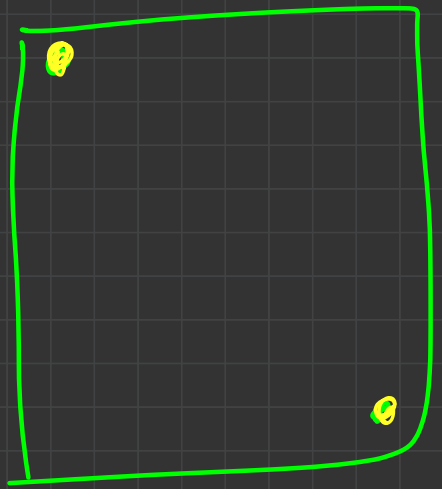
Answer Construction

Dynamic Programming 2.2

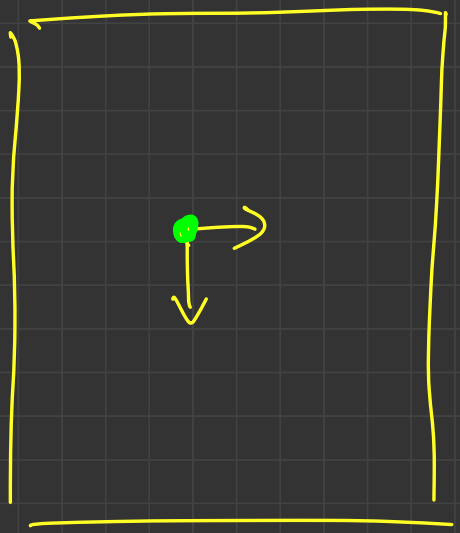
- Priyansh Agarwal

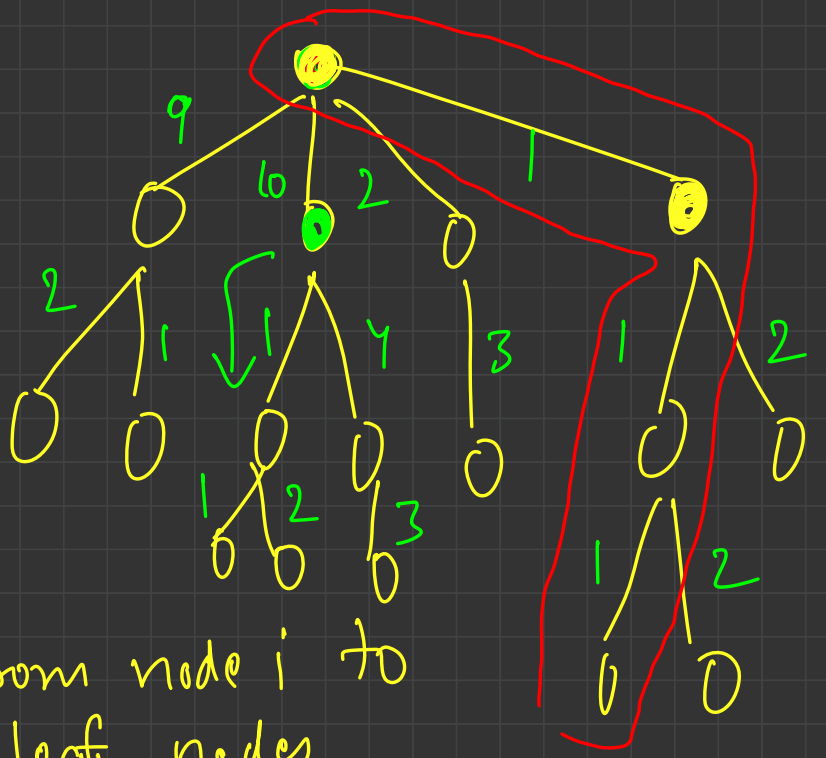


$dp[i][j]$ = min sum path from (i,j) to $(n-1, m-1)$

$dp[i][j] = grid[i][j] + \min(dp[i+1][j], dp[i][j+1])$

$dp[0][0]$



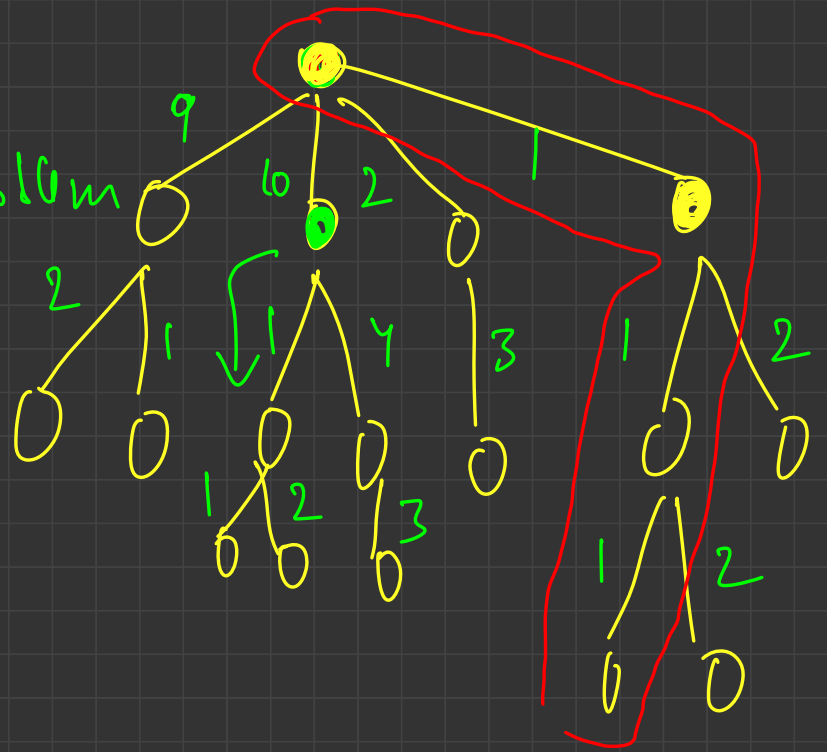


dp(i) = best sum path from node i to any of the leaf nodes

dp(i) = min over all children { edge cost + dp(child)

