Hard

K^{th} Smallest fraction

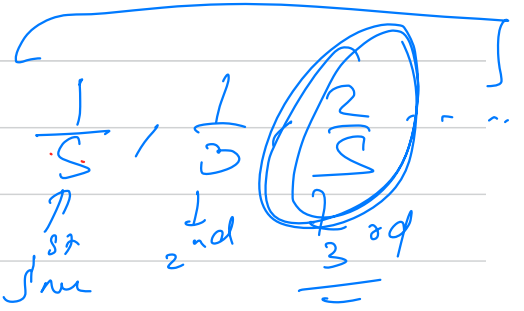
Q) You have a sorted list of prime numbers including 1. For every $p < q$ in the list, we define fraction of primes as p/q.

Find the K^{th} smallest fraction

↳ [1, 2, 3, 5]

↳ 2/5

$K = 3$



Constraints ⇒

↳ length of array $\leq 10^5$

$O(n^2)$

K^{th} order statistics

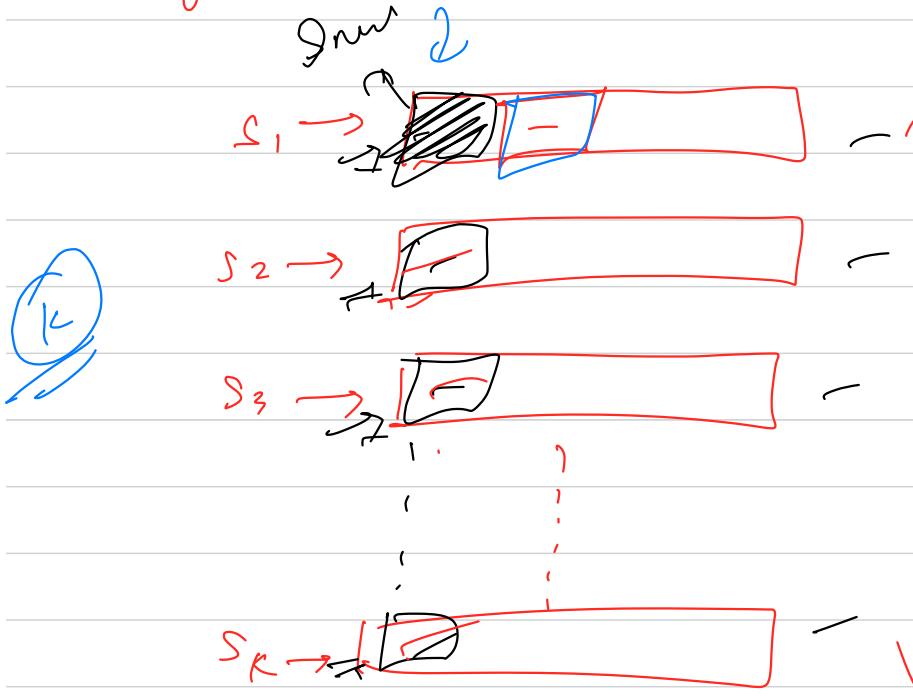
DP
Run
scan

- unsorted array, find K^{th} largest element
- K^{th} smallest fraction

K^{th} code

→ Quicksort, Map / greedy / code, PBDS, tournament method etc

Merge Sort → Subroutine → merge 2 sorted arrays



smaller of 2nd small

Easy

K sorted arrays

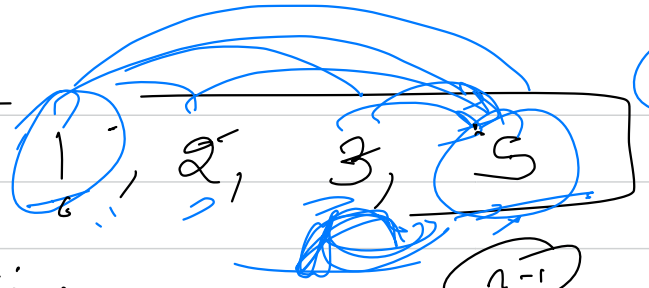
merge

Sprinkles

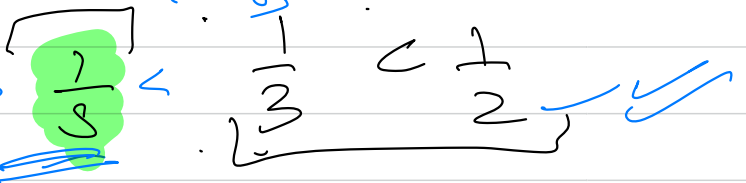
Heaps

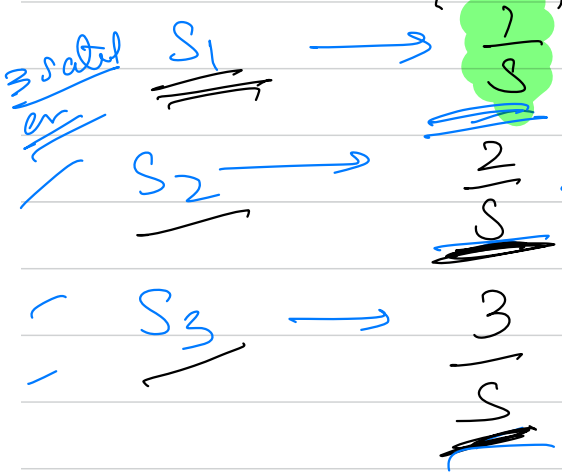
Space complexity

$O(n)$

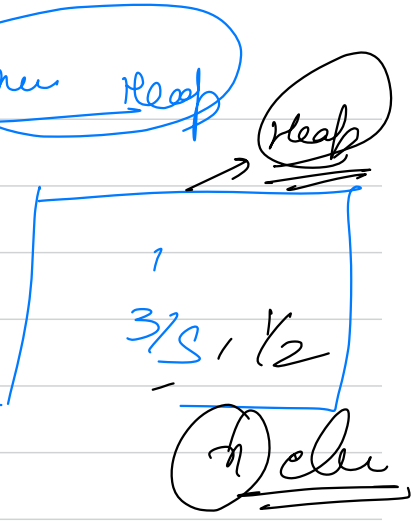


