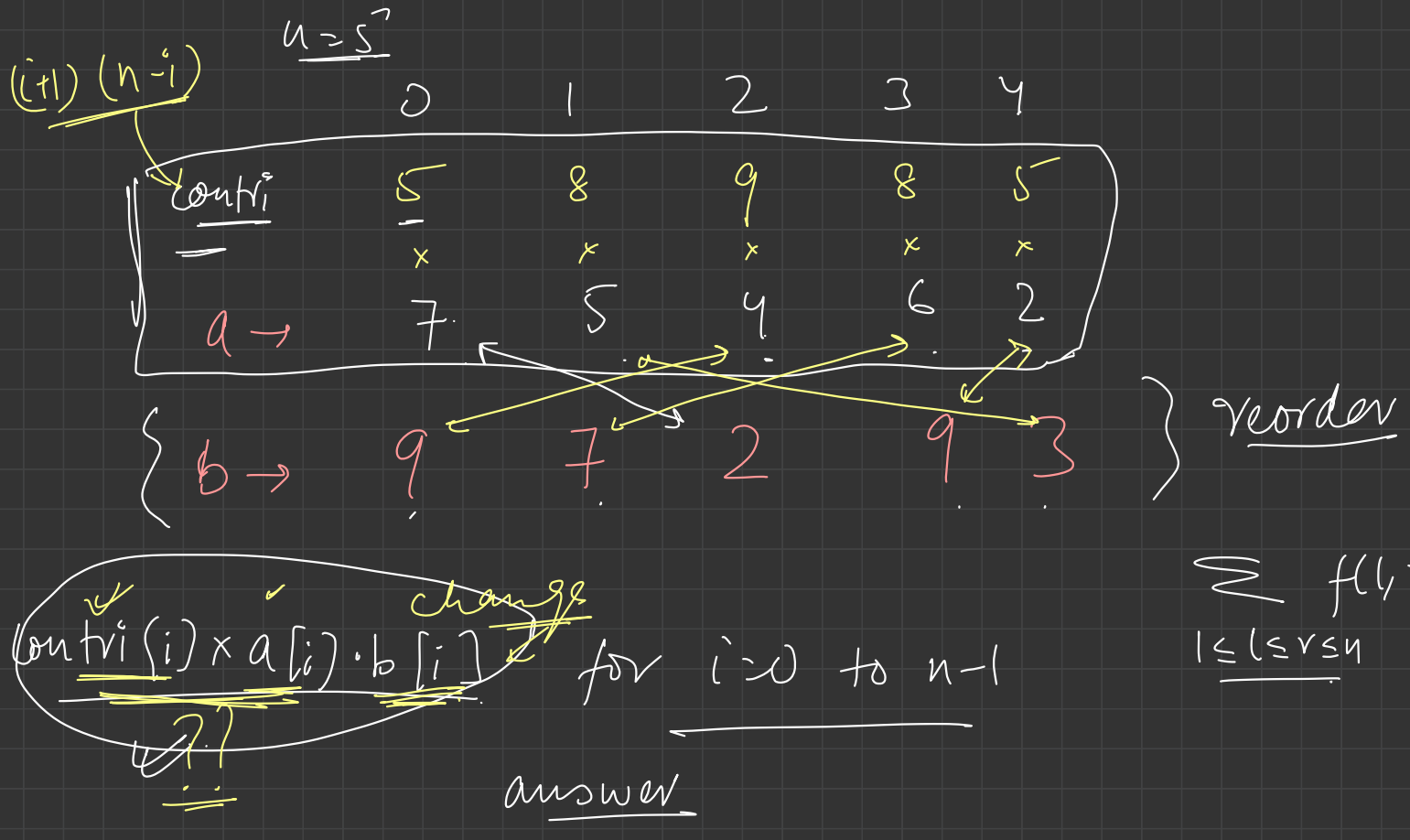


part of how many subarray?

start

$(i+1) \times (n-i)$

every index i will be present $(i+1) \times (n-i)$ times



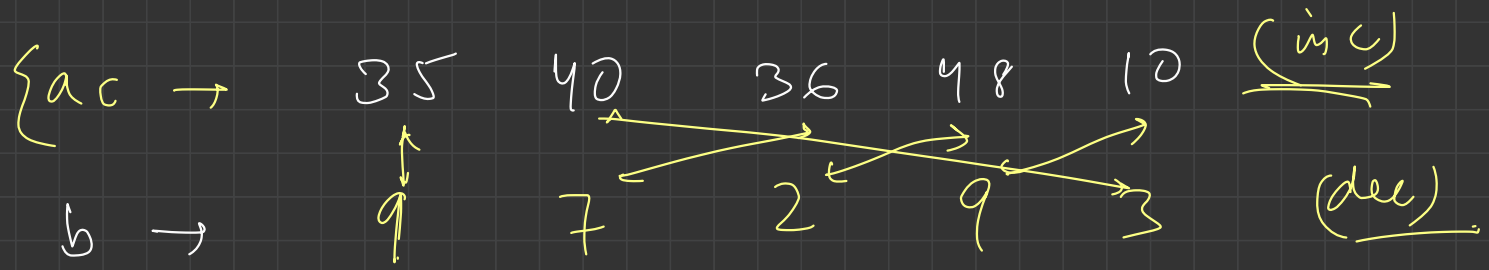
contri[i] = a[i] · b[i] (change)

for i = 0 to n-1

answer

$\geq f(l, r)$

$1 \leq l \leq r \leq n$


ac →
35
40
36
48
10
(inc)
b →
9
7
2
9
3
(dec)