Dynamic Programming Problem Solving

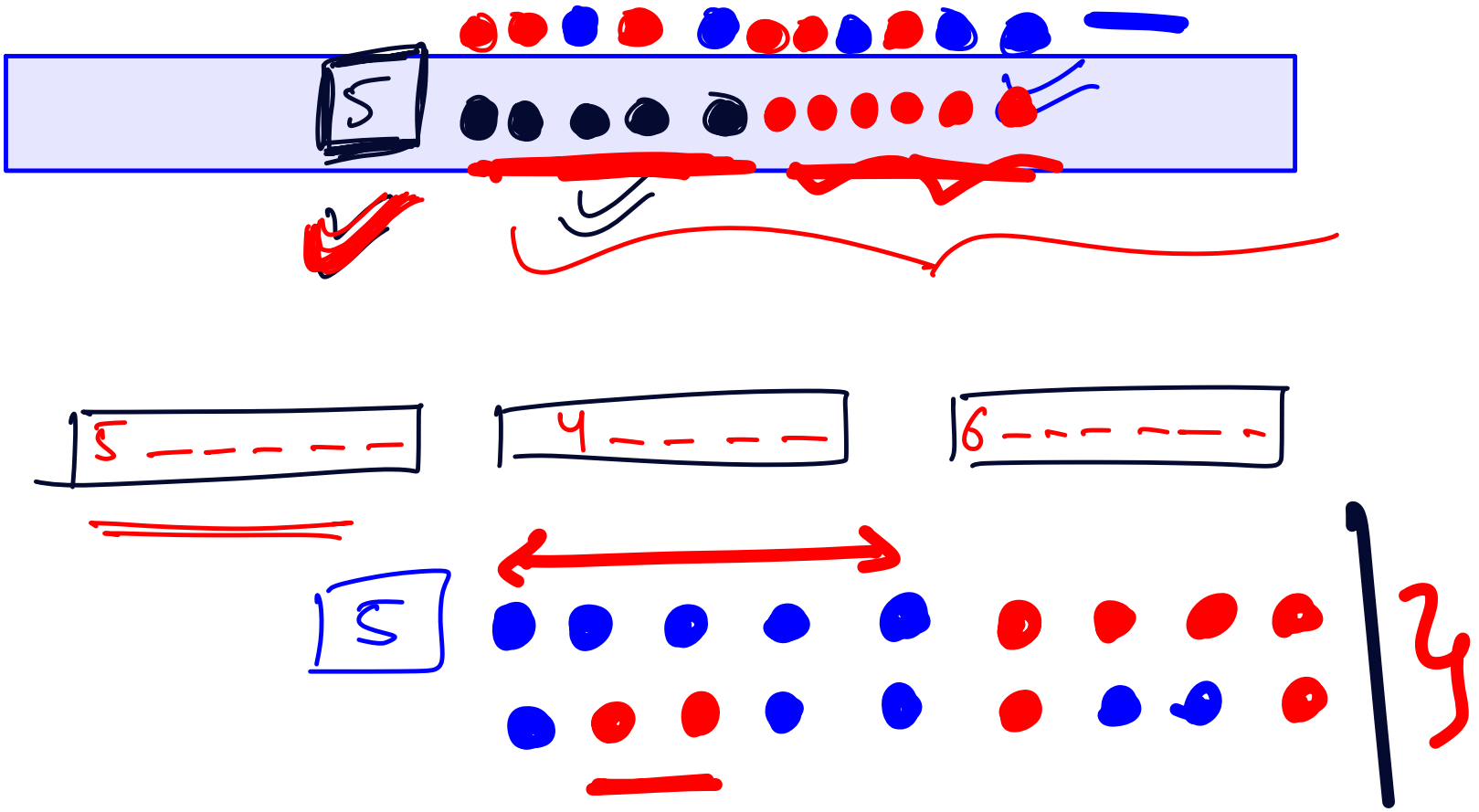
- Priyansh Agarwal

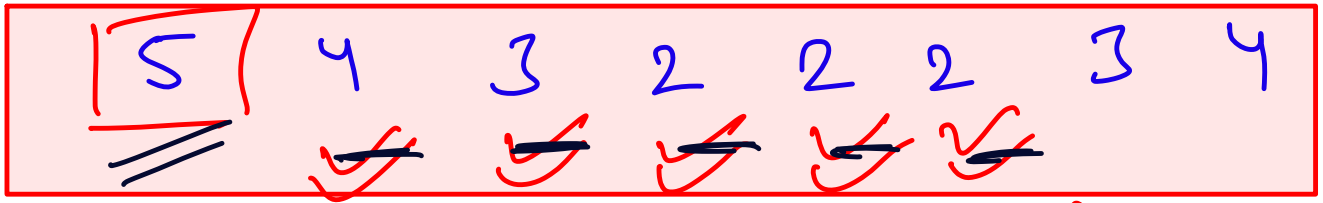
Problem 1: Block Sequence [CP-31 1500 P3]

