Doubt Clearing Session

Course on Game Theory and Greedy Algorithms

Triveni Mahatha • Lesson 4 • May 17, 2021

I'll wait for 10 minutes for doubts. If not, we can solve some more problems.

← carrots

each cut has to be a positive integer and cannot be a fraction.

'K' Rabbits

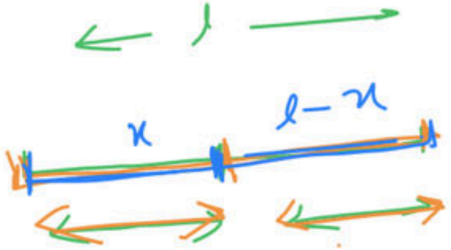
Task is to cut the carrots into 'K' pieces and min. the time taken to eat the carrots.

↑ sum total of time taken by each rabbit.

'x' carrot → x^2 time

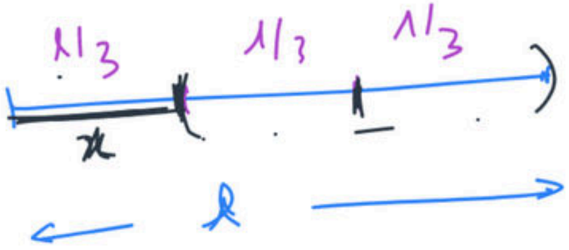
Solution

$\rightarrow \boxed{l^2}$



$\rightarrow \left(\frac{l}{2}\right)^2 + \left(\frac{l}{2}\right)^2 = \boxed{\frac{l^2}{2}} < l^2$

It is always better make more cuts to a carrot.



