USA  Tutor

Farmer John Solves 3Sum

By David Yang

Statement Summary

We have N<=5000 numbers and Q<=1e5 queries

A query consists of 2 indicies as the bound

Answer this query:

Find how many triples of indicies in the range (a,b) have a sum 0.

Brute Force

We can create a new subarray for each query and do this for 0 credit

Or we could answer each query in N time

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

Bad 3 Pointers

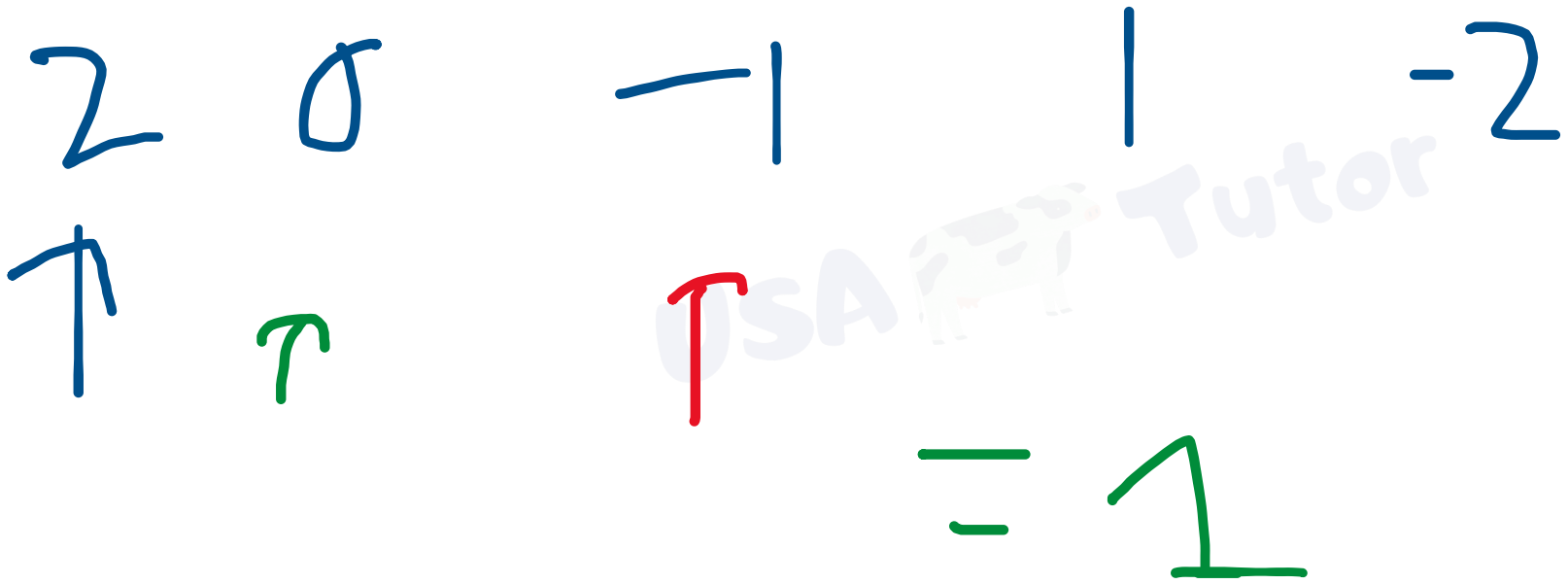
Why was this wrong

We want to fix a range (a,b)