Observations

① We want to remove minimum no. of elements from the array to make it beautiful

Removing min elements <==> Retaining max # elements


5
5
4
6
5
4



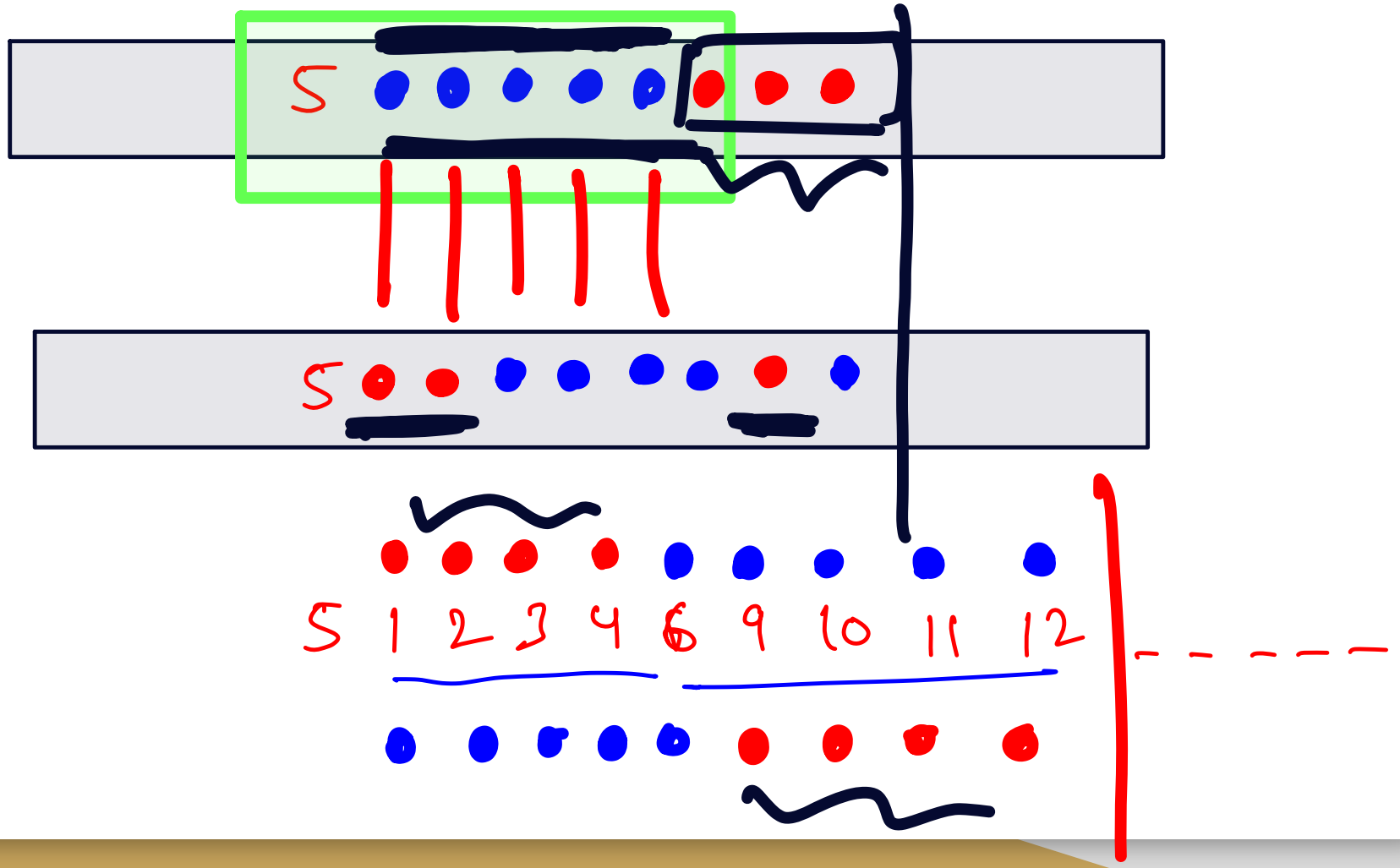
Priyansh

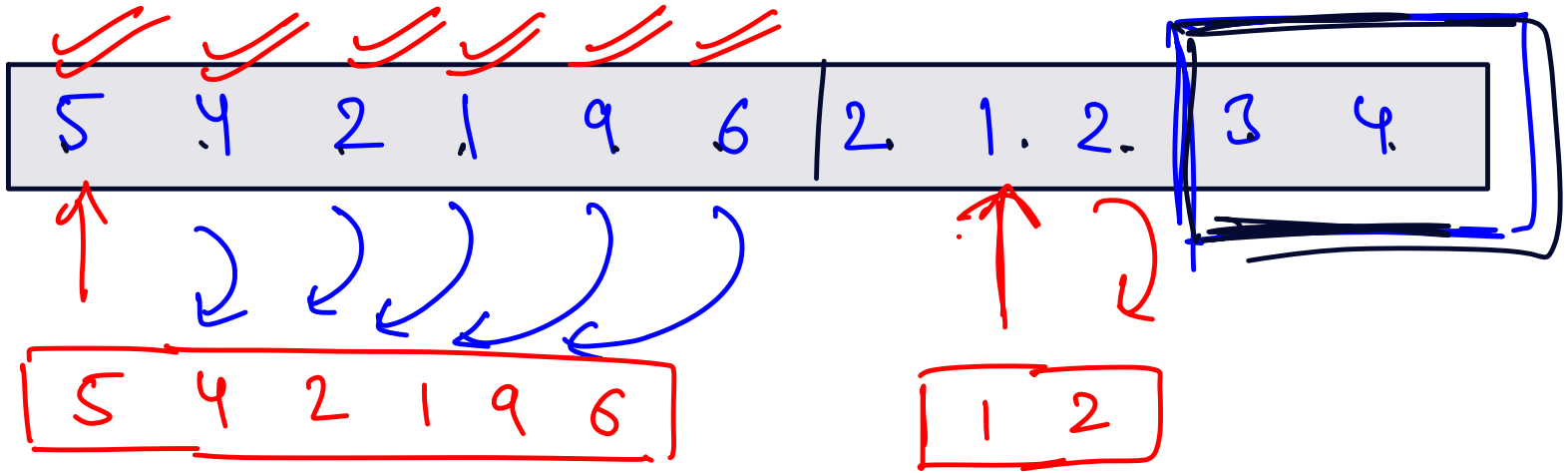
5 4 3 2 2 2 3 4

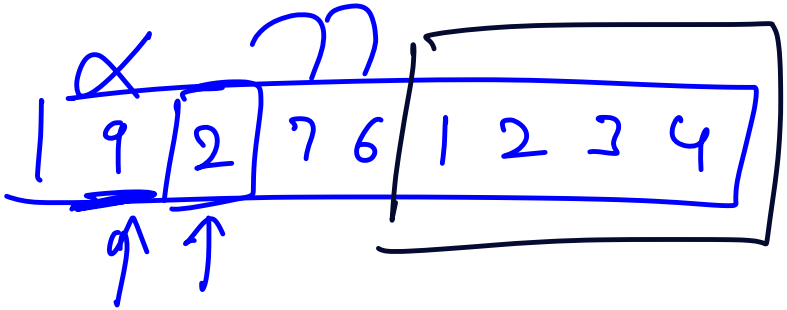