$$f(x) = x^2 + \left(\frac{l-x}{2}\right)^2 + \left(\frac{l-x}{2}\right)^2$$

$$f'(x) = 2x - 2\left(\frac{l-x}{2}\right) = 0$$

$$2x = l - x$$

$$\boxed{x = l/3}$$

$$z(x) = x^2 + (l-x)^2$$

$$z'(x) = 2x + 2(l-x)\times -1$$

$$= 0$$

$$2x = 2l - 2x$$

$$4x = 2l$$

$$\Rightarrow \boxed{x = l/2}$$

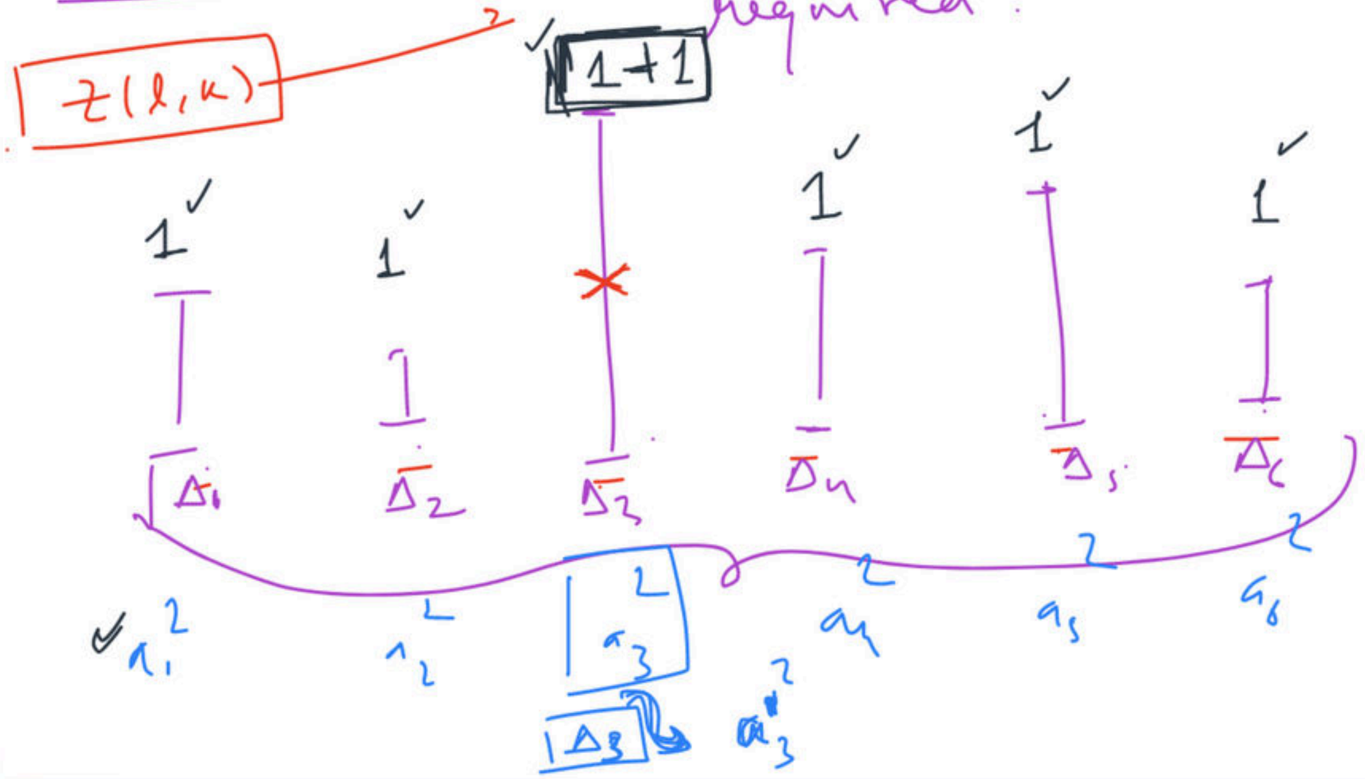
Observation: It is always good to make equal length cuts.

$l = 15$ $k = 4$

$\left\lfloor \frac{l}{k} \right\rfloor$ some parts $\left\lfloor \frac{l}{k} \right\rfloor + 1$ The remaining parts.

carrots (l, k) → optimal time that is required.

$z(l, k)$



6 carrots → 7 carrots.

You can iterate through the carrots and see which carrot gives you the best cut it. You choose to

Repeat the following until you've 'k' pieces in total:

↳ Iterate through each carrot and see which carrot gives you the best- △ one making an additional cut into it.

↳ Note that carrot and update its # cuts.

finally return the answer.

O(n × k) → by using a set instead of normal array. → O(k log n)

△: z(l, p) − z(l, p+1)

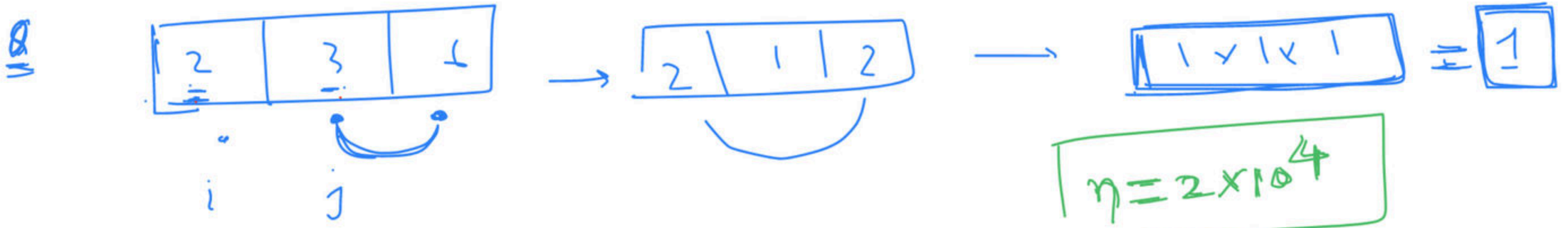
↳ set / Treaps

↓

O(k log n)