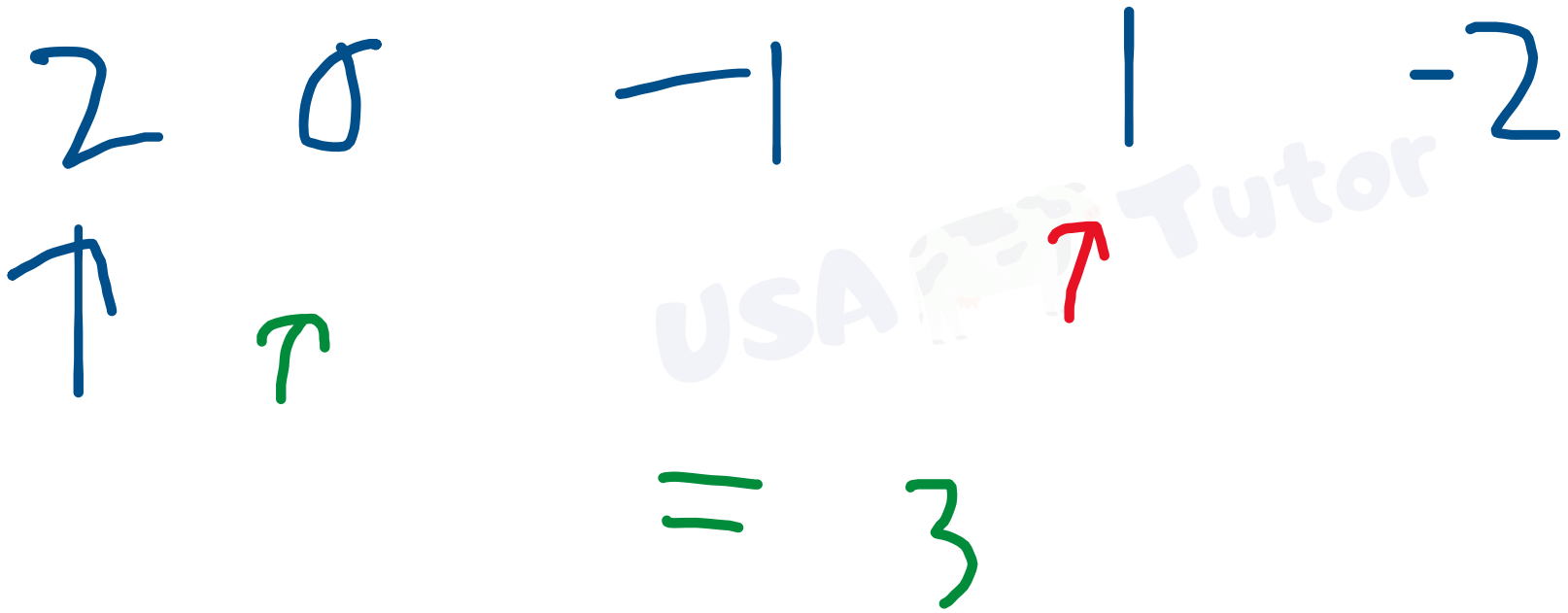
Then we sweep in between

Query 1