dp[root] = final subproblem

dp[leaf] = 0

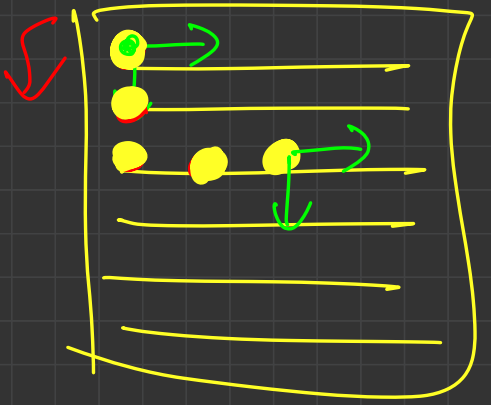


$dp[i][j]$ = min path from (i, j) to $(n-1, m-1)$

$dp[i][j] = g[i][j] + \min \begin{cases} dp[i+1][j] \\ dp[i][j+1] \end{cases}$

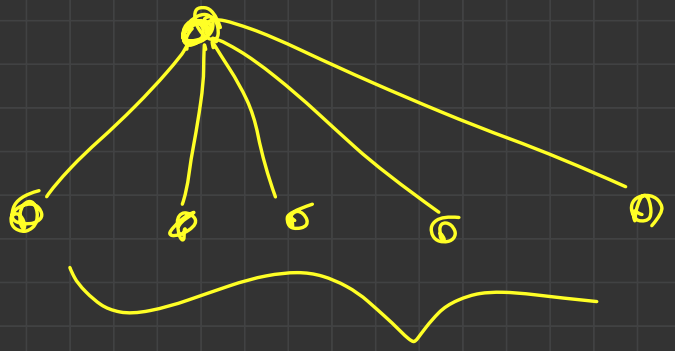
$dp[0][0] = 102$

$dp[1][0] < dp[0][1]$



dp table

While making a choice, if there are on avg T transitions per state, then I will need T iterations to make the best choice