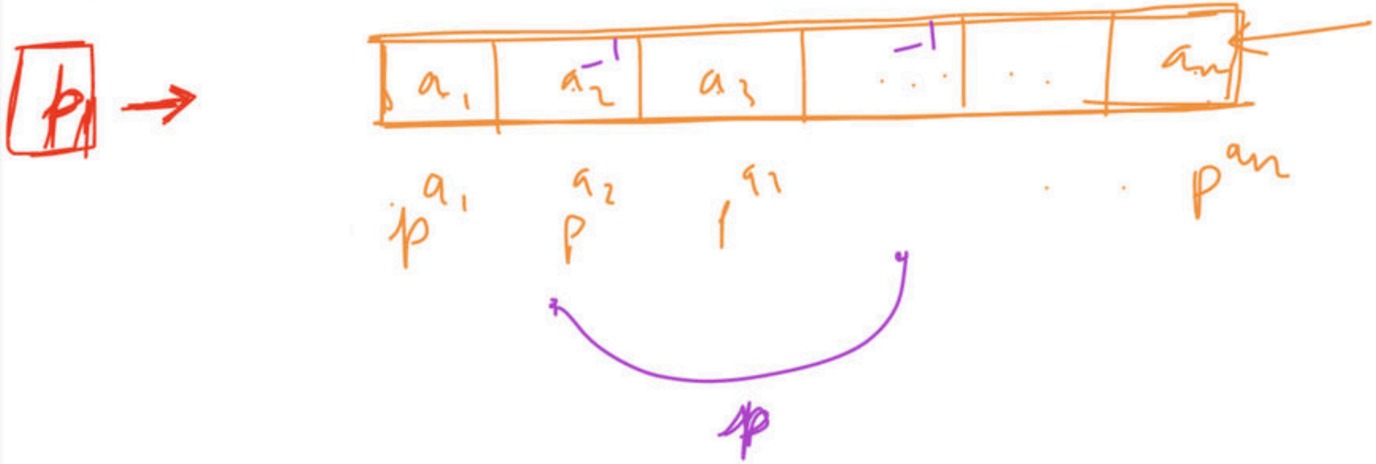
set < pair < int, pair < l, p > > > . myset

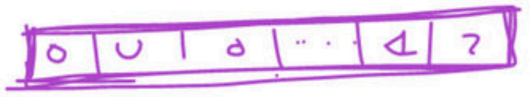
int, int



$n = 2 \times 10^4$

the number which divides a[i] & a[j] can be restricted to a prime number only.

