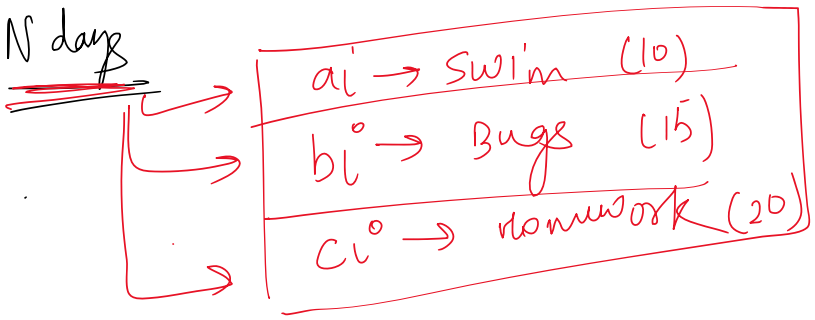
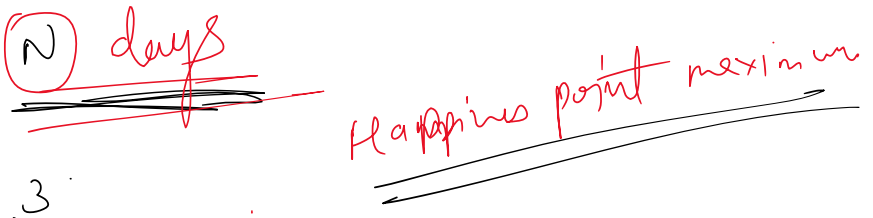
Day 11