5 4 3 2 2 1 3 4

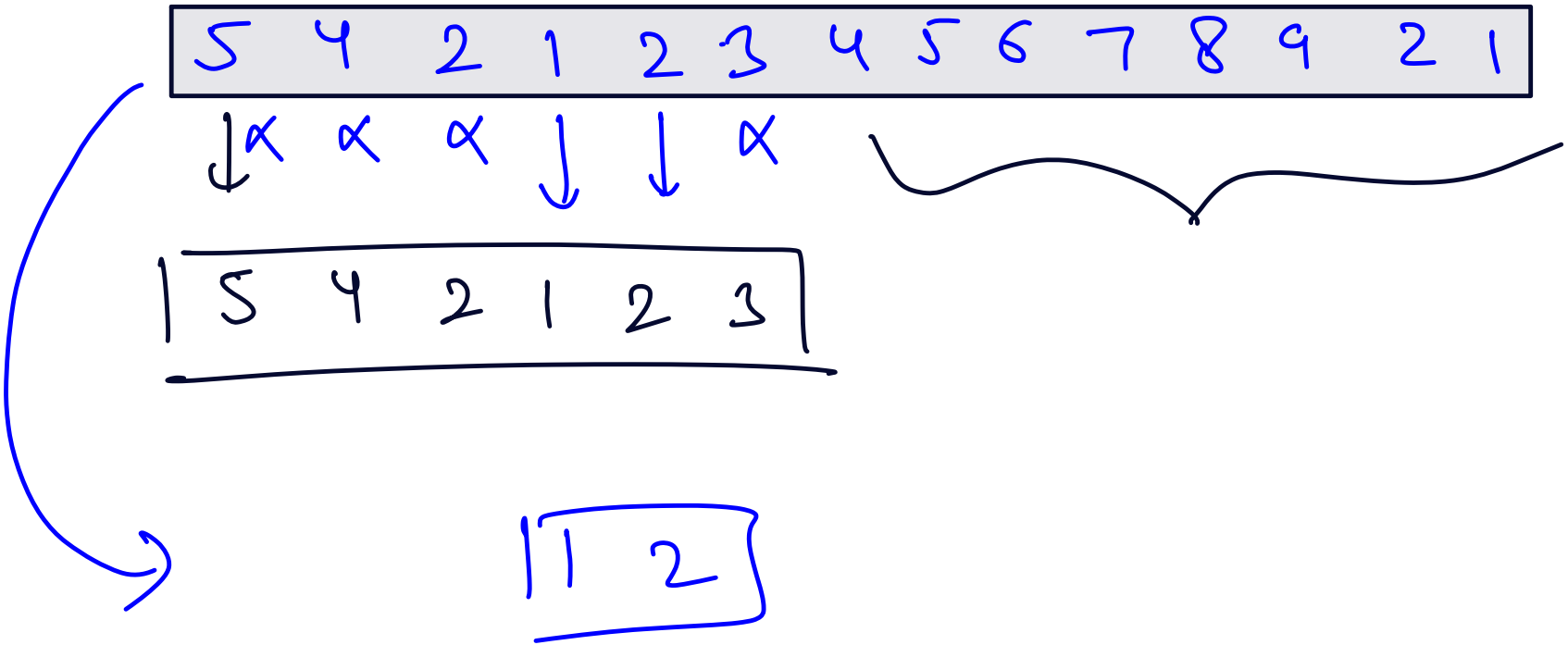
(2) if a[i] = x and we have decided to pick up a[i] as the first element in a block then we need x more elements to be picked ⟶ we should pick up

a[i+1], a[i+2] ..... a[i+x-1] as our next x elements.

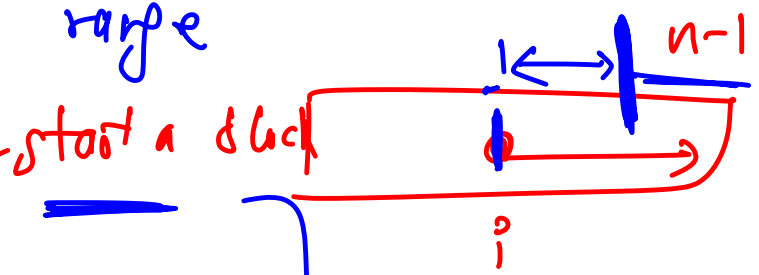

5
5
5 1 2 3 4 6 9 10 11 12


5 4 2 1 9 6 2 1 2 3 4
5 4 2 1 9 6
1 2


1 9 2 7 6 1 2 3 4


5 4 2 1 2 3 4 5 6 7 8 9 2 1
x x x x
5 4 2 1 2 3
1 2

State : $dp[i]$ = maximum no. of elements we can retain from $[i$ to $n-1]$ in the array such that all new blocks will be formed from this range