possible fractions





$O(n \log n)$



3rd smallest → 9/S

1st smallest → smallest

2nd smallest → 1/3

Hard

There are N workers, The i-th worker has quality q[i], & min wage expectation w[i];

We want to hire k workers. When hire a worker we must pay according to the following rules-

a) Every worker should be paid in the ratio of their quality compared to others.

b) Every worker should get atleast minimum expected wage

return minimum money required to hire k workers.

$K = 2$

$q \rightarrow (10, 20, 5)$

$w \rightarrow (70, 50, 30)$

ans → 105.00000

$70 + 30 = 100$

worker 0 → 70

worker 2 → 35

$K \le N \le 10^4$

$q[i], w[i] \le 10^7$

$\frac{q_1}{q_2} = \frac{10}{5} = \frac{2}{1} = \frac{w_1}{w_2}$

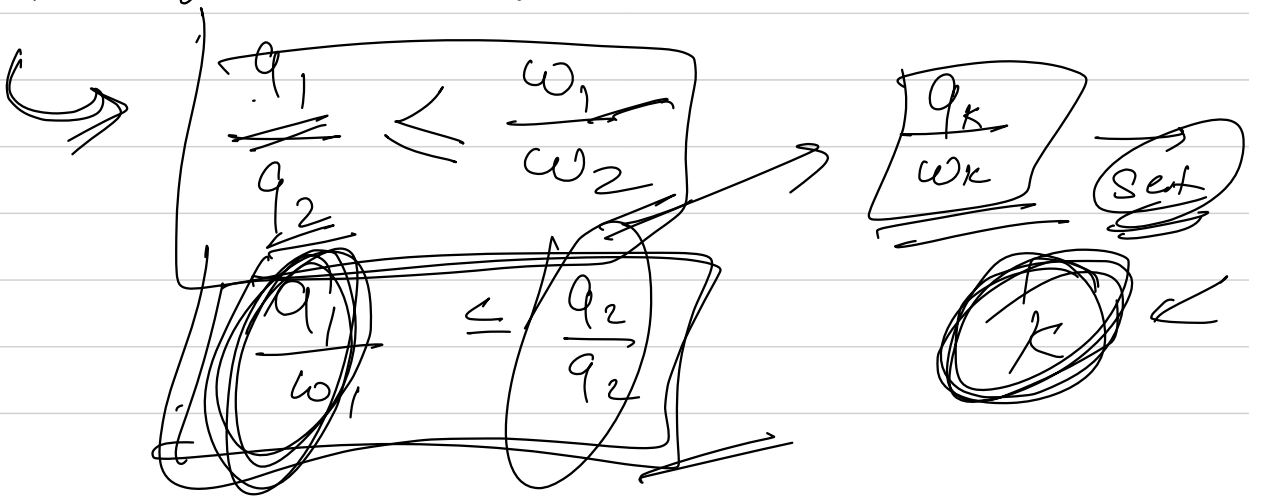
$\frac{w_1}{w_2} = \frac{q_1}{q_2} \Rightarrow \frac{10}{5} = \frac{w_1}{w_2}$