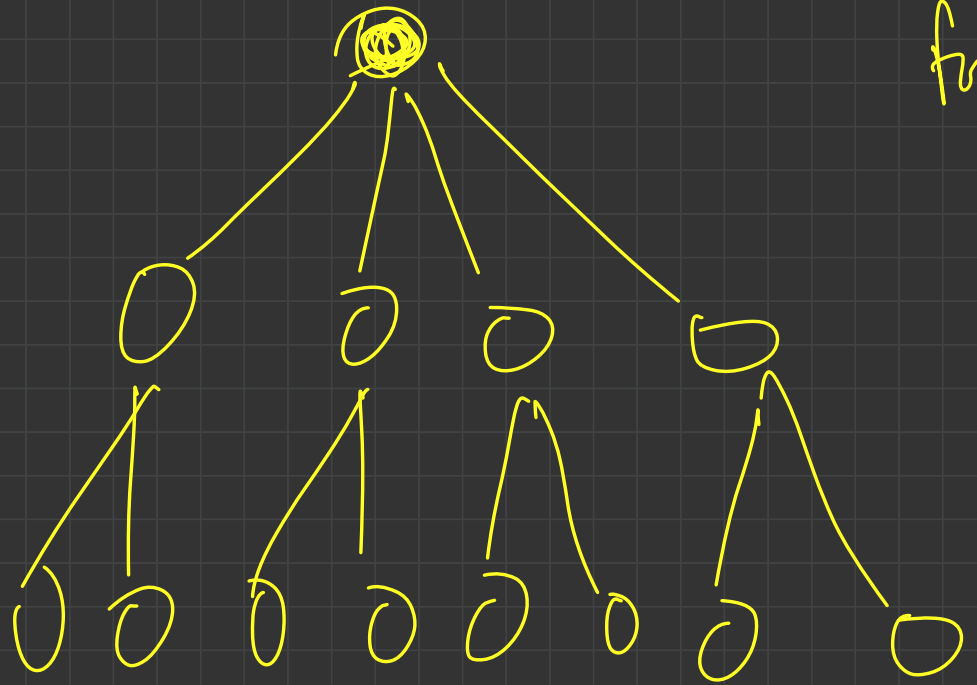
for every state

T transitions

DP population

Answer

Constr



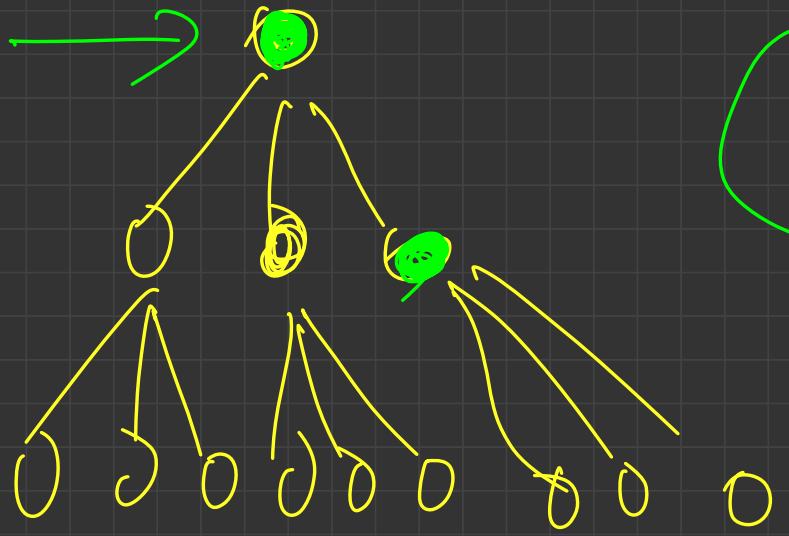
for every state there are T transitions

N states x T trans

$O(NT)$ — dp value calculation

$O(HT)$

Answer construction



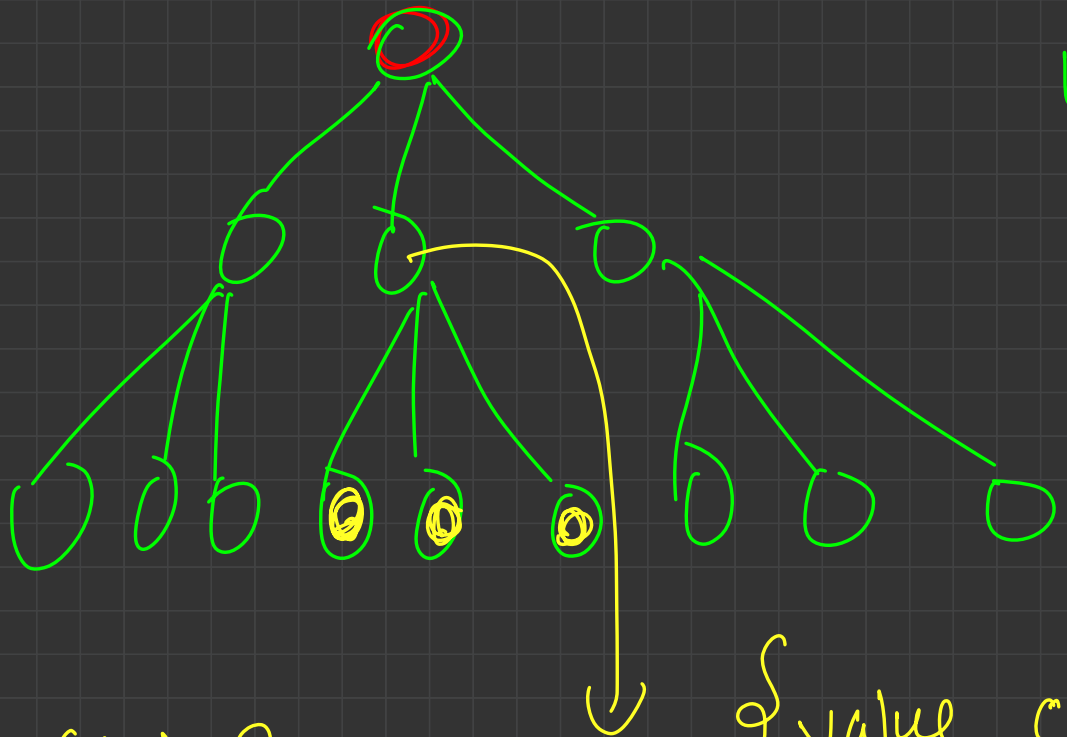
DF population

→ O(NT)

Answer Construction

→ O(HT)

O(HT) O(HT)



while populating dp states

O(H) ?

O(HT) ?

{value, chosen child}

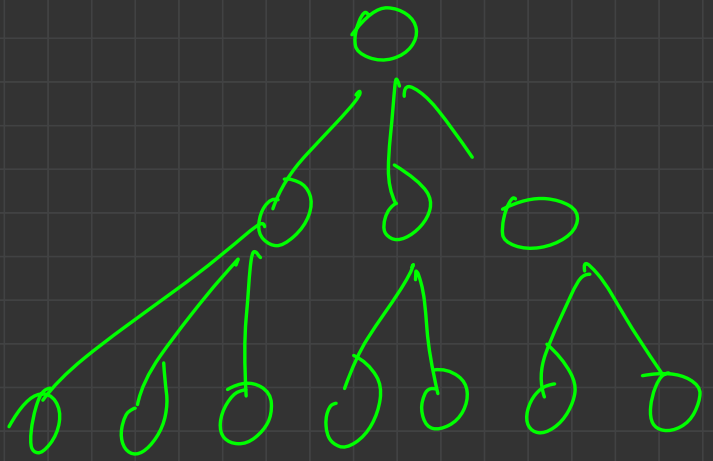
DP population

DP[i] = ——————

DP[i][0] = value

DP[i][1] = chosen child

O(NT) → O(2NT) + O(NT)



N , T , H

$N \leq 10^5$

$T \leq 10^3$

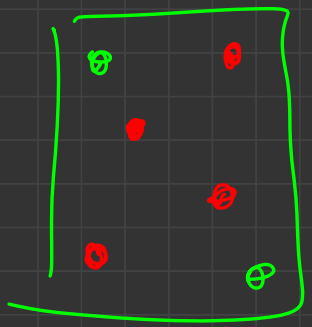
$H \leq 5$

$Q \leq 10^5$

$Q \rightarrow O(HT)$

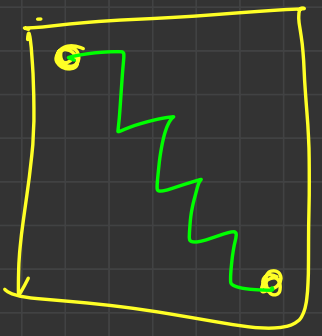
5×10^3

Q queries



multiple queries

1 query



Prob!

① Calculate answer at runtime without storing best choice

② Store best choice and calculate answer based on that

Answer Construction

- Grid problem: Find the actual path with the minimum sum.
- Minimizing coins problem: Find the actual choice of coins.
- At every state we are making some optimal choice.
    - If we store this choice, we can be sure that if we are at any state we know what is the best choice.
    - Start from the state that contains your final subproblem and keep making the best choice (which was already stored) until you reach the end.