transition : $dp[i] \Rightarrow$ start a block, skip

start a block → $1 + a[i] +$ $dp[i+1+a[i]]$

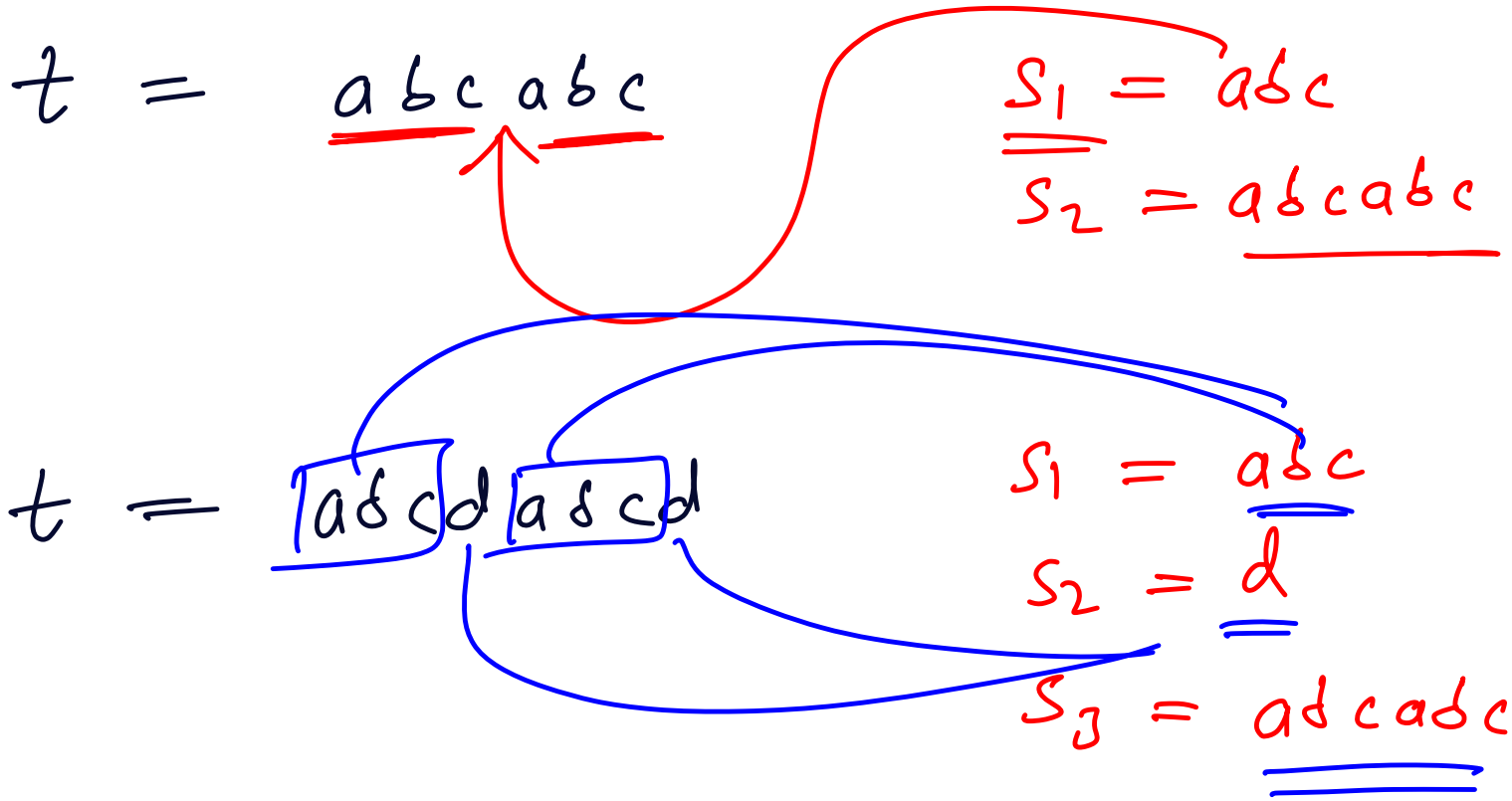
skip → $dp[i+1]$

Base Case :