$\frac{70}{35}$

→ Minimum price → we will hire k workers

→ hire based on quality ratio

↳ quality ratio defines wages



Knockout

$\frac{q_k}{w_k}$ → lesser the ratio more the weight

w_k → lesser the ratio lesser the weight

$\frac{w_k}{q_k}$

Sum of quality

Q

Sort the workers based on $\frac{w_k}{q_k}$

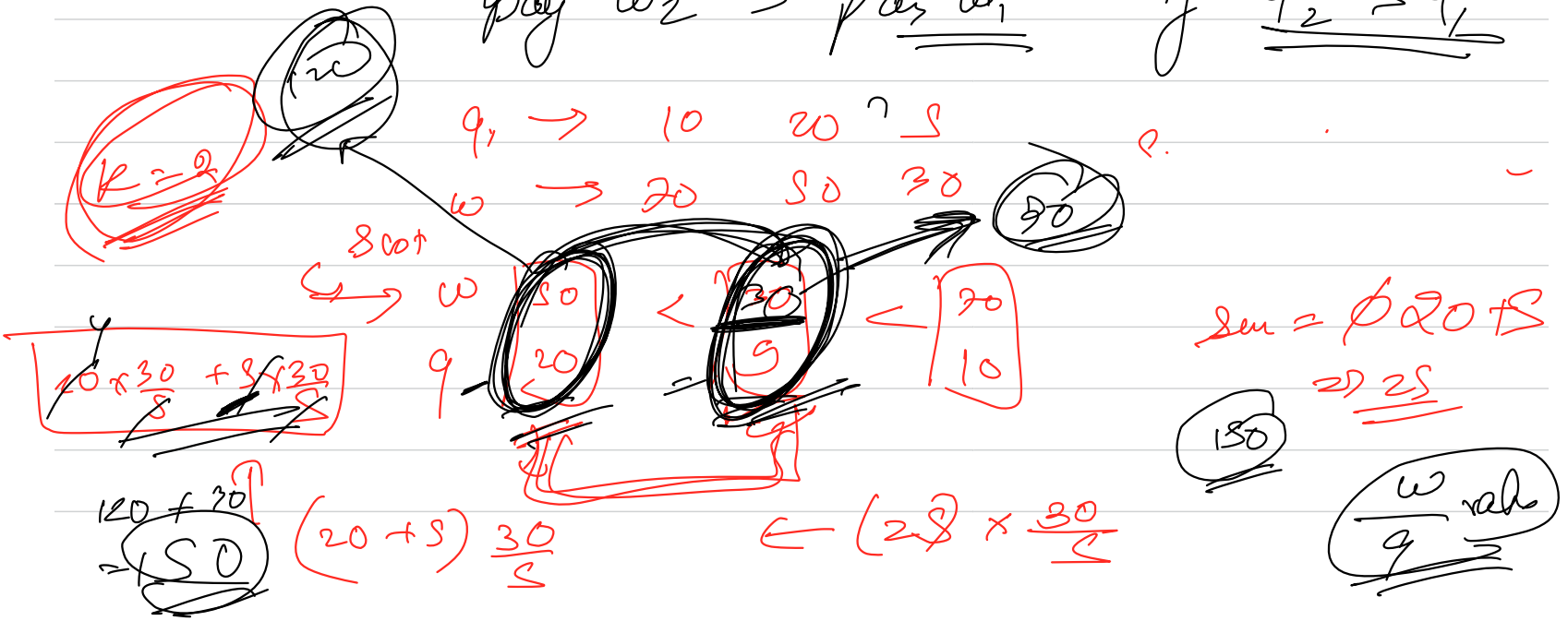
$\frac{w_1}{q_1} < \frac{w_2}{q_2} < \frac{w_3}{q_3} \quad \cdots \quad \frac{w_n}{q_n}$