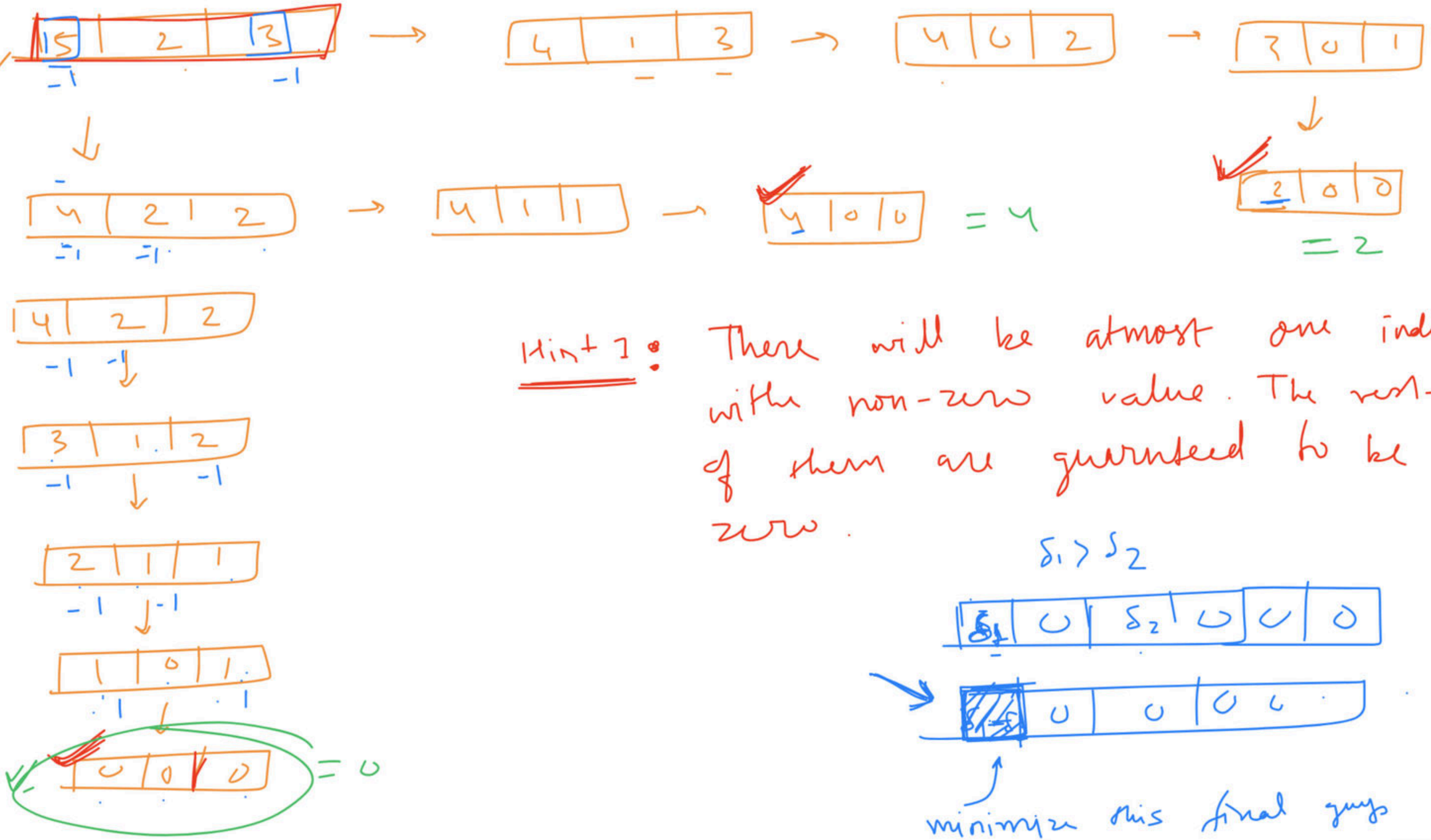


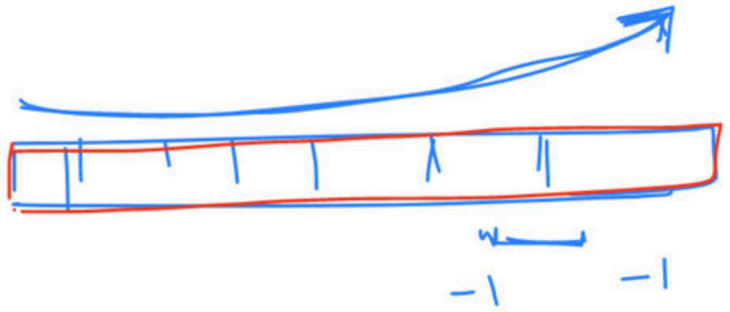
pp: 

minimize the sum total of the values in pp

$a_1 + a_2 + a_3$

$p^{a_1} \times p^{a_2} \times p^{a_3} = p$


Hint 1: There will be atmost one index with non-zero value. The rest of them are guaranteed to be zero.
$\delta_1 > \delta_2$
minimize this final guy



→ keep doing this

Repeat for each prime 'p':

↳ resolve the "pb" array using the idea of reducing the maximum value.