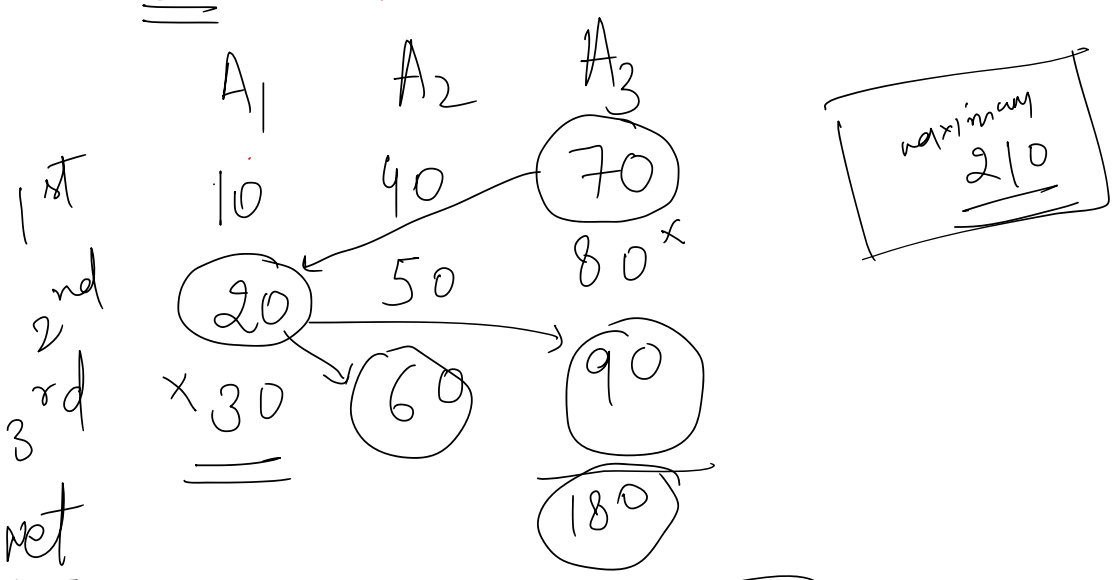
08 December 2022 22:01

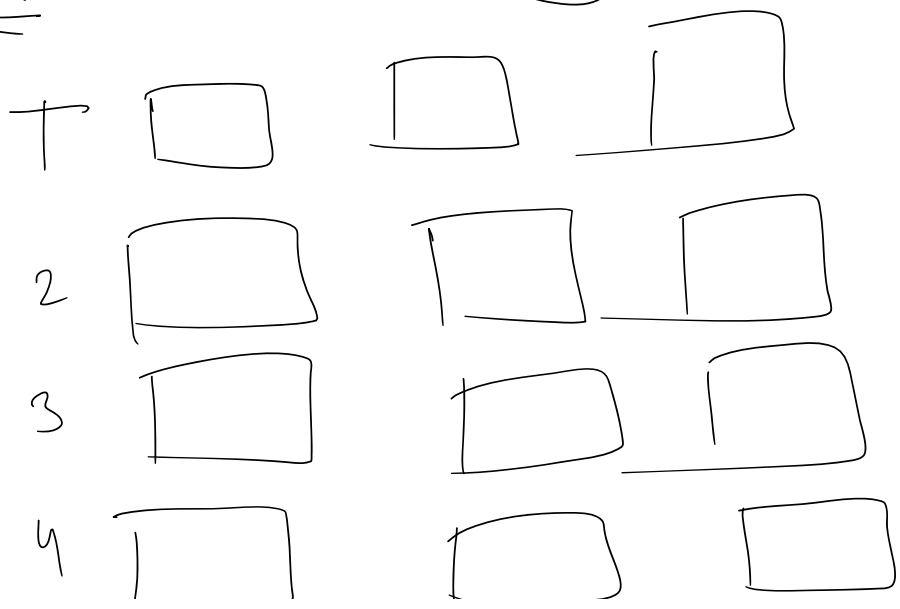


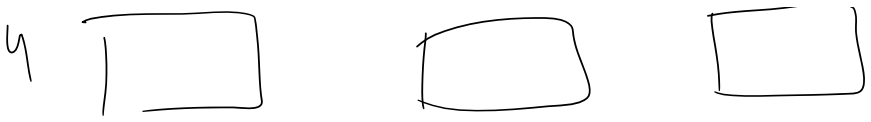
↓ Total

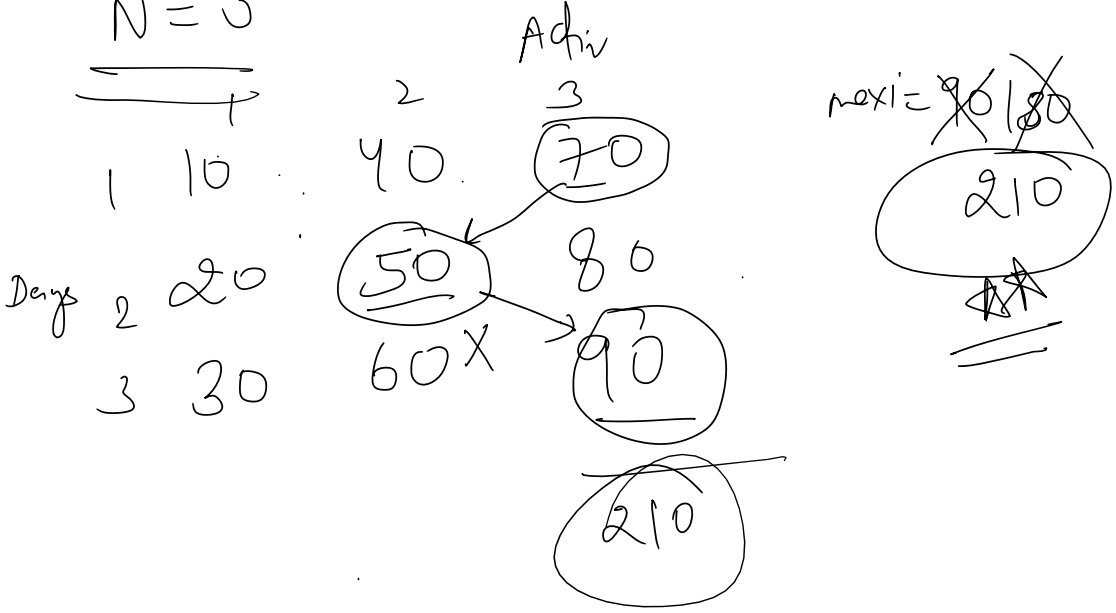


3





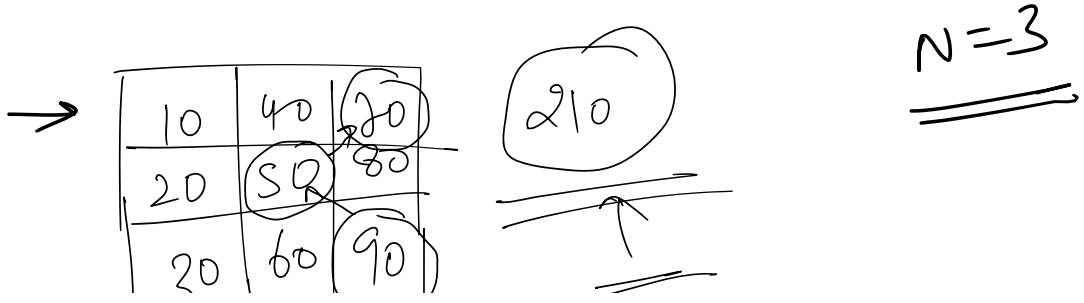


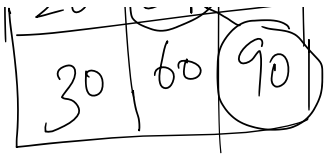


Day 1 →

$n^{th} \rightarrow A_1$

$(n+1)^{th} \rightarrow A_1 \times, A_2, A_3$



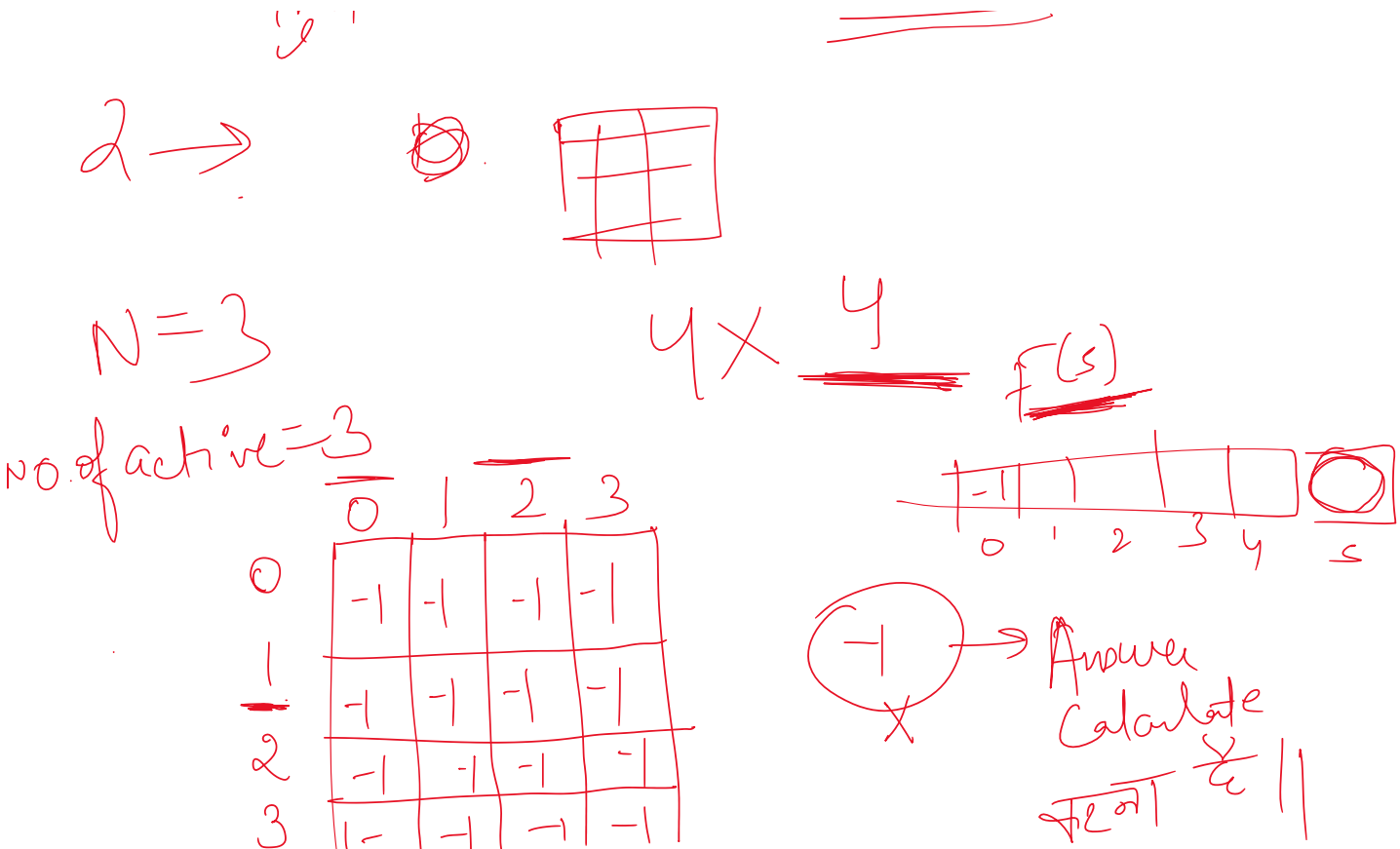



```
int solve(int no_of_days, vector<vector<int>>& acti, int prev_day_activity)
{
    if(no_of_days==0) return 0;
    int maxi = INT_MIN;
    for(int i=0; i<3; i++){
        if(prev_day_activity == i)
            continue;
        // Activity perform
        maxi = max(maxi, solve(no_of_days-1, acti, i)
                   + acti[no_of_days-1][i]);
    }
    return maxi;
}
```