Answer Construction - Grid Problem

```cpp
int n = 3, m = 3;
vector<vector<int>> grid(3, vector<int>(3));
vector<vector<pair<int, int>>> dp(n, vector<pair<int, int>>(m, {-1, 0}));
// 0 -> take a down direction
// 1 -> take a right direction
int f(int i, int j){
    if(i == n || j == m)
        return 1e9;
    if(i == n - 1 && j == m - 1)
        return grid[n - 1][m - 1];
    if(dp[i][j].first != -1)
        return dp[i][j].first;

    int ans1 = f(i + 1, j);
    int ans2 = f(i, j + 1);
    if(ans1 < ans2){
        dp[i][j].second = 0;
    }else{
        dp[i][j].second = 1;
    }
    return dp[i][j].first = grid[i][j] + min(ans1, ans2);
}
```

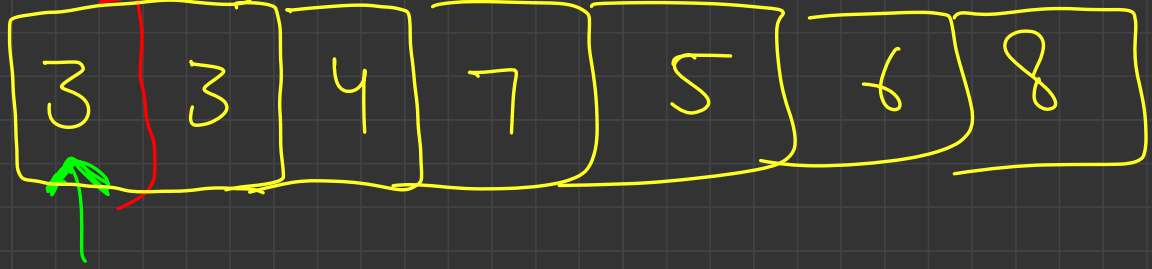
Answer Construction - Grid Problem

```
void solve(){
    for(int i = 0; i < 3; i++){
        for(int j = 0; j < 3; j++){
            cin >> grid[i][j];
        }
    }
    cout << f(0, 0) << nline;
    pair<int, int> current = {0, 0};
    while(current != mp(n - 1, m - 1)){
        cout << current.first << " " << current.second << nline;
        if(dp[current.first][current.second].second == 0)
            current.first++;
        else
            current.second++;
    }
    cout << current.first << " " << current.second << nline;
}
```

Problem Solving

Problem 1: Link

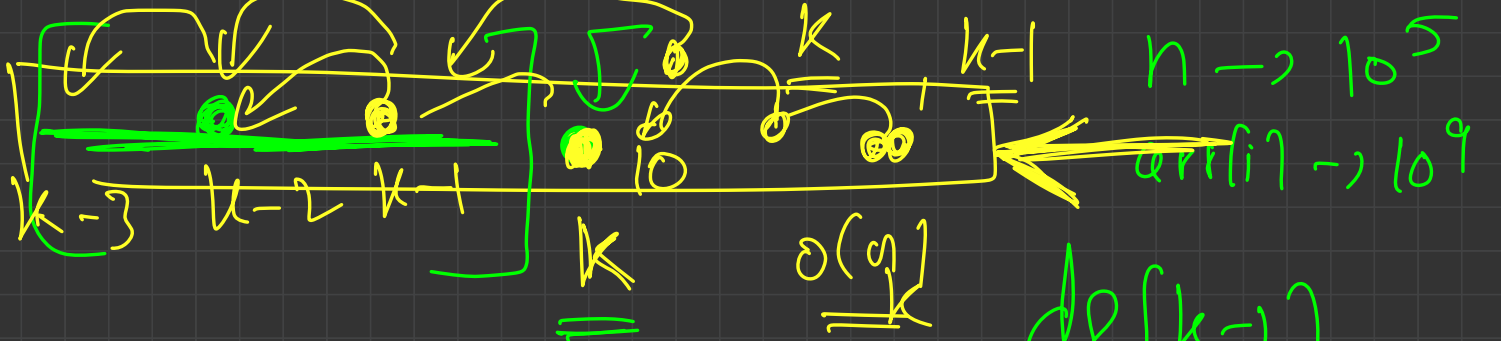
Problem 2: Link



dp[k] = max. length subseq ending with a value k

dp[4] = 2 dp[7] = 1 dp[3] = 1

dp[8] = 2 dp[5] = 3 dp[6] = 4



dp[k-1]

