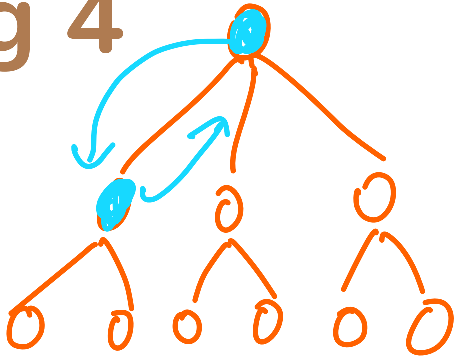
- state optimisation
- transition ''
- non-integer dp parameters

Dynamic Programming 4

- cyclic dp states



- Priyansh Agarwal

State Optimization

4d 5d

- Ask yourself do you need all the parameters in the dp state?
- If you have dp[a][b][c], and a + b = c, do you need to store c as a parameter or can you just compute it on spot?
- If you can compute a parameter in dp state from other parameters, no need to store it.
- Which parameters should you remove? Highest

$dp[i][j][k] \Rightarrow$

$T.T \rightarrow O(n)$

$i \leq n$

$j \leq m$

$k \leq n$

$(n \cdot m \cdot n) \cdot (n)$

$O(n^3 \cdot m)$

$dp[i][j][k]$

$i + j = k$

$dp[i][j]$

dp[i][j][k]

why do we need parameters

dp[1], dp[2], dp[3]

dp[2][3][5], dp[4][1][5]

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

$dp[\underline{\underline{i_1][j_1}}][k_1]$, $dp[\underline{\underline{i_2][j_2}}][k_2]$

$\boxed{i+j=k}$

$$(i_1 == i_2)$$

&&

$$(j_1 == j_2)$$

Can k_1 and k_2 be different

$dp[i_1][j_1][k_1]$ $dp[i_2][j_2][k_2]$

$i_1 == i_2$

&&

$j_1 == j_2$

Yes | No

$dp[i_1][j_1]$ $dp[i_2][j_2]$

$dp[i_1][j_1][k_1]$

$dp[i_2][j_2][k_2]$

$(1, 2)$

$(1, 2)$

$(2, 3), 5$

$(3, 2), 5$

1 2 1 1 2 2 1 1 1 2 2 2

no. of subseq. s.t

n length

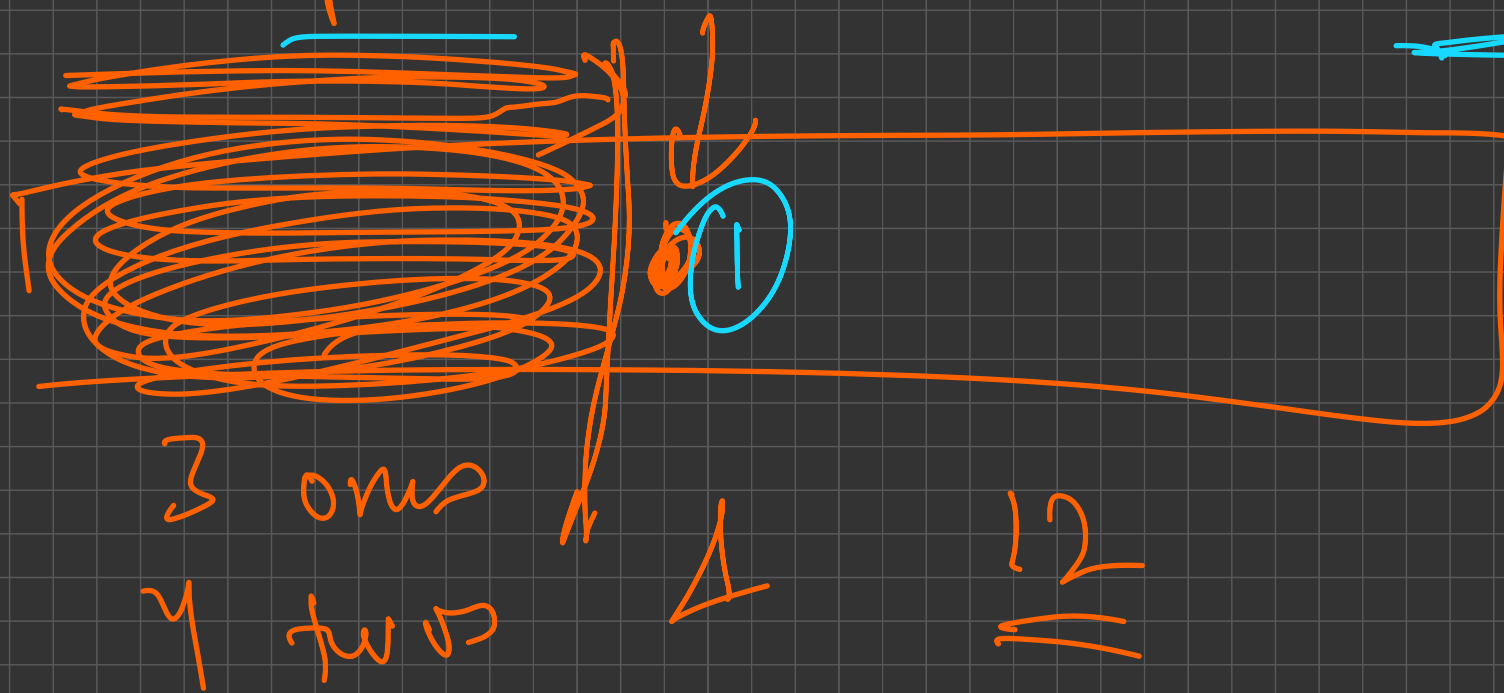
$\#1s \le k_1 = 4$

$\#2s \le k_2 = 5$

dp[index][ones][twos]

= no. of subsequences that you can have provided you are standing at index = index

and no. of ones picked so far = ones and no. of twos picked so far = twos



dp[12][3][4]

1

dp[index][ones][twos]

dp[index+1][ones][twos]

dp[index+1][ones+1][twos]

dp[n][ones][twos]

1 1

0

if $\left.\begin{array}{l} ones \le k_1 \\ twos \le k_2 \end{array}\right\}$ α

$\boxed{index = ones + twos}$

dp[0][0][0]

3 5 5 5 5 3 3 5 5 3 3 5 5 5 3 3

$\# 3s \le k_1$

2^n

$\# 5s \le k_2$

dp[i][j][k]