(1) Choice Diagram
(2) Bar Call

Top to Bottom

** जितने variable तेरे Recursive calls में change हो रहे तो उतने Dimension कि पूरी D.P. बनानी पड़ती है।।



Recursion

```
int solve(int no_of_day, vector<vector<int>> &acti, int activity_of_previous_day){
    if(no_of_day == 0){
        return 0;
    }

    int maxi = INT_MIN;
    for(int i = 0;i < 3;i++){
        if(i == activity_of_previous_day){
            continue;
        }
        maxi = max(maxi,solve(no_of_day-1,acti,i) + acti[no_of_day-1][i]);
    }
    return maxi;
}
```

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

S.C- $O(N)$

```cpp
int solve(int no_of_day, vector<vector<int>> &acti, int activity_of_previous_day,vector<vector<int>> &dp){
    if(no_of_day == 0){
        return 0;
    }
    if(dp[no_of_day][activity_of_previous_day]!= -1){
        return dp[no_of_day][activity_of_previous_day];
    }
    int maxi = INT_MIN;
    for(int i = 0;i < 3;i++){
        if(i == activity_of_previous_day){
            continue;
        }
        maxi = max(maxi,solve(no_of_day-1,acti,i,dp) + acti[no_of_day-1][i]);
    }
    return dp[no_of_day][activity_of_previous_day] = maxi;
}
```

T.C- $O(2N)$ $\rightarrow$ $O(N)$

S.C- $O(N)$

loop $\Longleftrightarrow$ recursion

top to bottom

(1) Tabulation (Bottom to top)

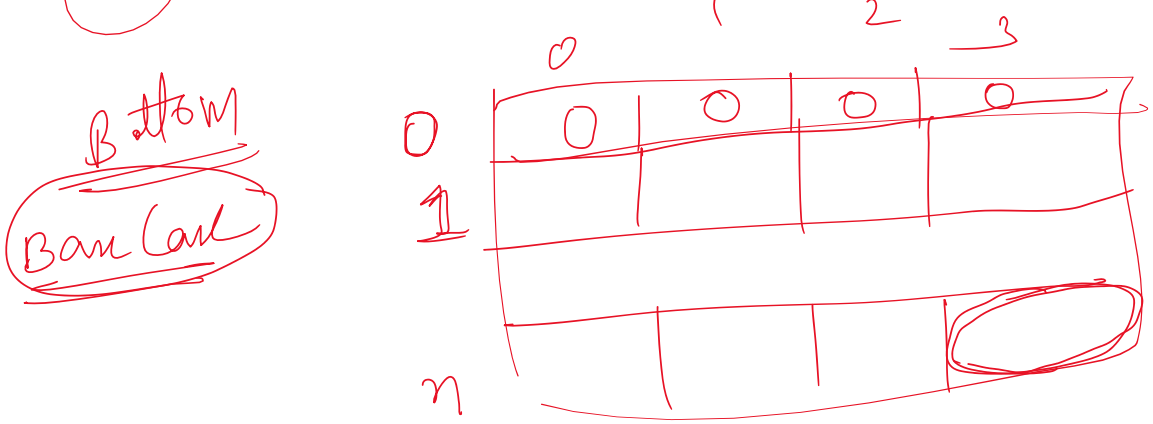
(2) -1 // 0

(2) Recursion Base case $\Longleftrightarrow$

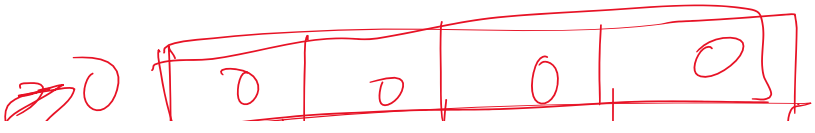
(2) Recursion Base case ⟷ D.p Initialize dp[0]