$dp[n-1] = 0$, $dp[n] > 0$

final subproblem → $n - dp[0]$

Problem 2: Color with Ocurrrences


t = abcabc
S1 = abc
S2 = abcabc
t = abcdabcd
S1 = abc
S2 = d
S3 = abcabc

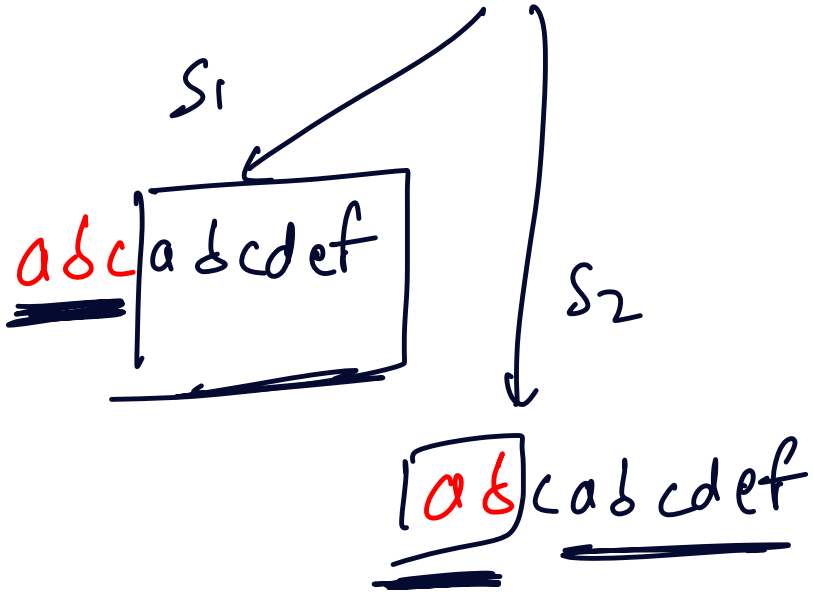
t = abcabcdef

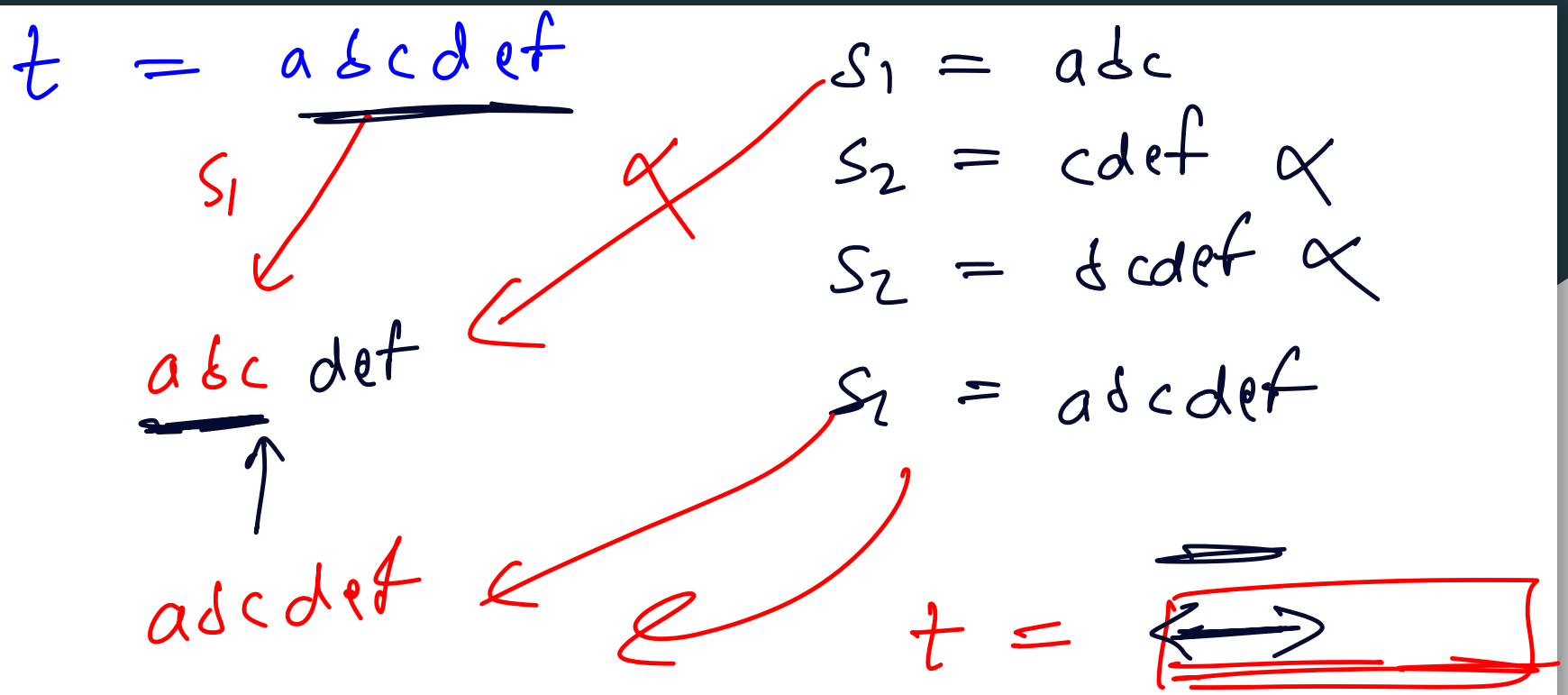
s_1 = abc

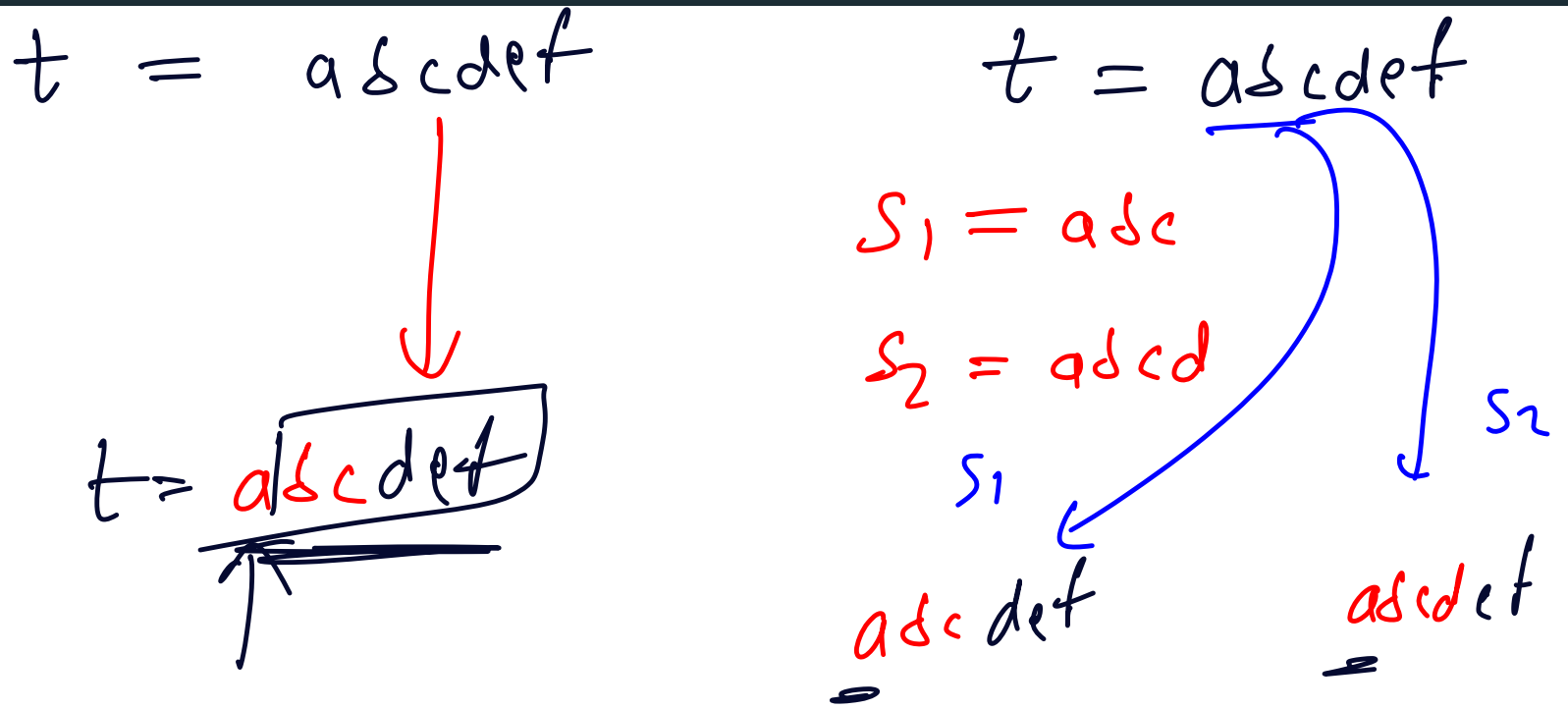
s_2 = ab

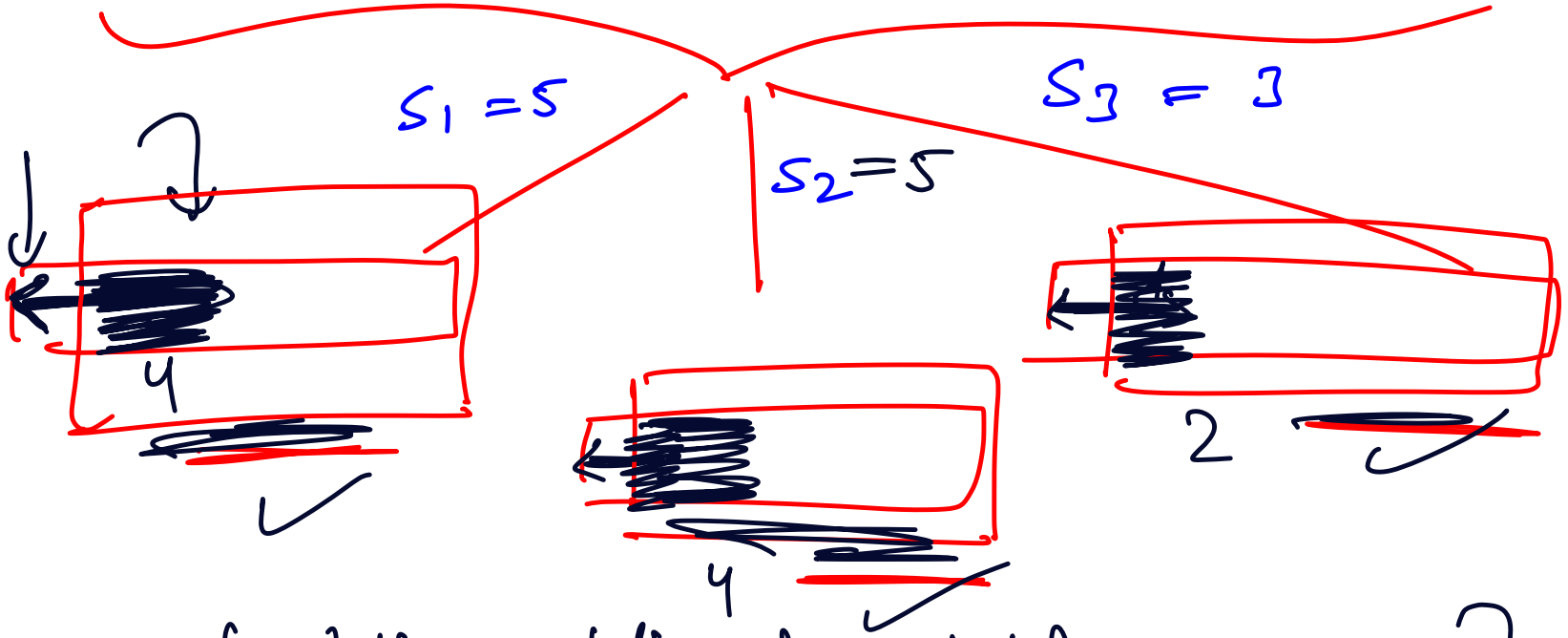
s_3 = ad

s_4 = de




t = abcdef
s1
abc def
abcdef
s1 = abc
s2 = cdef ∝
s2 = bcdef ∝
s1 = abcdef