↳ The remaining array gives you the product contributed by this prime ⇒ b

$a[x]$ ← remaining guy.

final answer $= \prod p^{a[p]}$

$2 \le p \le 10^9$

Won't work

T.C. $= O(\pi(10^9) \times n \log n)$

$= \frac{10^9}{\log n} \times 2 \times 10^9 \log n =$ Huge

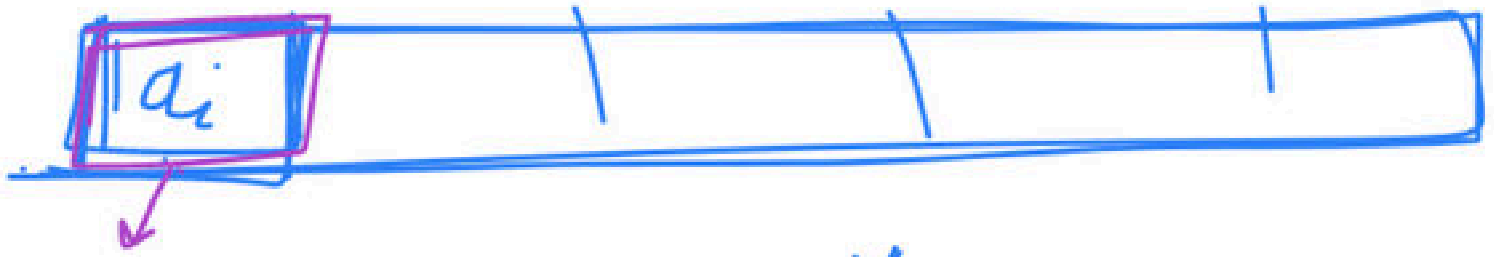
G1 : $2 \le p \le \sqrt{10^9} \Rightarrow O(\bar{\pi}(\sqrt{10^9})\, n \log n) = \frac{4 \times 10^4}{\log n} \times 2 \times 10^4 \times \log n$

$= 8 \times 10^8 \rightarrow$ 2 sec

G2 : $\sqrt{10^9} < p \le 10^9$

We need some other strategy for the larger primes.

prime factorize all the "a_i's"