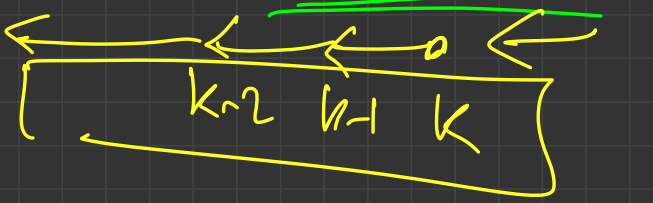
dp[k] = dp[k-1] + 1

dp[k]

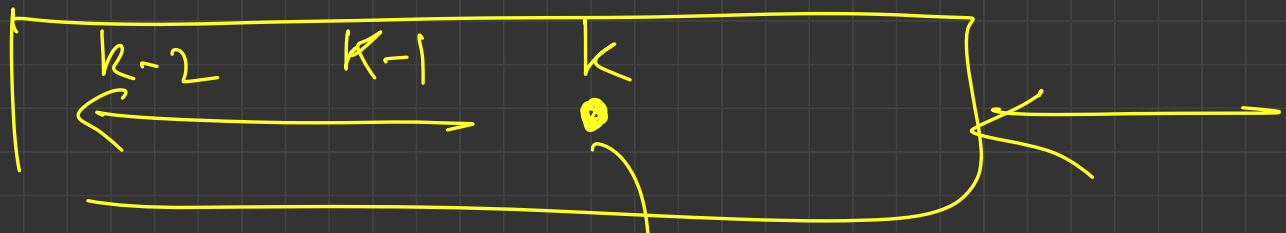
10

0