=> no. of possible arrays such that

you are at index = i and

you have used j <u>3s</u> and k 5s so far

$$dp[i][j][k] \leftarrow \begin{cases} dp[i+1][j+1][k] \\ dp[i+1][j][k+1] \end{cases}$$

$$dp[n][j][k]$$

1 0

when $\left.\begin{array}{l} j \le k_1 \\ k \le k_2 \end{array}\right\}$ o/w

$dp[0][0][0]$

$i = j + \cancel{k}$

dp[i][j] = no. of arrays s/t you are at index = i & you have used j 3s so far

$$\boxed{i} \begin{cases} 3 \\ 5 \end{cases}$$

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

$$dp[n][j] \begin{cases} 1 \\ 0 \end{cases}$$

$\boxed{n^3}$

$1 \rightarrow$ if $\boxed{j \leq k_1}$

$\& \boxed{n-j \leq k_2}$ }

0

$dp[0][0]$

$$dp[a][\delta][c]$$

$$a + \delta = c$$

$$\boxed{a \cdot k_1 + \delta \cdot k_2 = c}$$

$$\delta = (c - ak_1)/k_2$$

$dp[a][b][c]$

$a \cdot c \rightarrow 10^5$

$a \cdot b \rightarrow 10^8$

$b \cdot c \rightarrow 10^7$

$a \leq 1000$

$b \leq 10^5$

$c \leq 100$

linear relation b/w

a, b, c

$a = b + c$

$$\cancel{5}a + \cancel{6}\delta = \cancel{8}c$$

Transition Optimization

- Observe the transition equation.
- Can you do some pre-computation to evaluate the equation faster?
- Using clever observations.
- Using range query data structures

3 more weeks

euler tour

$$dp[i] = dp[i-1] + dp[i-2] \text{ ---- } dp[0]$$

$O(n)$

$$\boxed{dp[0] = 1}$$

$i \le n$

states $\rightarrow O(n)$. T·T / state = $O(n)$

$dp[0] = 0$ $dp[1] = 1$, $dp[2] = 2$

$O(n^2)$

---- $dp(n) = n$

$0 + 1 + 2 \dots\dots n$ $\dfrac{n \cdot (n+1)}{2} = O(n^2)$

$$dp[i] = dp[i-1] + dp[i-2] \ldots\ldots dp[0]$$

$$\underline{\underline{sum[i]}} = dp[i] + dp[i-1] \ldots\ldots dp[0]$$

$$\boxed{\begin{aligned} dp[0] &= 1 \\ dp[1] &= 1 \end{aligned}} \quad , \quad sum[0] = 1$$

$$, \quad \boxed{sum[1] = 2}$$

$$dp[2] = \underline{\underline{dp[1] + dp[0]}} = sum[1]$$

$$dp[0] = 1 \quad , \quad sum[0] = 1$$

$$\left.\begin{aligned} dp[i] &= sum[i-1] \\ sum[i] &= sum[i-1] + dp[i] \end{aligned}\right\}$$

Transition optimisation using a data structure

$$dp[i] = dp[i-1] + \boxed{dp[i-2] \cdots dp[0]}$$

$$dp[i-1] = dp[i-2] + dp[i-3] \cdots dp[0]$$

$$dp[i] = dp[i-1] + dp[i-1]$$

$$= 2 \cdot dp[i-1]$$

$$dp[0] = 1, \quad \underline{dp[1] = 2 \cdot dp[0]} \quad \propto$$

$$dp[0] = 1 \qquad dp[1] = 1$$

$$dp[i] = \underline{\underline{2 \cdot dp[i-1]}}$$

$$dp[2] = 2 \cdot dp[1] = 2$$

$$dp[3] = 2 \cdot dp[2] = 4$$

$$dp[4] = 2 \cdot dp[3] = 8 \leftarrow 2^3$$

$n^2 \longrightarrow \underline{\underline{O(n)}} \longrightarrow \underline{\underline{O(\log n)}}$ $\underline{\underline{dp[n]}}$

T.O

$dp[0] = 1$, $dp[1] = 1$

$dp[n] = 2^{n-1}$

$\underline{\underline{\log n}}$

$\underline{\underline{dp[1]}} = \underline{\underline{dp[0]}}$

$\underline{\underline{dp[1] = 2dp[0]}}$

$\underline{\underline{dp[i]}}$ = some constant

$+ \min(dp[i-1] \, \ldots \, dp[i-k])$

$\underline{\underline{k \le n}}$ $\quad\quad$ $\underline{\underline{i \le n}}$

$$dp[i] = 5 + \underline{\underline{\min}}\left\{ dp[i-1], \ldots, dp[i-100] \right\}$$

$$\underline{\underline{i \leq n}}$$

$$\underline{\underline{O(n \cdot 100)}}$$

$$\begin{cases} dp[i] = 5 + *(\text{multiset.begin}()) \\ \underline{\text{multiset.erase}(dp[i-100])}, \ \underline{\text{multiset.insert}(dp[i])} \end{cases}$$

$dp[0]$

$dp[i]$

$$dp[i] = 5 + \min\{dp[i-1], dp[i-2], \ldots, dp[i-50]\}$$

$dp[i+1]$

$dp[i-49]$

$\log.(100)$

k

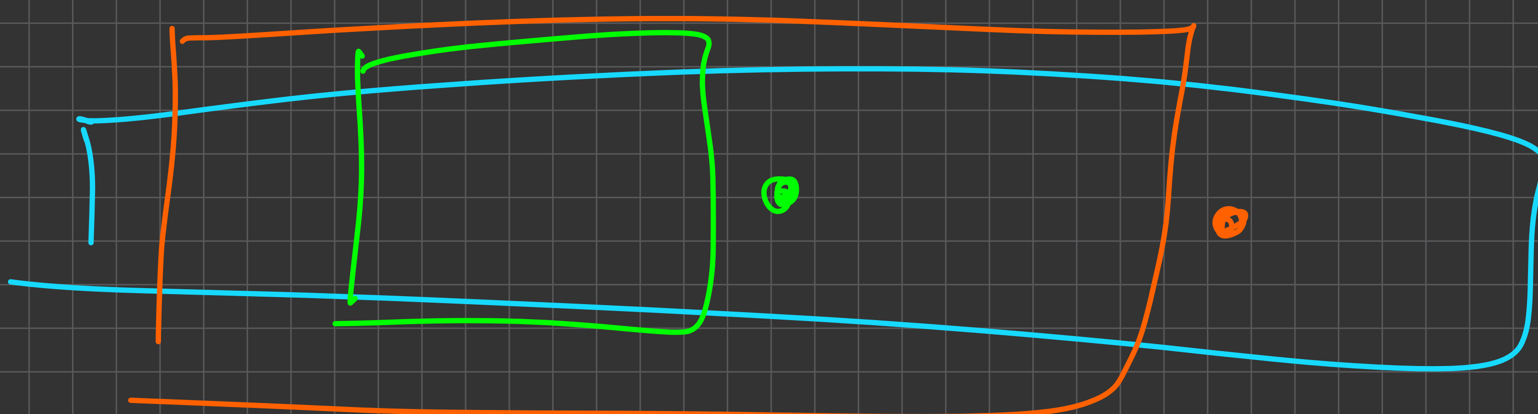
$O(n \cdot k) \longrightarrow O(n \cdot \log k)$

$$dp(i) = a_i + \boxed{\min\left(dp[i-1] \ldots dp[i-a_i]\right)}$$

$$\begin{array}{cccccc} 1 & 1 & 2 & 2 & 4 & 3 \\ a_1 & a_2 & \ldots & & & a_n \end{array}$$

$$a_i \le i$$

$$(\qquad\qquad) \min$$



Representing non-integer parameters

- How will you store the dp states if instead of integer parameters you had a string or a vector or a map or any complex data type?
- Use a map instead of an array.
- Tradeoff map<pair<int, string>> DP or vector<map<string>> DP

dp[i][j][k]

≥ 0

10^9

$(i, j, k) \geq 0$

# of states
$(i, j, k) \leq 10^6, 10^7$

dp[i]["Priyansh"]

vector<string>

vector<string> A

A[0] → Priyansh

A[Priyansh] = something

v("Priyansh") = something

v[0], v[1] = "Priyansh")

map < pair<int, string> , int > dp

10^3 10^3 10^2

10^7

input → calculated —No→ Calculate → Store

calculated ↓ Yes → return

Store ↓ → Return

int arr[n]

arr[k] = some value

map[key] = value

256 MB

int arr[10^9]

int arr[10^7]

$2 \cdot 10^7$

$3 \cdot 10^7$

dp[3][5]

map < pair<int, int>,int>

dp

int dp[100][100]

dp[3][5]

dp[{3,5}]

dp[int][character]

dp[int][double]

string $\longrightarrow$ SPSSSPP??PS??SSS

$n \leq 10^5$

n characters $\longrightarrow$ S, P, ?