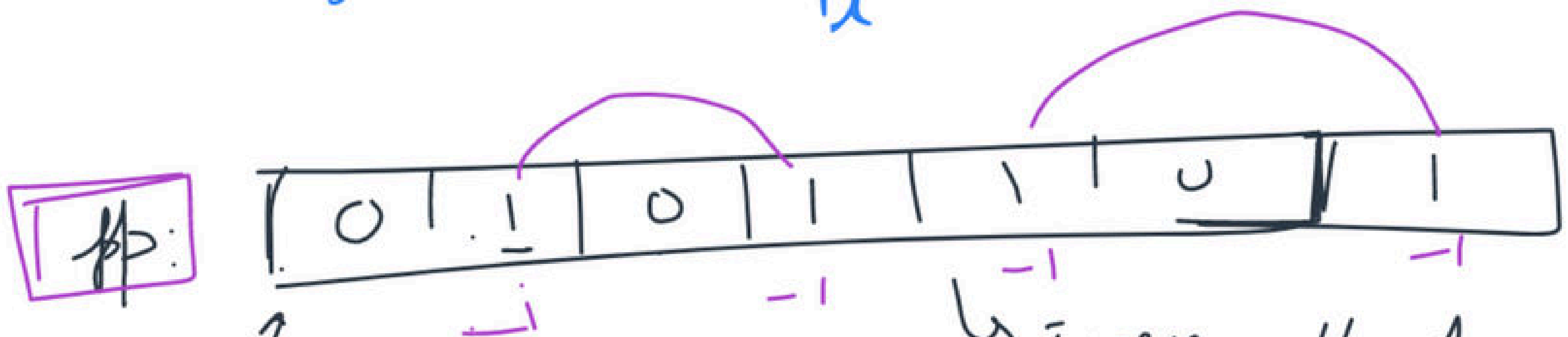
$a_i = x \times p_l^{1/0}$

$a_i \le 10^9$

$a_i = p_l^{x} < 10^9$

$x < \log_{\sqrt{10^9}} 10^9$

$x < 2$

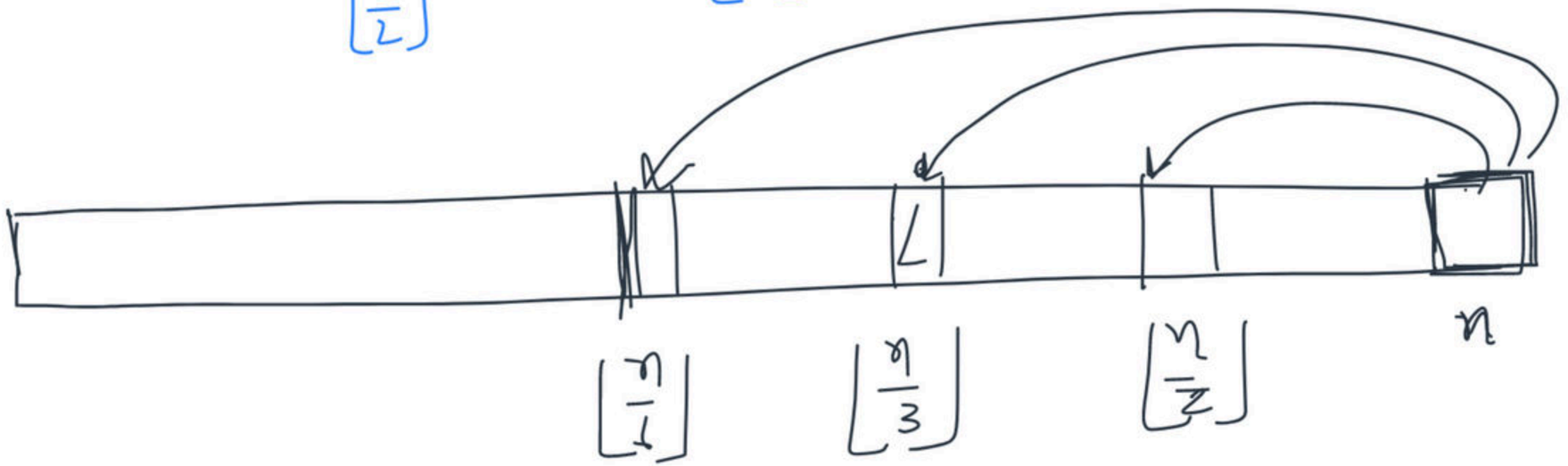


for large prime

↳ Even # of ones then the 'pp' array can be reduced to 0 otherwise just '1' non zero element remains.

Problem: 'n' ← $\lfloor n/2 \rfloor$, $\lfloor n/3 \rfloor$, $\lfloor n/6 \rfloor$ if these values become '0' then it is removed. The player who made the last move wins.

6 $\xrightarrow[\lfloor 6/2 \rfloor]{A}$ 3 $\xrightarrow[\lfloor 3/6 \rfloor]{B}$ 0 'B' won.



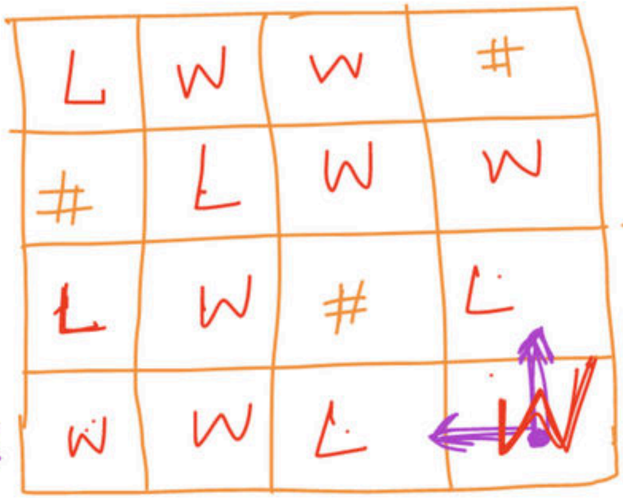
if any of this is losing then 'n' is winning o/w 'n' is losing.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	W	W	W	W	W	L	L	L	L	L	L	W	W	W	W