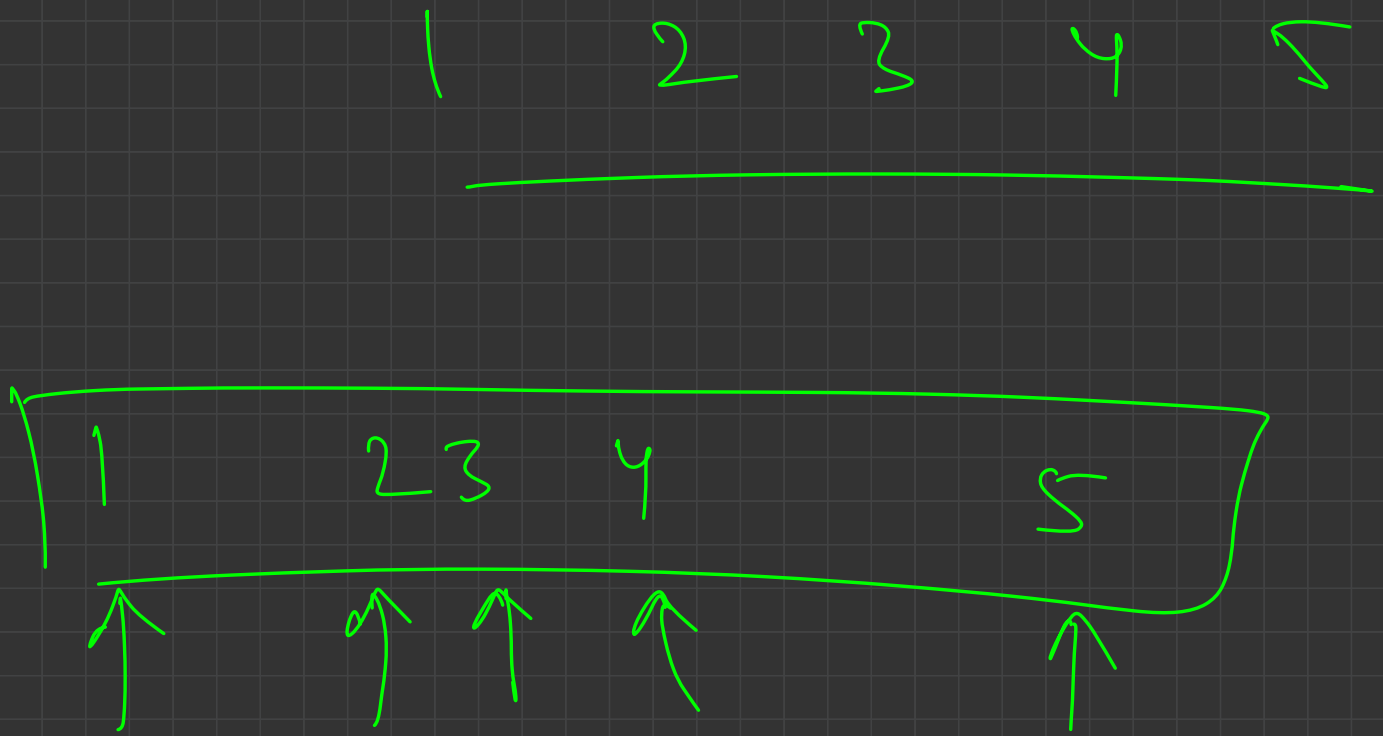
1 —— 10^9

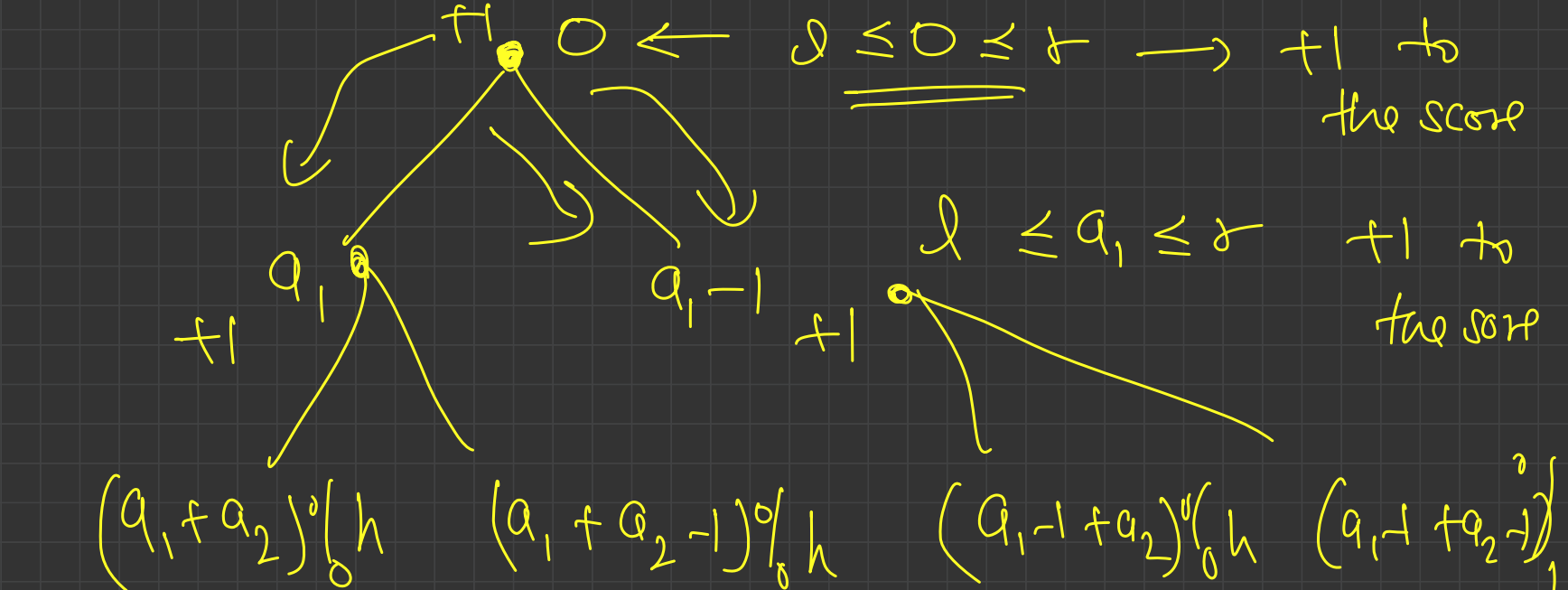


dp[0] —— dp[10^9]