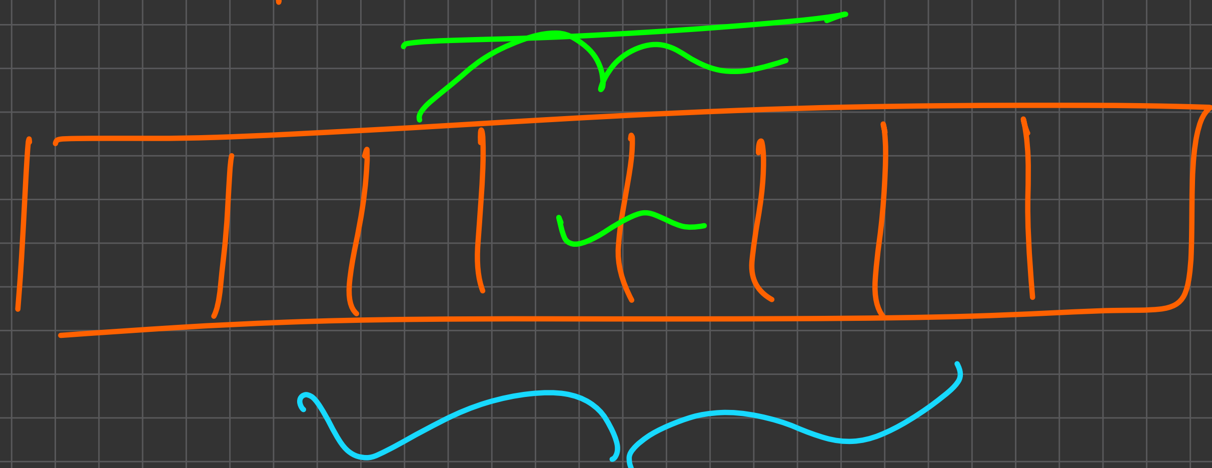
replace every ? with a S or P

provided there is no palindrome

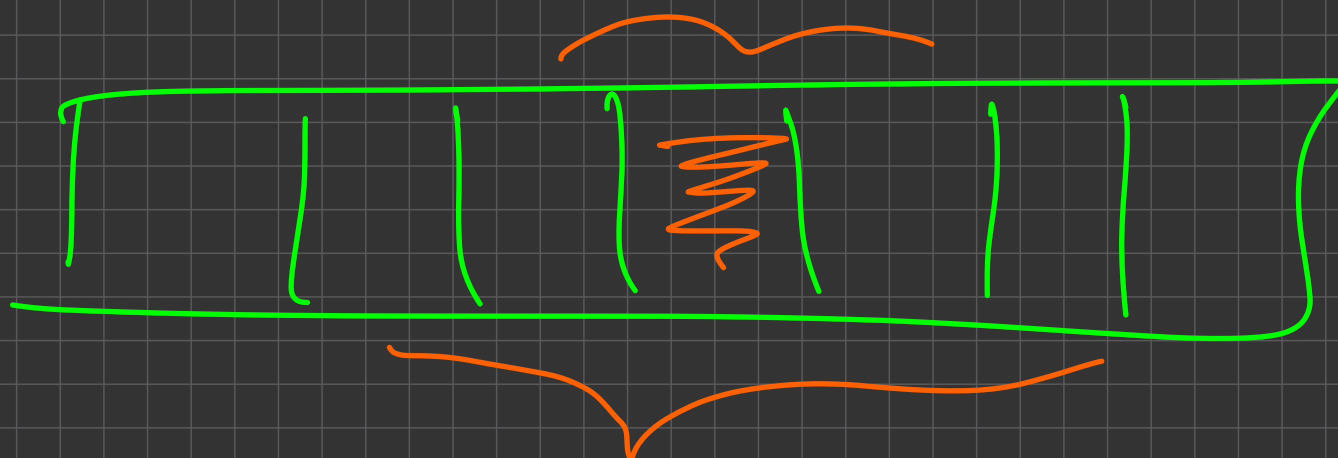
in this string of length ≥ 5

2^n

palindrome of length 8



palindrome of length $\underline{\underline{7}}$



find out if it is possible to not have a palindrome ≥ 5 length if there is no palindrome of length 5