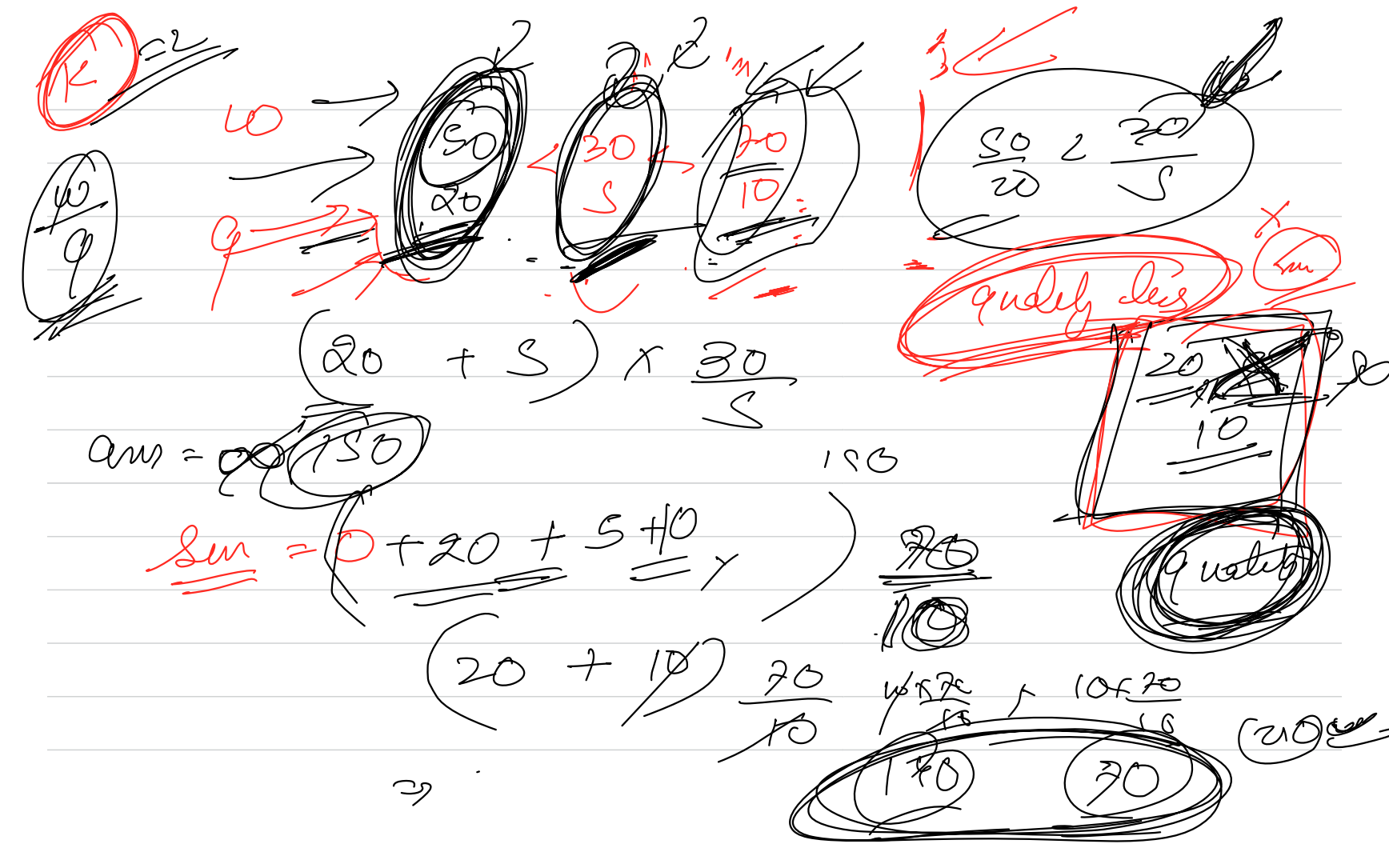
we fixed w_2 q_2

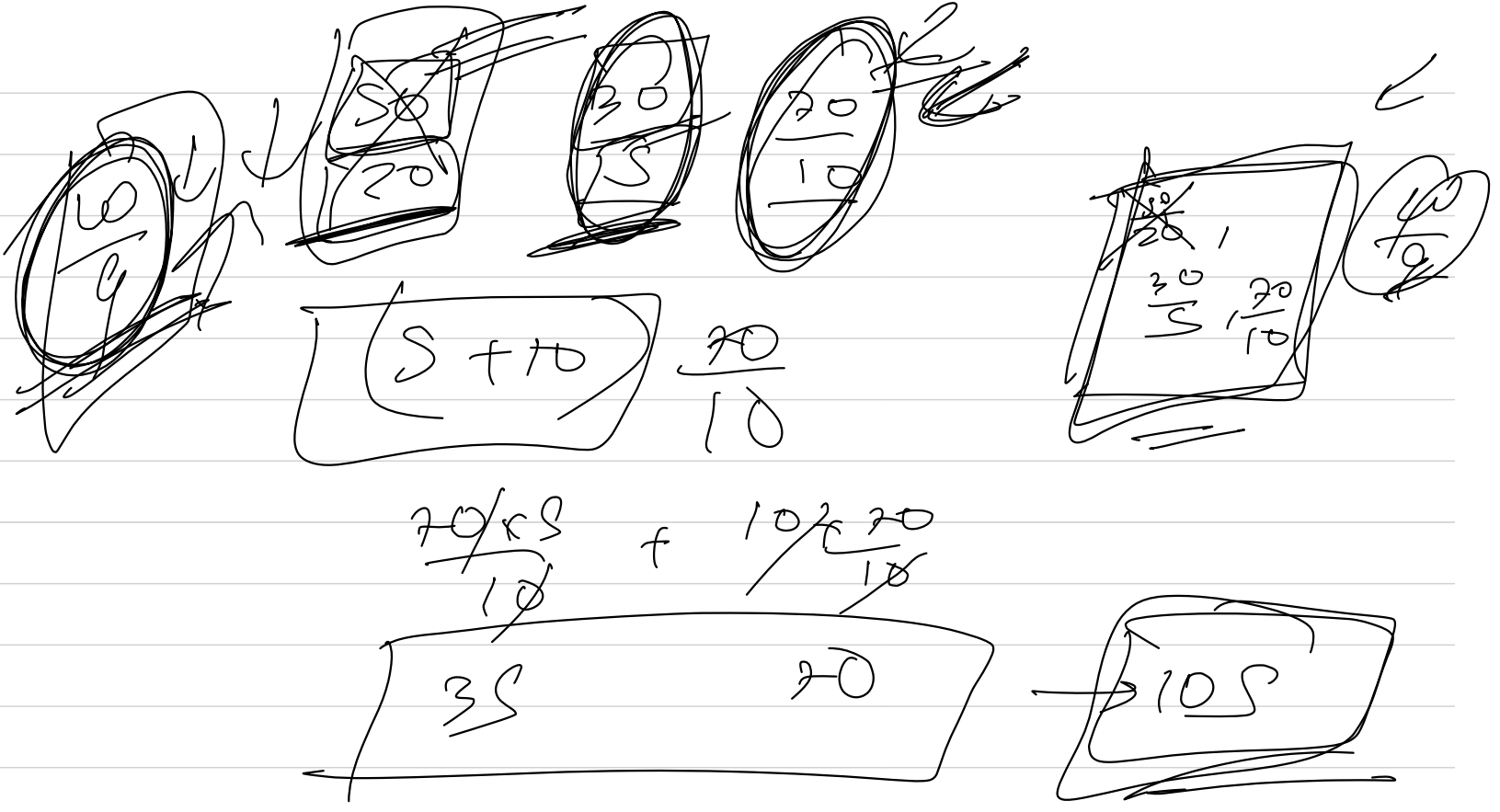
Q

$\frac{w_1}{q_1}$, $\frac{w_2}{q_2}$

pay w_2 > pay w_1 if $q_2 > q_1$




(20 + 5) × 30/5
ans = 150