t =


t = abcdef
t = abcdef
s1 = abc
s2 = abcd
s2
s1
abcdef
abcdef

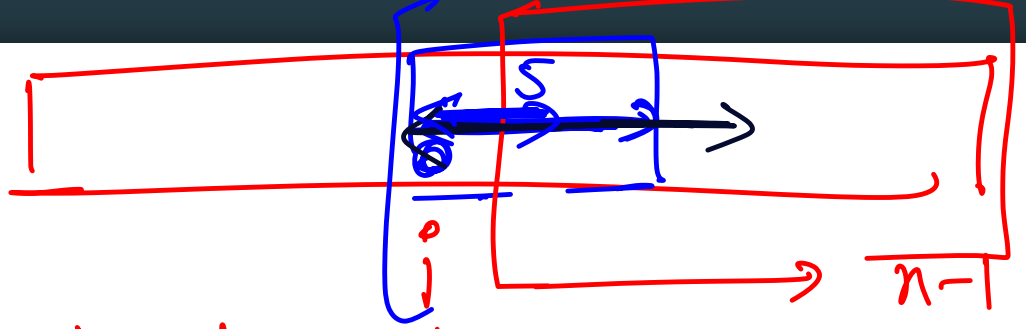


→ no of letters left in total

→ no of letters in given colored already

State :

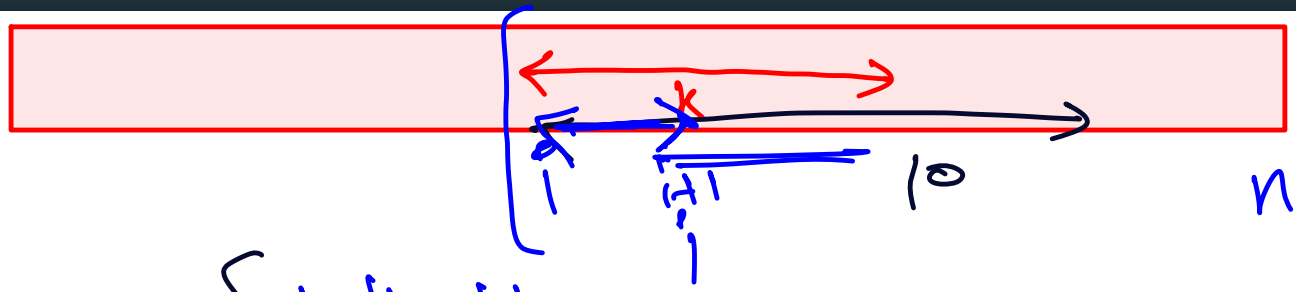
dp[i][k] = min no of operations to color all characters from i to n-1 in Red such that k characters from (i) to (i+k-1)th index are already colored.



$dp[i][k]$ = don't put anything, $dp[i+1][k-1]$

put

$1 + \min_{\forall j} \{ dp[i+1][\max(k-1, s_j - 1)] \}$



$dp[i][k]$ { don't pick → $dp[i+1][k-1]$

min

pick → $s_1 = 10$ → $1 + dp[i+1][|s_1| - 1]$

$s_2 = 2$ → $1 + dp[i+1][k-1]$ }

$k = 5$

s_3 → $1 + dp[i+1][\max(s_{i-1}, k-1)]$

Base Case

dp[i][k]

if $k = n - i$

$dp[i][k] = 0$

$k \leq n - i$

must happen for a valid state

if $k > n - i$

then

$dp[i][k] = \text{int}$

final subproblem : $dp[0][0]$

Problem 3: Flowers

(next class)

Problem 4: Removal Game [CSES]

(next class)