$\Rightarrow$ 6 7

~~7~~

8 9 $\times$

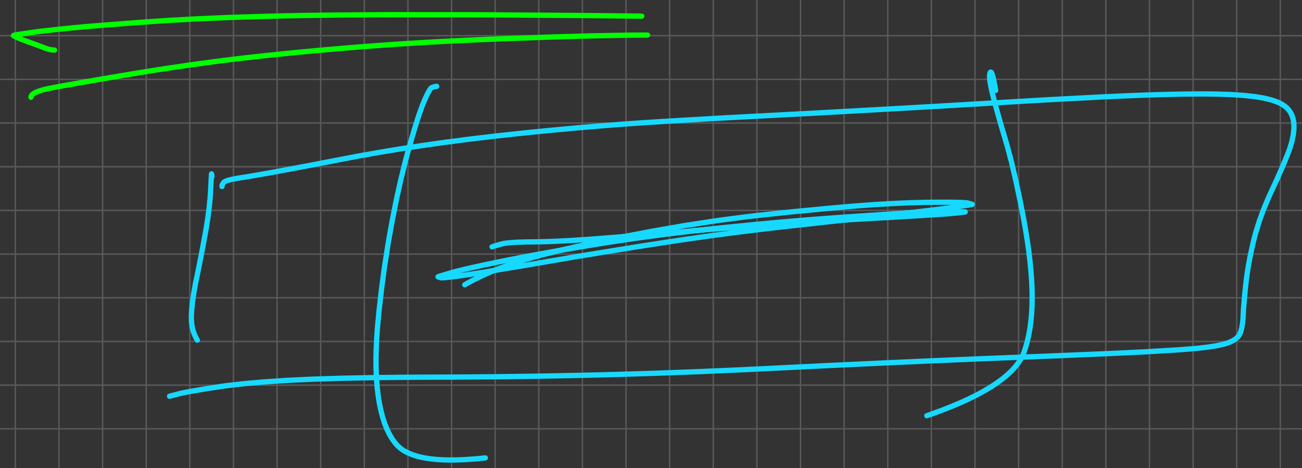
9 11 $\times$

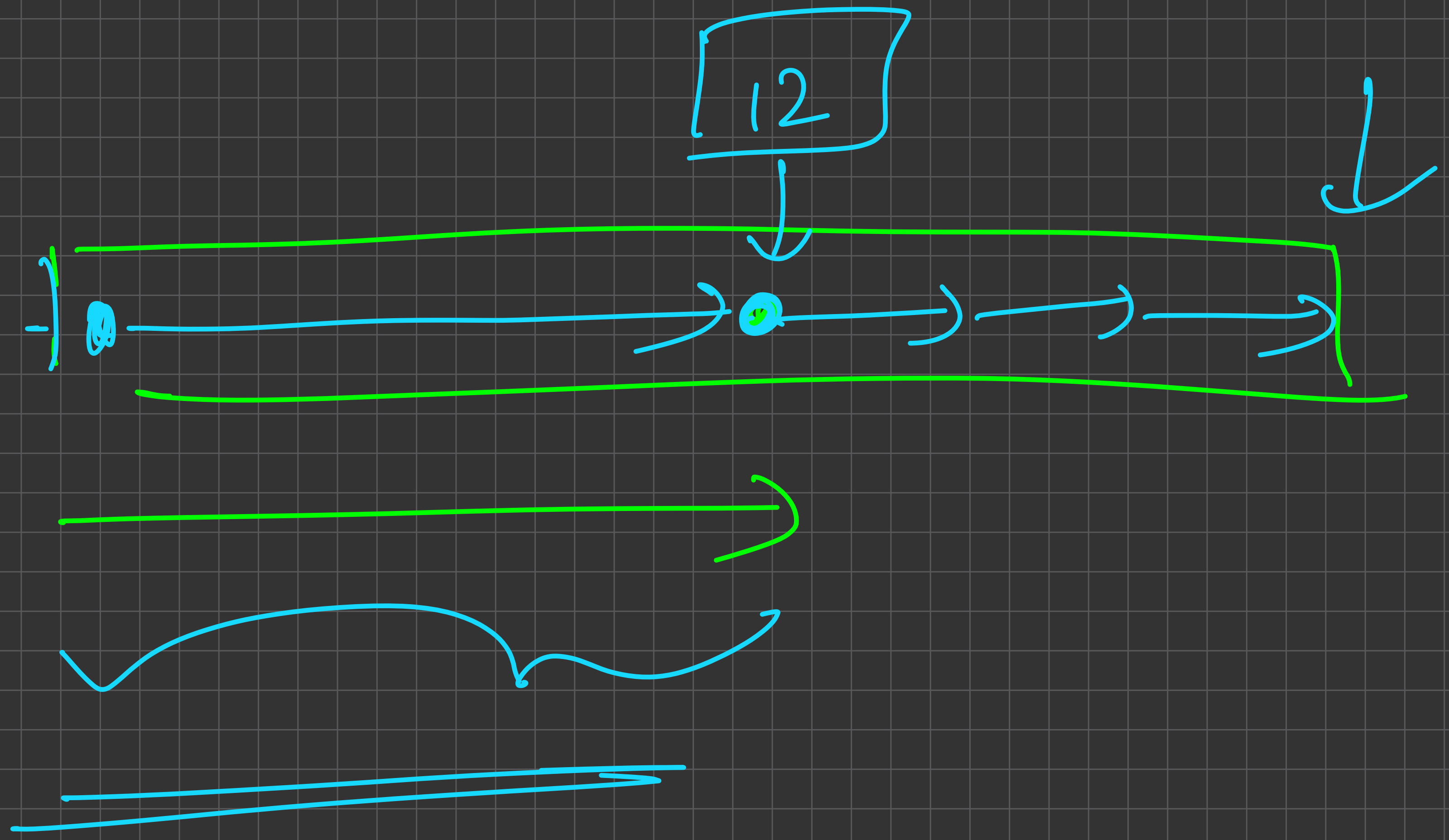
10

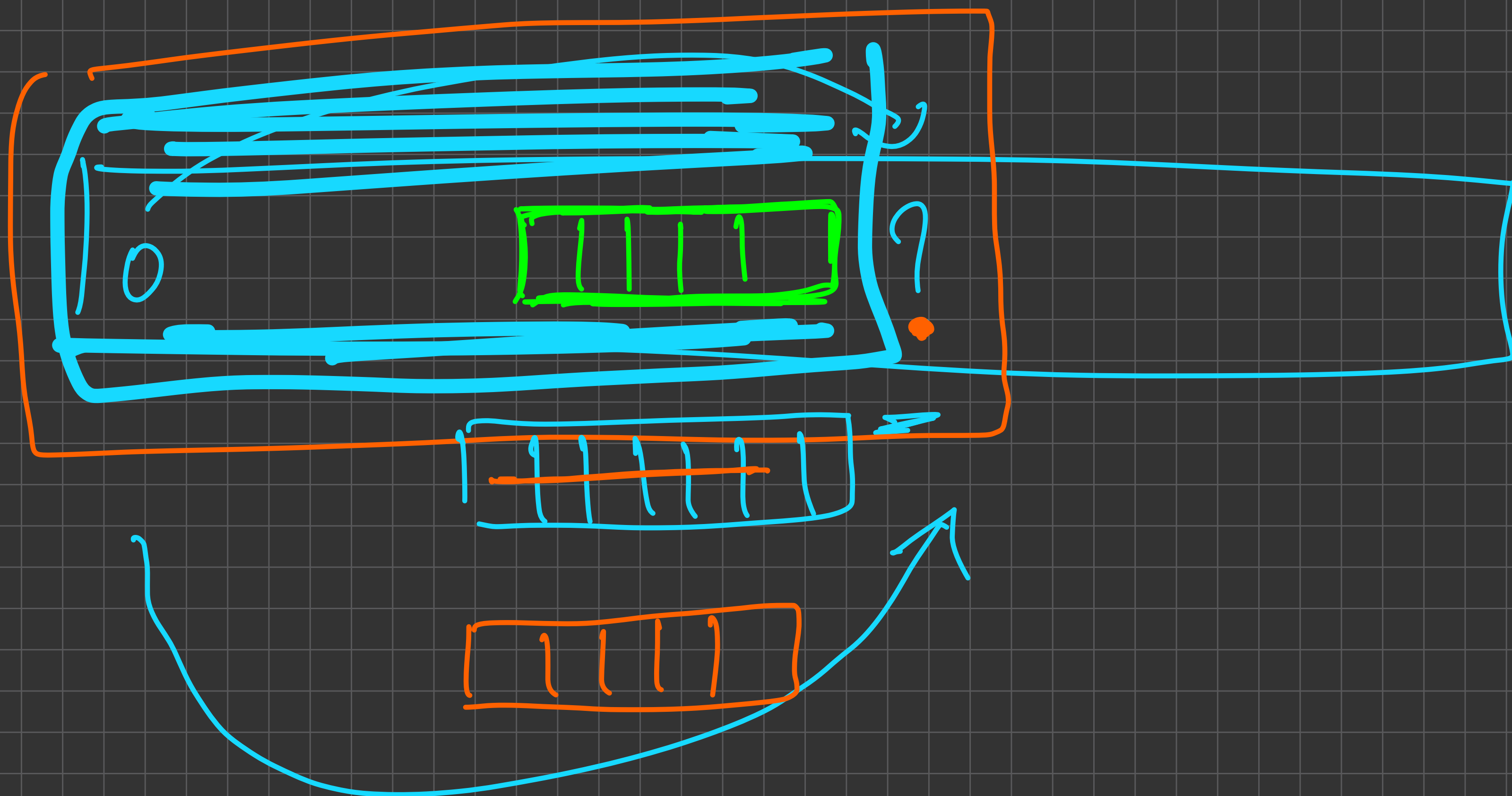
we dont want any palindrome

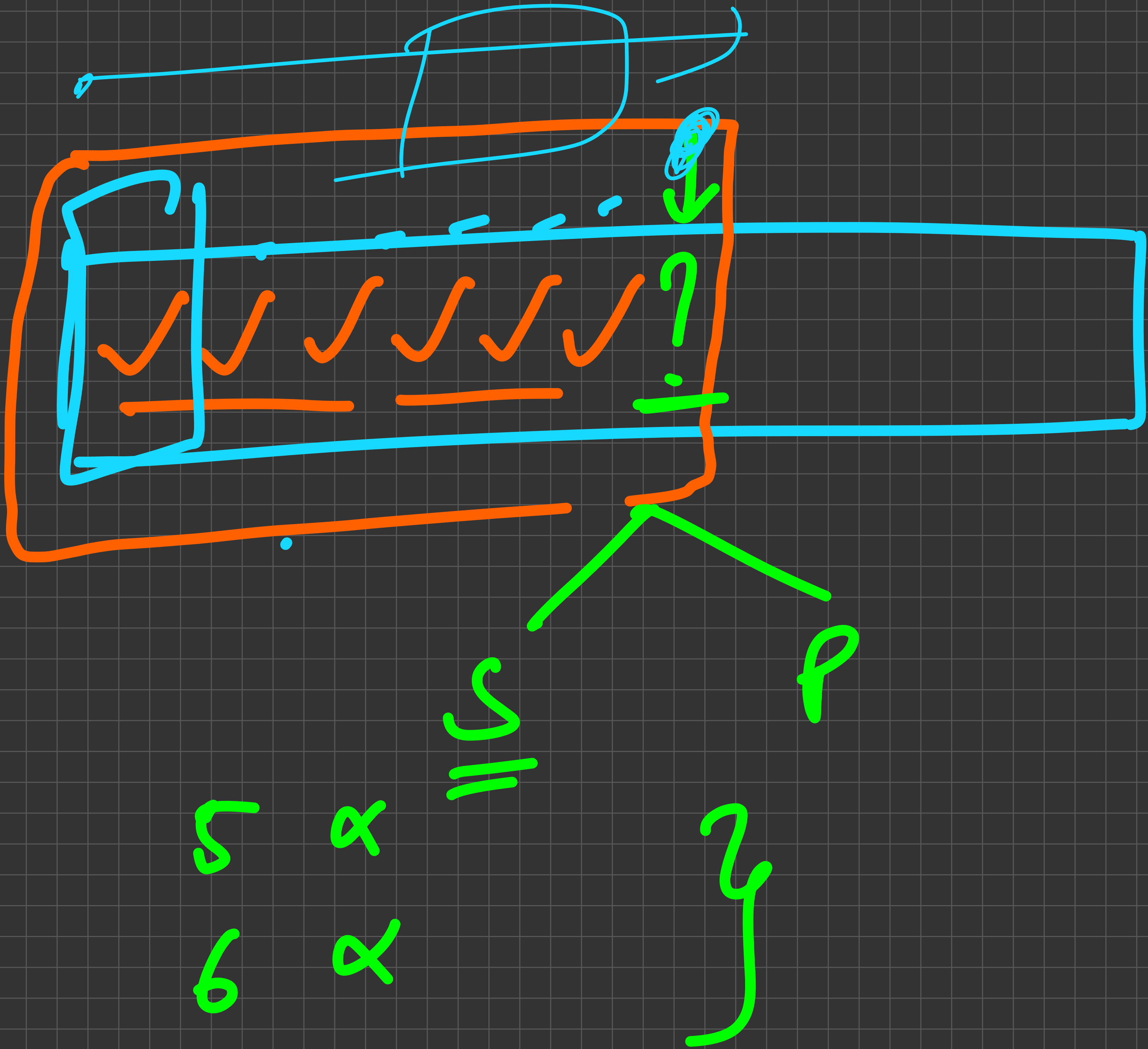
≥ 5 length

5 & 6




12


0
?


?
S
P
5 α
6 α