✓ Time Complexity = O(n)

Question

1) Players take turns.
2) You can move left or up.
3) When you cannot move then you lose.
4) You can move as far as possible without getting blocked in the specified direction.
   - Rook / Elephant in a game of Chess



n x n grid

Who'll win this game??

⇒ Both the players play optimally.

Time Complexity: $n \times n \times (n+n) = O(n^3)$

$O(n^2)$ The optimal solution

$dp_1(r,c)$: If there is any losing position above this cell

$dp_2(r,c)$: If there is any losing position to the left of it

$ans[r,c] = (dp_1(r-1,c) \mid dp_2(r,c-1)) \; ? \; 'W' : 'L'$.

$dp_1(r,c) = (dp_1(r-1,c) \mid ans[r,c] == 'L')$

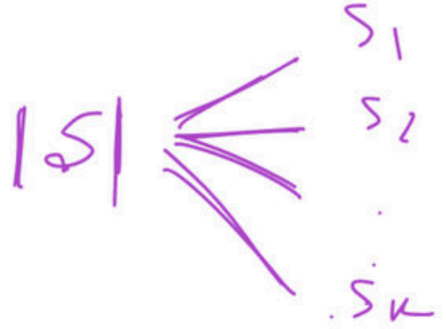
Revision

↳ Combinatorial Game ??

- → two players take alternate turns.
- → very intelligent
- → Can do huge amount of calculation per second.
- → Both of them try to win and play optimally.

The outcome depends

- who plays first
- what is the state of the game.



If any of S_i is losing then 'S' is winning.
Otherwise the 'S' is losing.

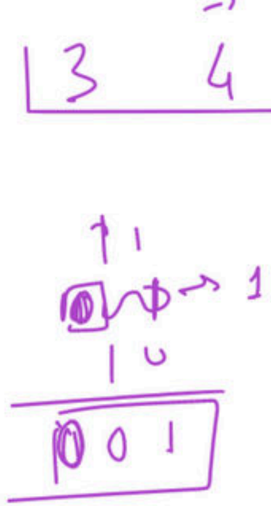
Nim Game

Optimal strategy: to keep the xor-sum equal to zero after each move if you want to win.

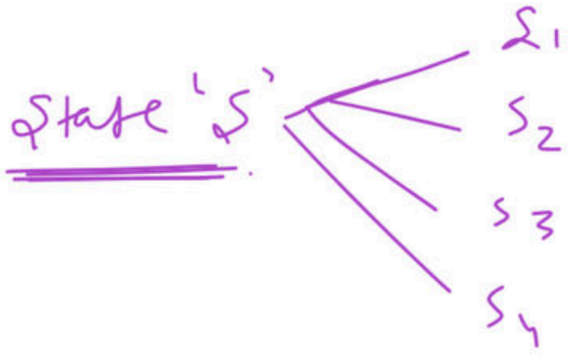
if xor-sum $\neq 0$ → winning state

if xor-sum $= 0$ → losing state

3	4	2



Greedy Algorithms



Obvious way: to consider all possibilities

⇓

Dynamic Prog.

Greedily discard all possibilities except any particular 'one' — Greedy Algo.

↳ You need solid argument - that why it is going to give you the correct answer.

1) Coming up with a solution → Solve small cases and observe pattern.

2) Proof Hardest part
   - Greedy stays ahead.
   - Exchange Argument.

Thank You Very Much