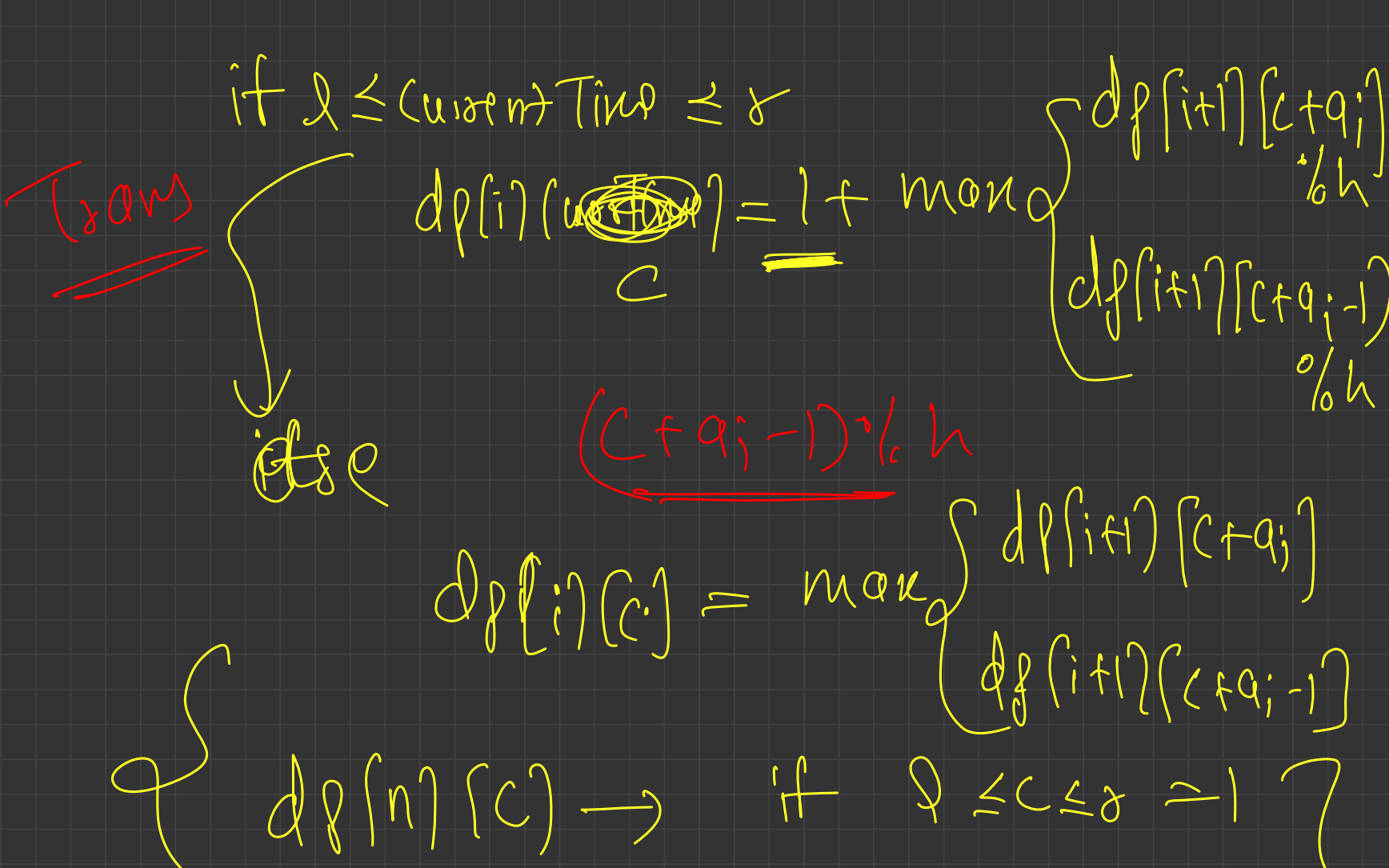
dp[k] → val, index


1 2 3 4 5
1 23 4 5



State

$dp[i][curTime]$ = max good sleeping times you can have starting from the ith sleep such that current time is $curTime$


if l ≤ Current Time ≤ r
Turns
dp[i][(c+q_i)%h]
dp[i+1][(c+q_i)%h]
dp[i+1][(c+q_i-1)%h]
else
(c+q_i-1)%h
dp[i][c] = max
dp[i+1][c+q_i]
dp[i+1][c+q_i-1]
dp[n][c] → if l ≤ c ≤ r ≃ 1

base case

else

-1

dp[i][j]

final subproblem

n

(n + a_i - 1) % 4

$$((c + a[i]) \% (h-1)) \% h + h$$

$$(x + a_i - 1 + h) \% h$$

$$((c + a[i]) \% h - 1) \% h$$

$$((c + a[i]) \% h) - 1) \% h$$

$$\frac{(C + (a[i] - 1 + W)) \% L}{\geq 0}$$

$$L \geq 0$$