dp[index][last 5 characters]

= is it possible to replace all question marks from (index to n-1) s.t there is no palindrome of length of 5 or 6

✓ ✓ ✓ ✓ ✓ ✓ ✓ $\boxed{?}$ S S S P P S S — -

? ? S S S

S S

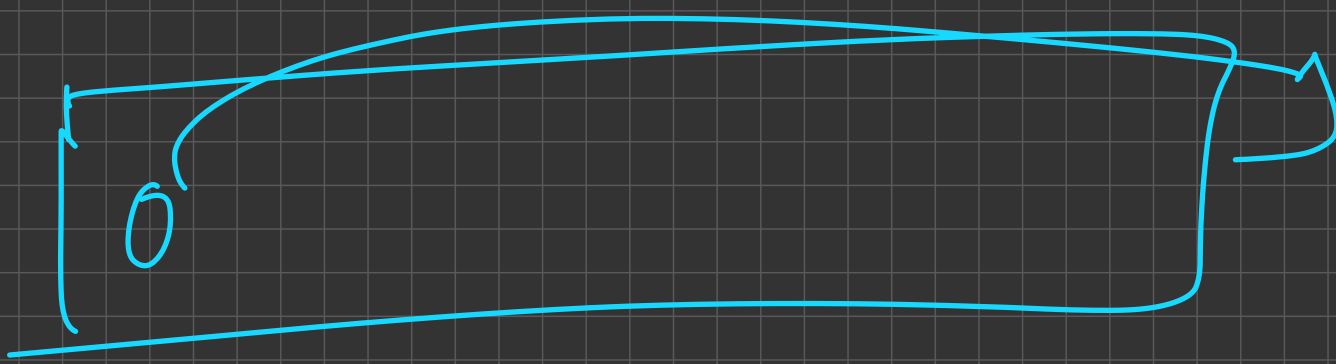
S P

P S

P P

dp[n][any 5 character]

= true



dp[index]["last 5 characters"]