Sum = 0 + 20 + 5 + 10 = 150
(20 + 10) 20/10
⇒

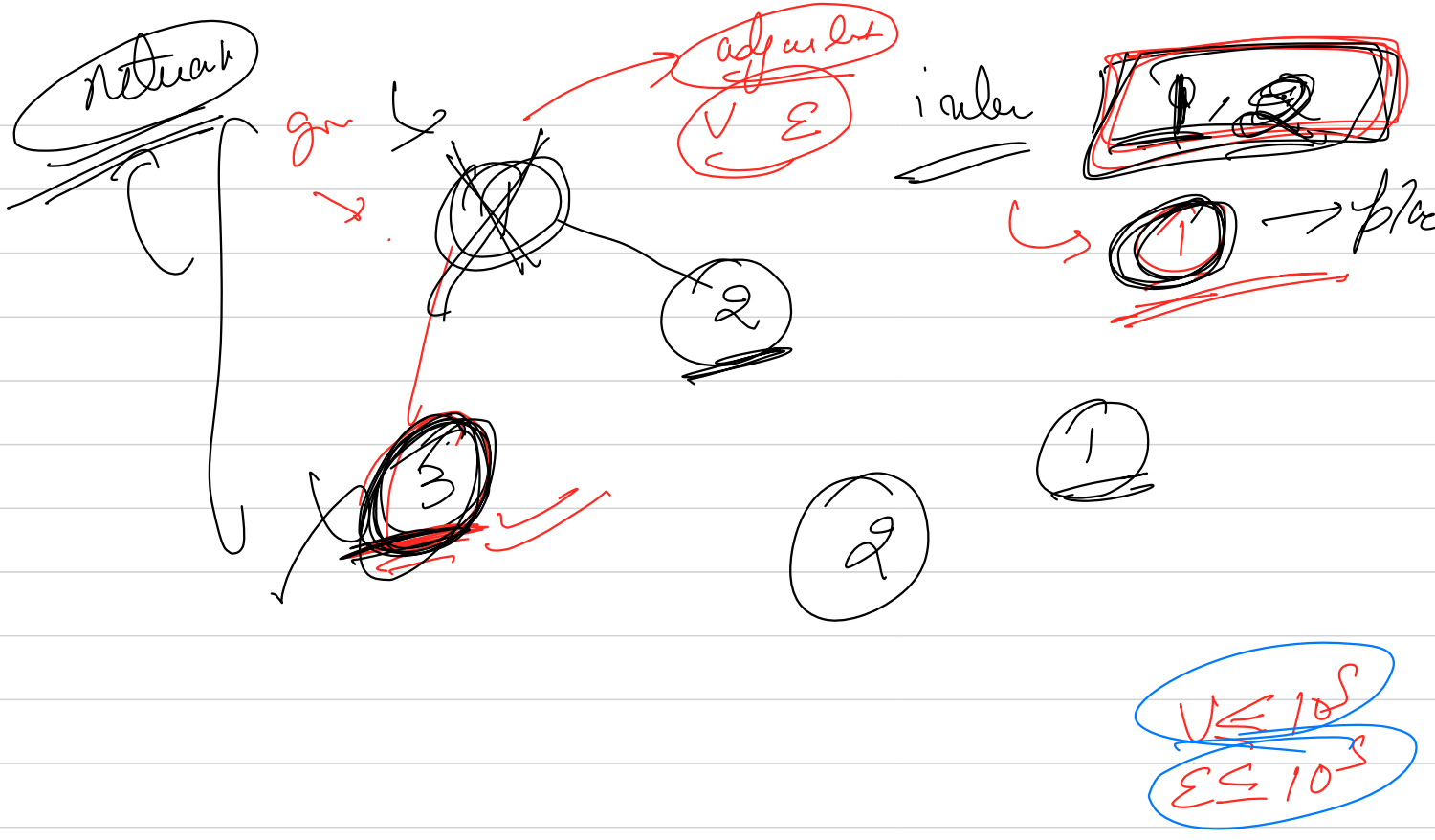

10
50
20
20
5
20
10
5+10
20
10
30
20
5
20
10
20×5
10
+
10×20
10
35
20
105

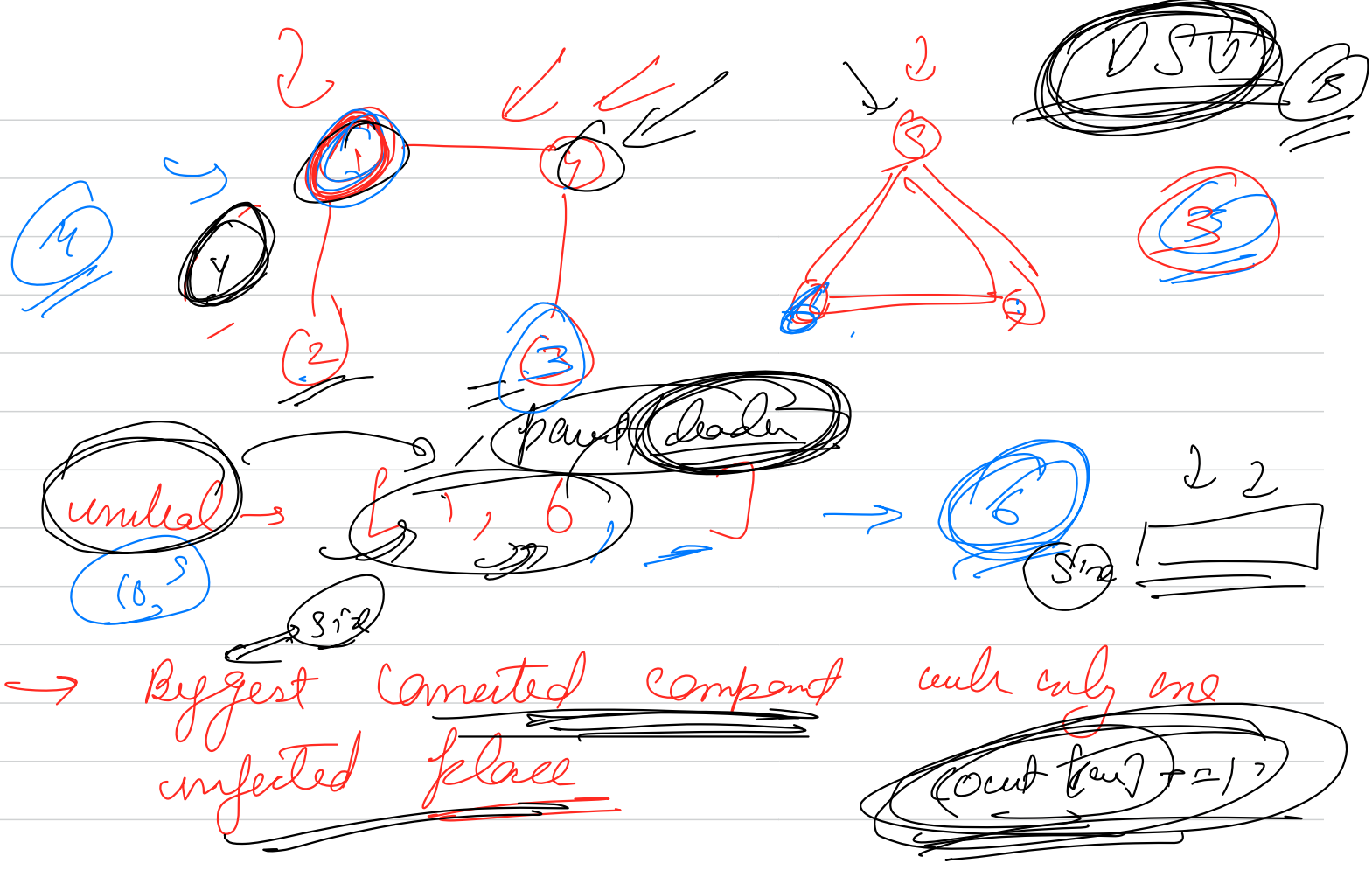
Q2