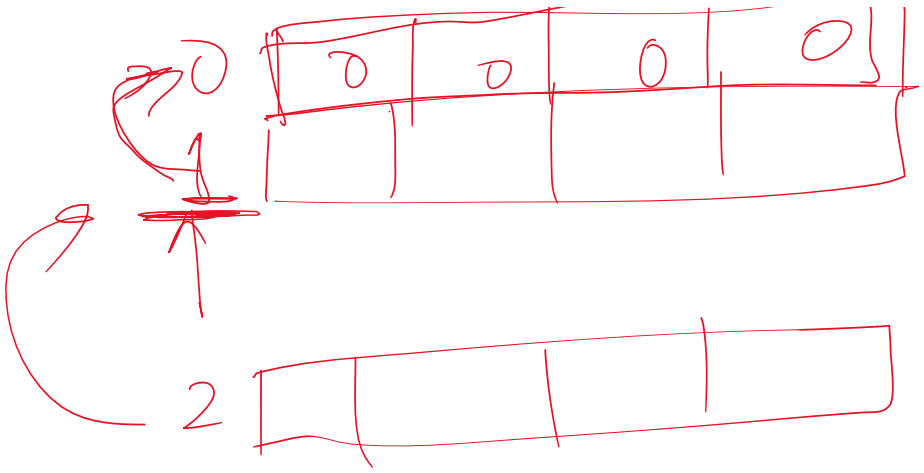
(3) for(int i=1; i<=n; i++) {

(4) Recursive ⟹ dp[n-1]



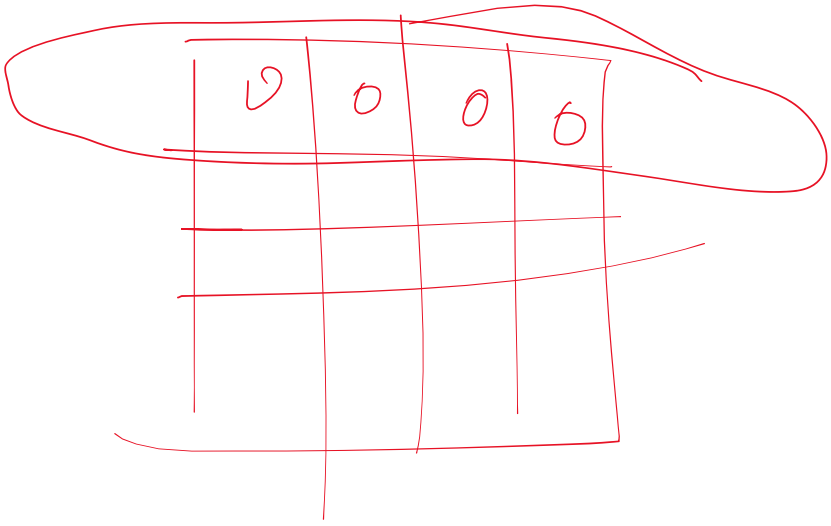
(5) Answer will be present on the starting values of changing variable in recursive calls.