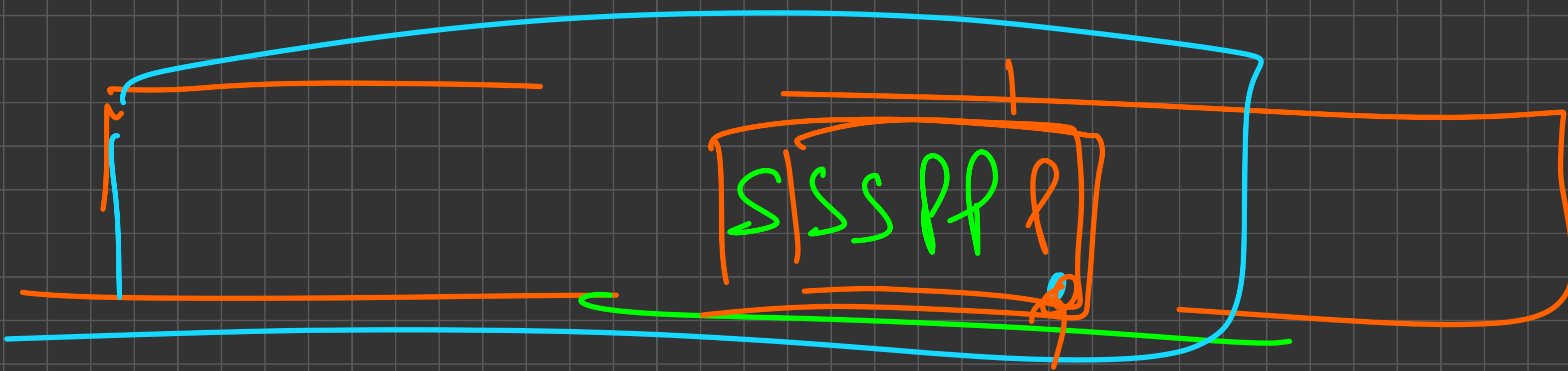
s p ?

if last 4 characters + s don't make a palindrome

&&

last 5 characters + s don't

dp[11]["sssPP"] = ?



dp[11]["sssPP"] = dp[12]["ssPPS"]

|| dp[12]["ssPPP"]

dp[index][l5]

= you have already filled all the ? marks from 0 to index − 1,

is it possible to replace all ? from index to n−1 ensuring that there is no palindrome of len 5 or 6 dp[index]["last 5"]

= dp[index+1]["last 4 characters" + s]

dp[0]["~~dddd~~"]

S

dp[0]["*****"]

?????p

?ss?p

State

Transition

B - C.

R · S

State :

dp[index][last 5 characters] =

we have already filled the string from 0 to index-1 and we have ensured that there is no palind of leng 5 or 6 in this substring, is it possible to replace ? in the string from index to n-1 such that there are still no palind of length 5 or 6

Transition

① if index has 5

if ~~5~~ doesn't make a palindrome with last 5 or last 6

dp[index][last 5] = dp[index+1][last 4 + s]

if doesn't hold return

dp[ind][last i] = false

② ✓

③ ?

B.C.

$$dp[n][\text{anything}] = \underline{\underline{\text{true}}}$$

F.S.

$$dp[0]["\underline{\underline{\text{TLETL}}}"]$$

$$dp[0]"\underline{\underline{\text{LLLLL}}}"]$$

T.C

states

n : 32

S.P

dp[i][last 5]

n 32

O(1 iterations)

$n \cdot 32 \cdot 11 \longrightarrow$ T.C

non integer

dp

param

S.C $= 32n$

C Google Kickstart

AIR 1 GR 6

$\boxed{n \le 12}$ 30

$n \le 10^5$ 100

2^n

A B C D

800 1400

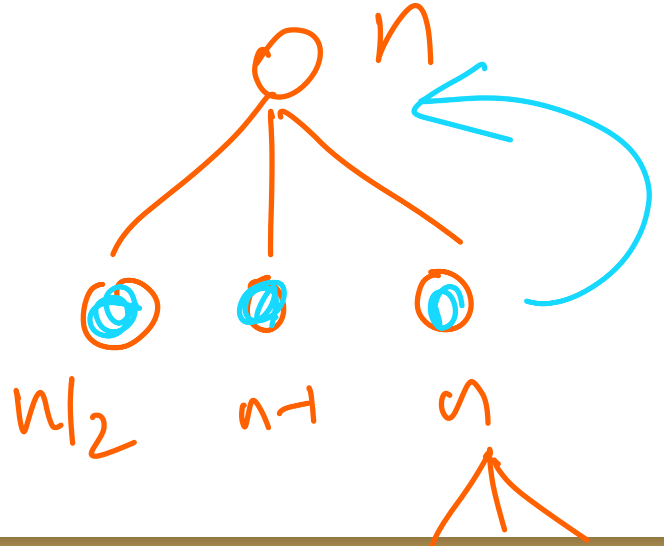
2100 ~~2000~~

Problem:

- State:
  -
- Transition:
  -
- Base Case:
  -
- Final Subproblem:
  -

Cycling DP states

- What happens when your current state is dependent on itself?
  - dp[i] depending on dp[i] itself