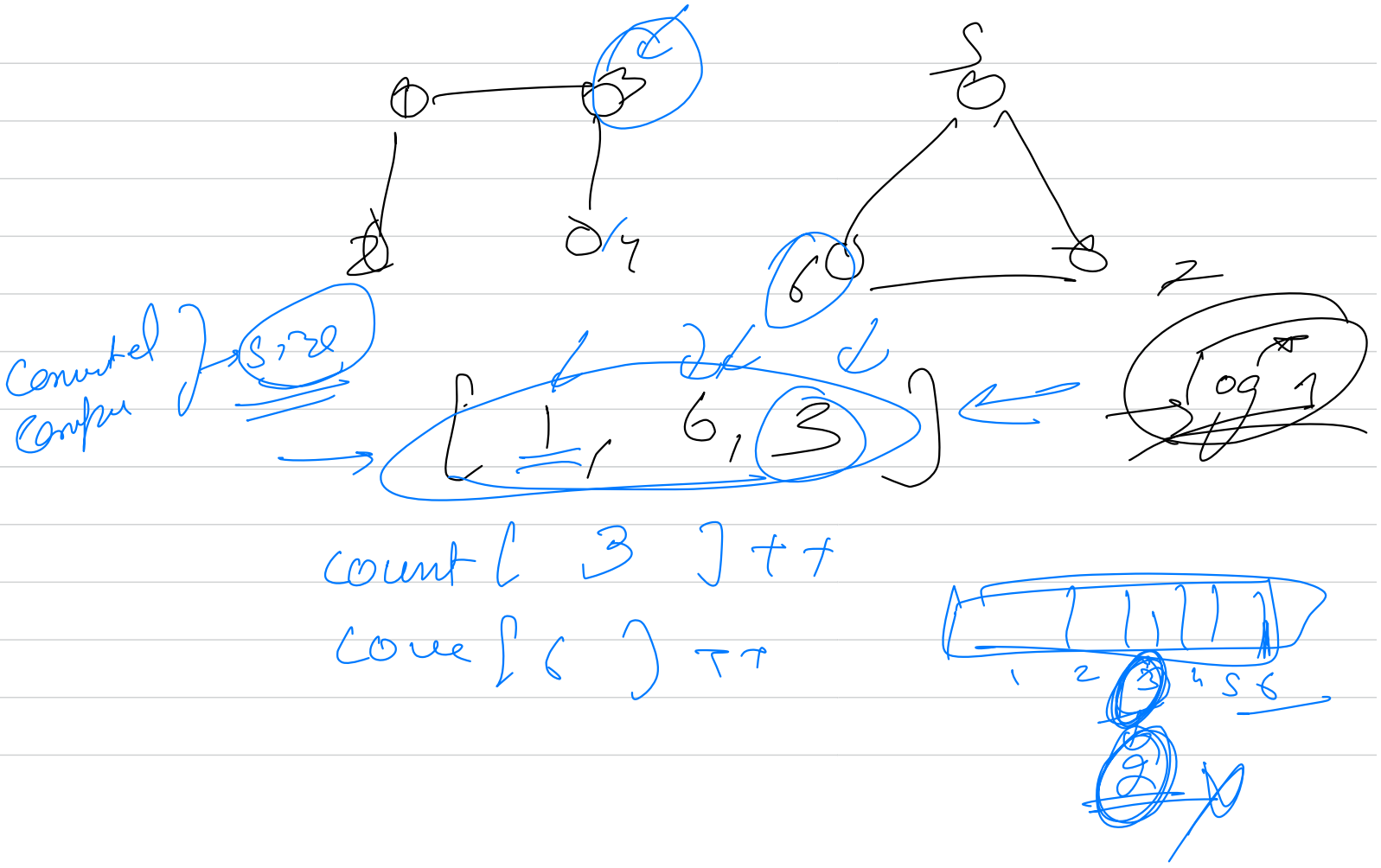
You have some places connected to each other. The virus infection of COVID-19 is spreading in these places.

Some places got infected initially. For 2 places which are directly connected & atleast one of them is infected then other will also get infected.

If we remove any one place from the initial infected list. Return the place that if removed will minimize the infection spread.