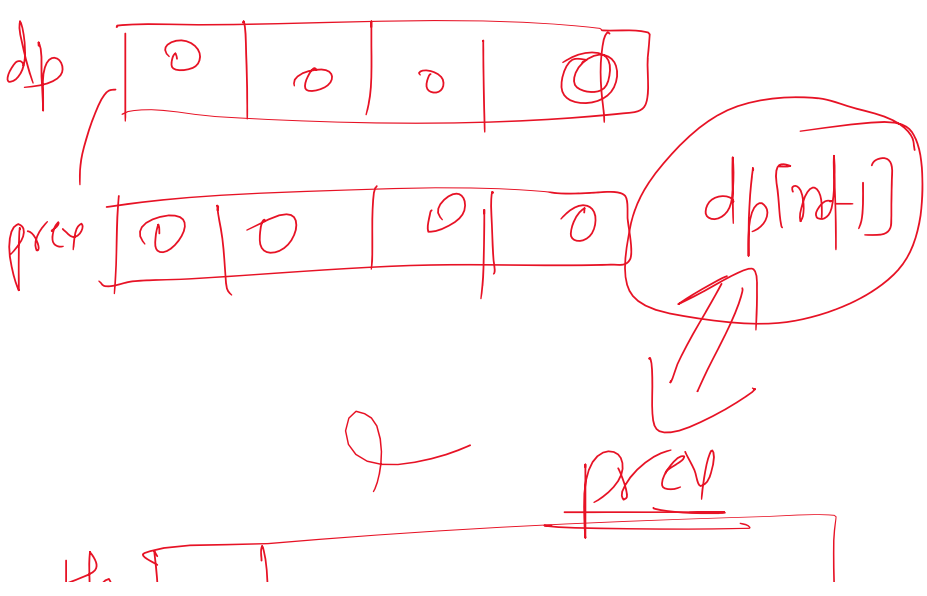


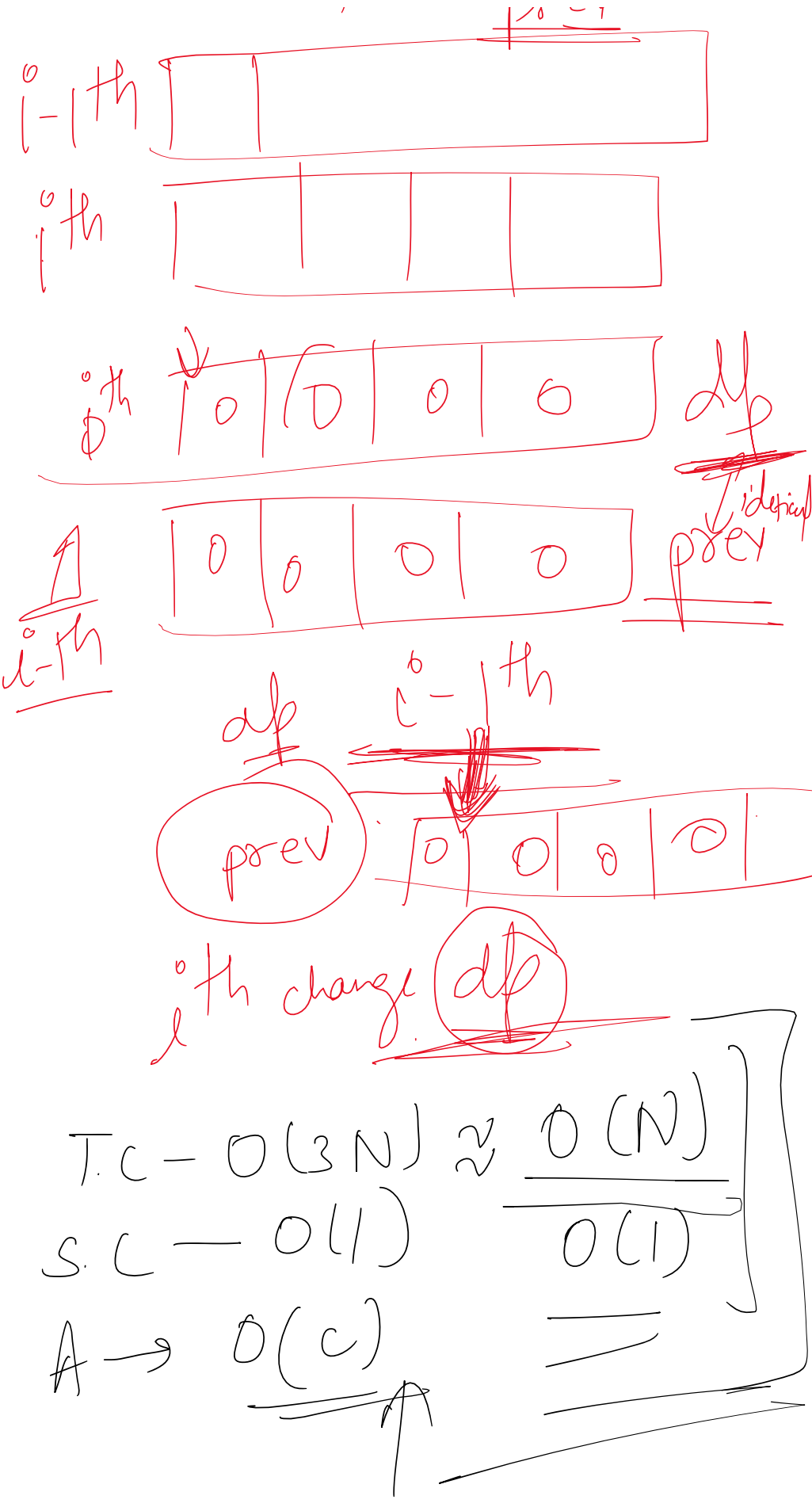


T.C – $O(3 \times N) \approx O(N)$

S.C – $O(3 \times N) \approx O(N)$







T.C — O(3N) ≈ O(N)

S.C — O(1) O(1)

A → O(c)