Problem 1

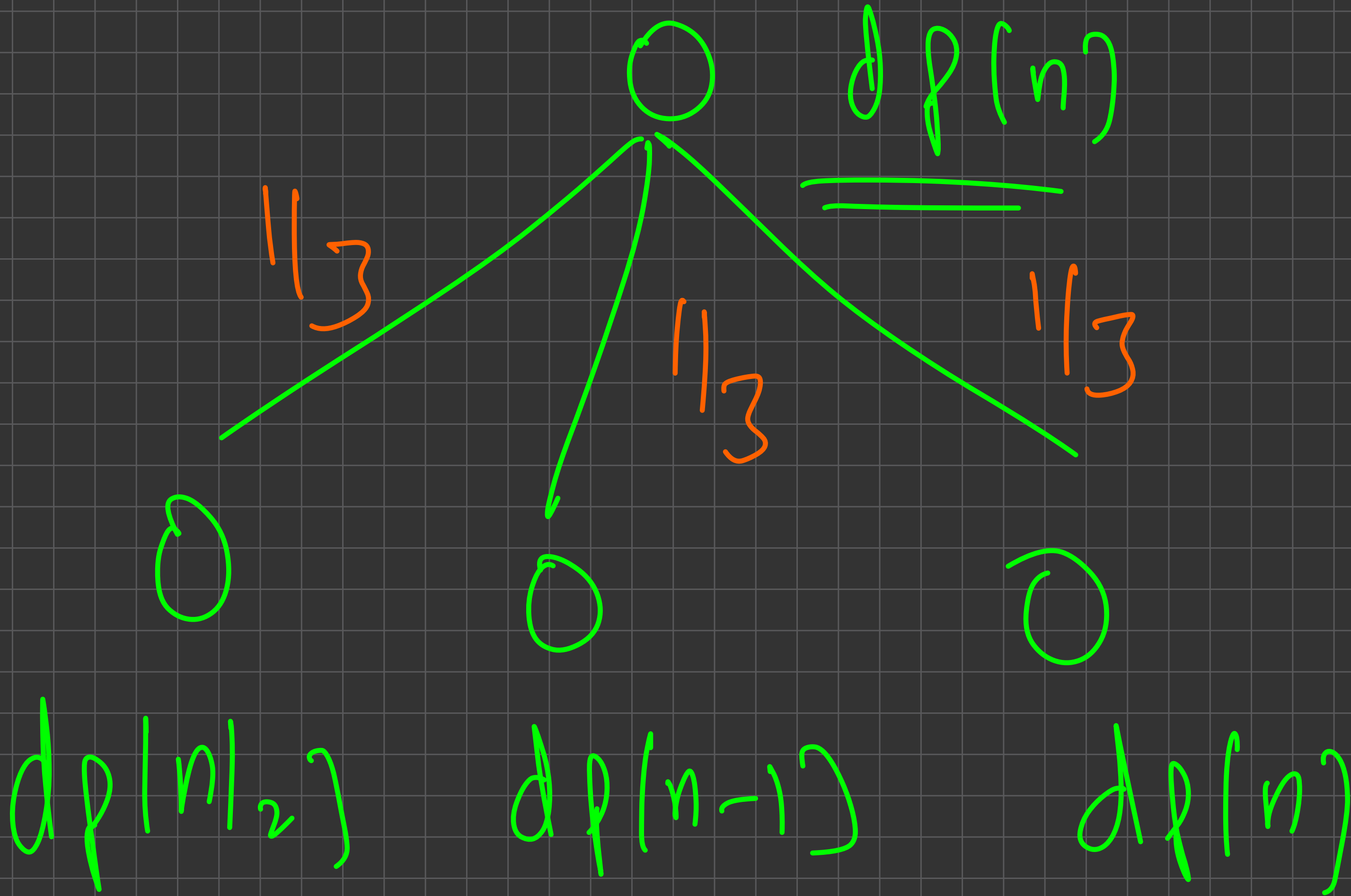
n

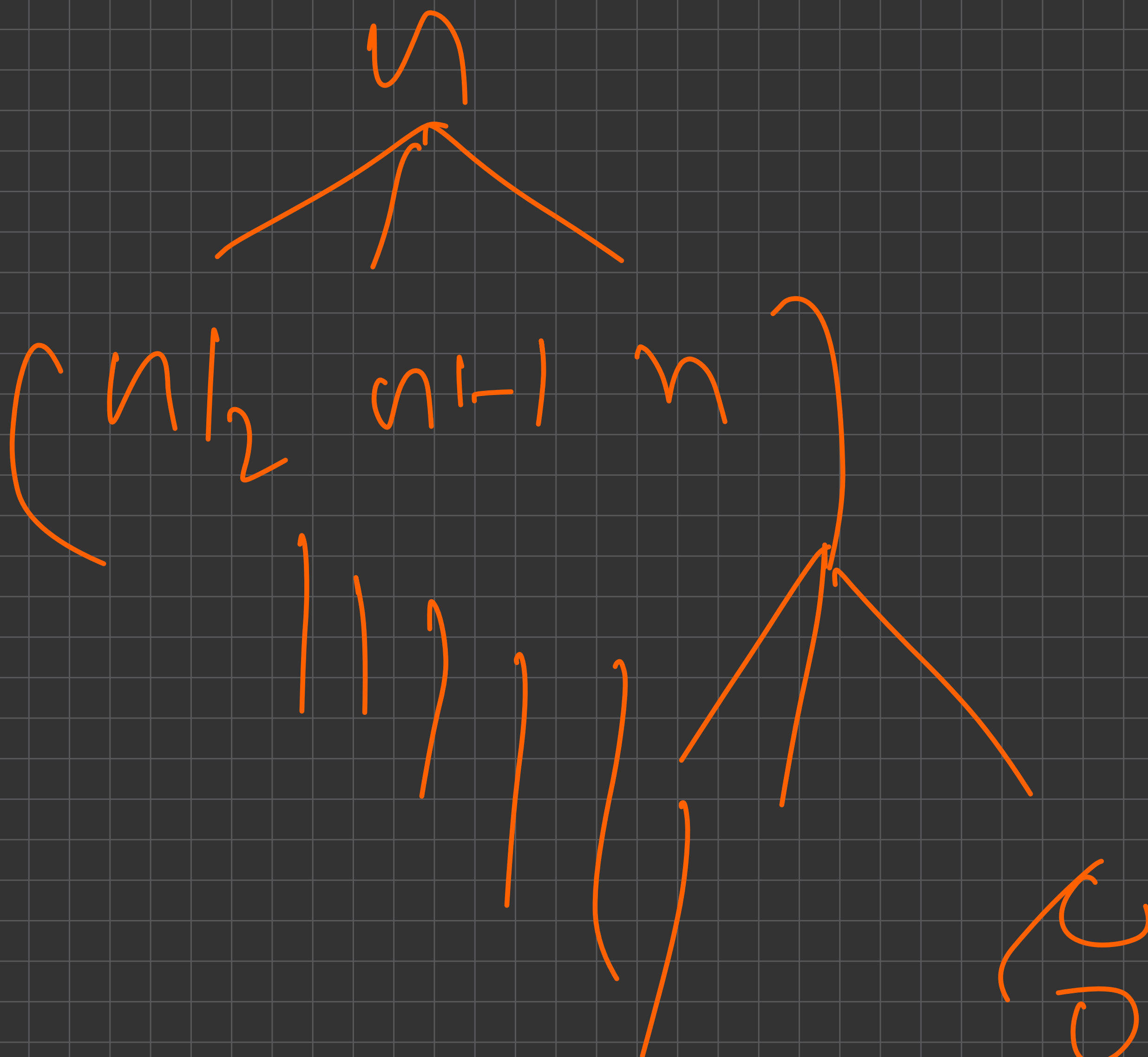
$n-1 \quad n/2 \quad n$

- Given a positive integer N <= 1e6, at every step the following 3 things can happen to N with equal probability.
  - N = N / 2
  - N = N - 1
  - N remains unchanged
- Find expected number of steps it will take to convert for N to become 0