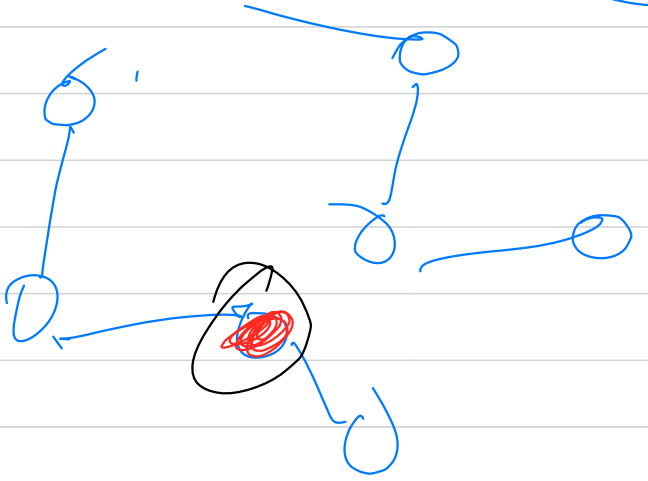

network
graph
adjacent
V E
index
1
place
2
3
1
2
V ≤ 10^5
E ≤ 10^5

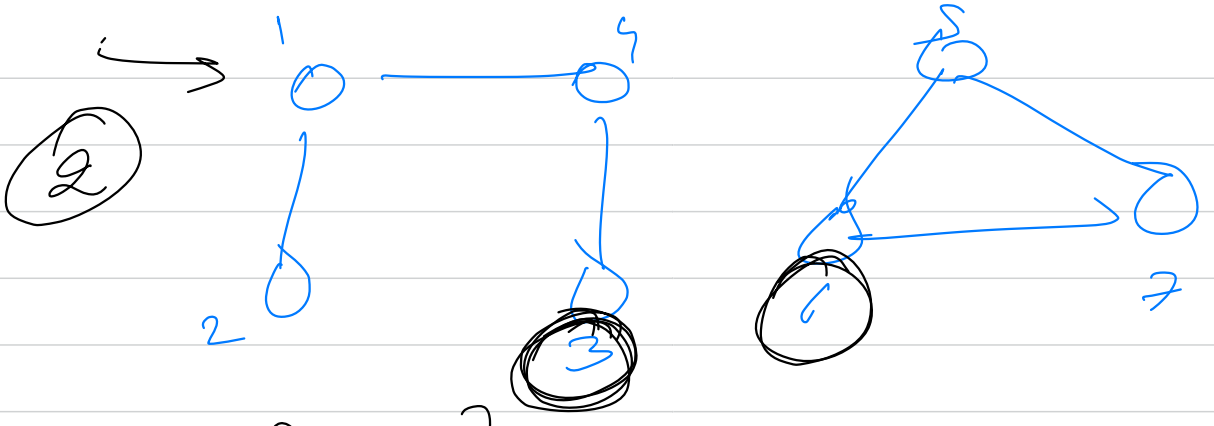

DSU
unite
size
Biggest created component with only one infected place
count[u] += 1;

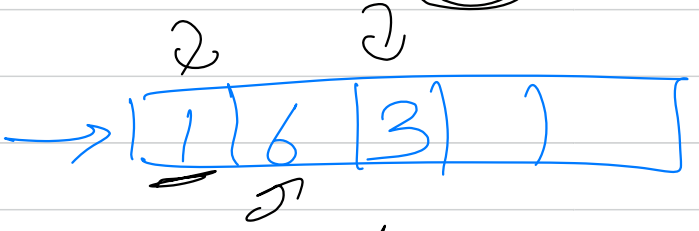

count[3]++
count[6]++
1 2 3 4 5 6

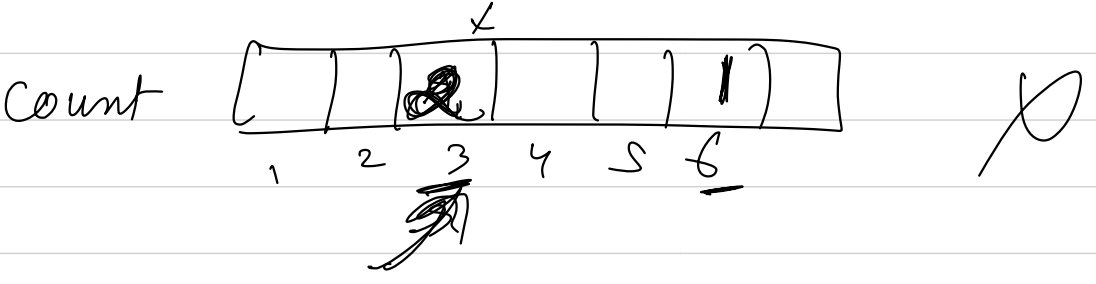
Amazon / Doordash

answer → any one file




1
4
2
2
3
5
1
7


2
2
1
6
3


count
1
2
3
4
5
6

fraction object {

fraction double value;

numerator ida

denominal ida

}

→ {

value → 1/3

n-ida → 0

d-ida → 3

}