$dp(0) = 0$

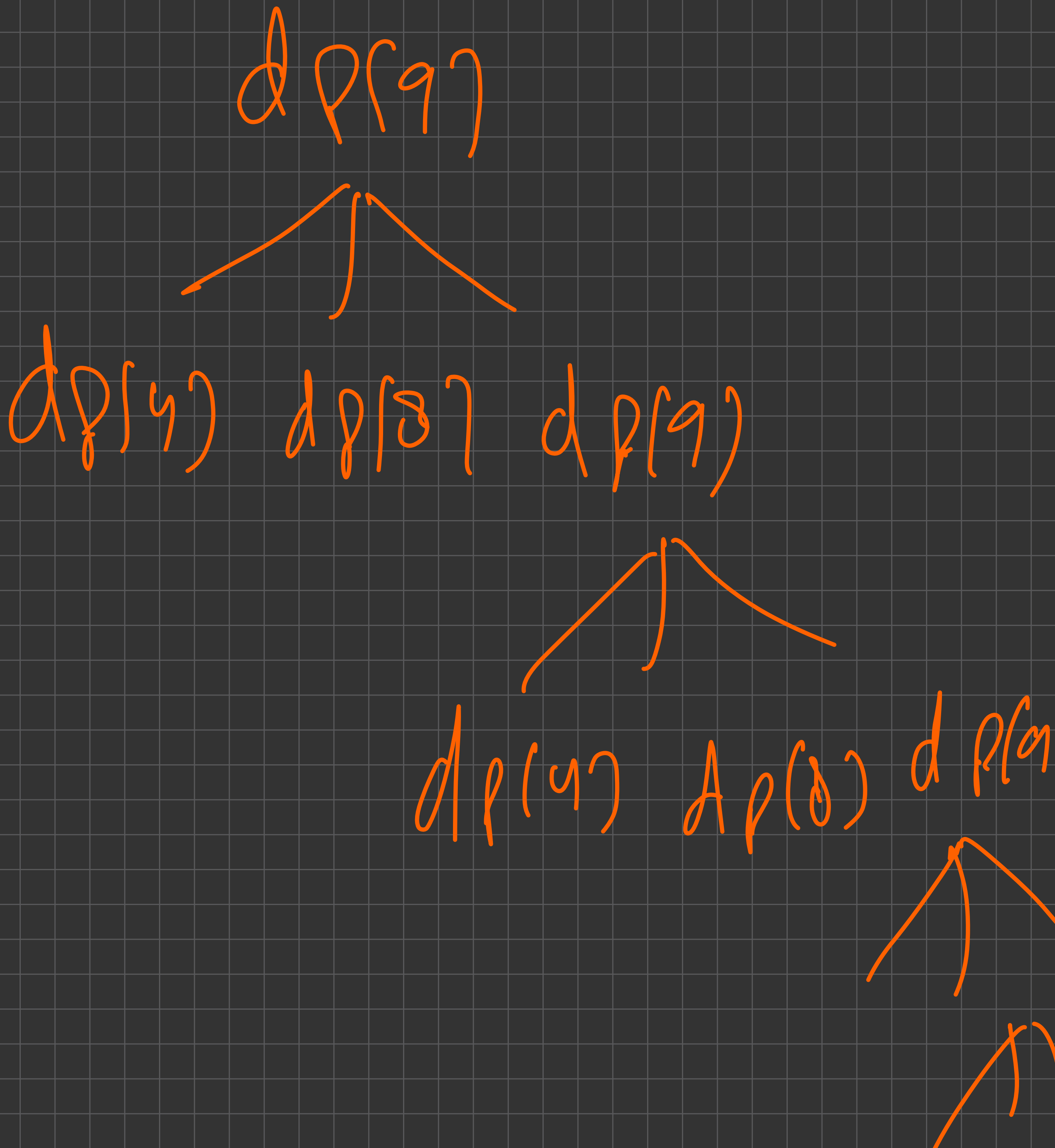
dp[n] = expected no. of steps to convert n to 0





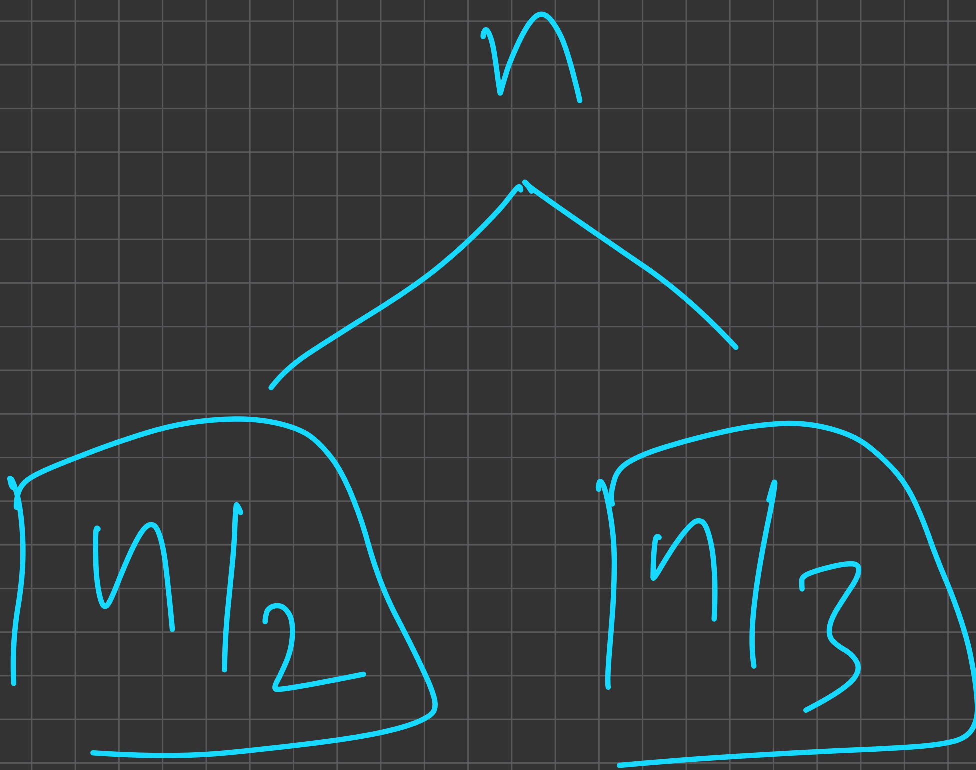
$O = O + o + o + \omega + o$

dp[9]



expectation $= \sum$ possibility $\cdot$ probability

$$dp(n) = \tfrac{1}{3} \cdot dp(n/2) + \tfrac{1}{3} \cdot dp(n-1) + \tfrac{1}{3} \cdot dp(n) + 1$$



$$dp(n) \leftarrow \begin{cases} (1 + dp(n/2)) \\ (1 + dp(n/3)) \end{cases}_{\min}$$

$$
\begin{tikzcd}
 & n \arrow[dl, dash] \arrow[dr, dash] & \\
n/2 & & \boxed{n/3}
\end{tikzcd}
$$

$$dp(n) = \min\left(1 + dp(n/2),\ 1 + dp(n/3)\right)$$

$$n$$

$$\boxed{n/2} \qquad n-1 \qquad n$$

with equal probability

what is the expected no of steps

$$dp(n) = \begin{cases} (1 + dp(n/2))\,\frac{1}{3} \\ (1 + dp(n-1))\,\frac{1}{3} \\ (1 + dp(n))\,\frac{1}{3} \end{cases}$$

$$dp(n) = \left(\left(1 + dp(n/2)\right) \cdot \tfrac{1}{3} + \left(1 + dp(n-1)\right) \cdot \tfrac{1}{3} + \left(1 + dp(n)\right) \cdot \tfrac{1}{3} \right)$$

$$dp(n) = dp(n/2)\cdot 1/3 + dp(n-1)\cdot 1/3 + \underline{\underline{dp(n)}}\cdot 1/3 + \underline{1}$$

$$\underline{\underline{2/3}}\cdot dp(n) = dp(n/2)\cdot 1/3 + dp(n-1)\cdot \underline{1/3} + 1$$

$$dp(n) = \left(dp(n/2) + dp(n-1)\right)/2 + 3/2$$

probability of an event happening = p

then.

expected no. of steps in which it will happen = $1/p$

---

probability to get 5 or 6 when you toss a die

$$\frac{2}{6} = \underline{\underline{1/3}}$$

$$1/3 =$$

Toss a die, what is the expected no. you will get

$$1 \cdot 1/6 + 2 \cdot 1/6 \ldots 6/6$$

$$= \underline{\underline{(1+2+\ldots 6)/6}} = \frac{6 \cdot 7}{2 \cdot 6}$$

$$= \underline{\underline{3.5}}$$

expectation $= \sum$ every possibility $\times$ probability

$$= 2 \cdot 5/6 \text{ tries}$$

$$3 \quad 1/6 \text{ tries}$$

$$2 \cdot 5/6 + 3 \cdot 1/6$$

$$= 13/6 = 2.166$$

Problem 2: Link