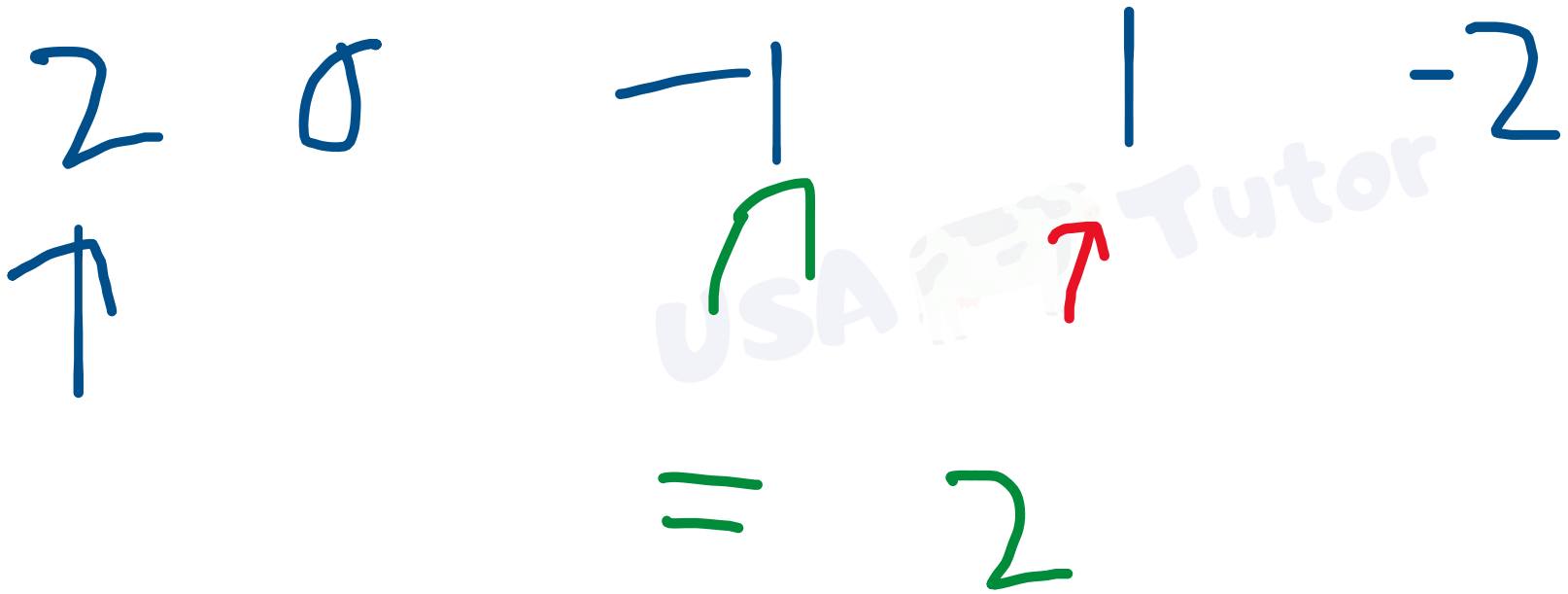
Query 1



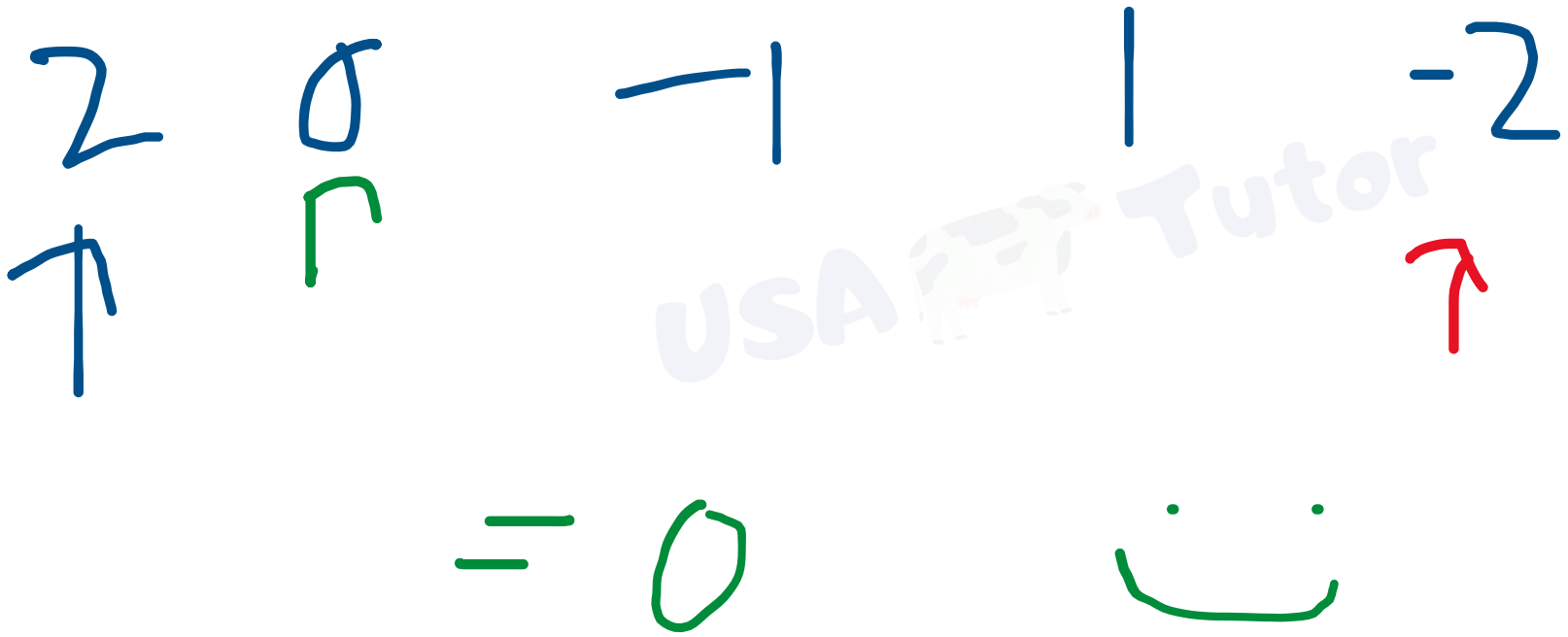
Query 1



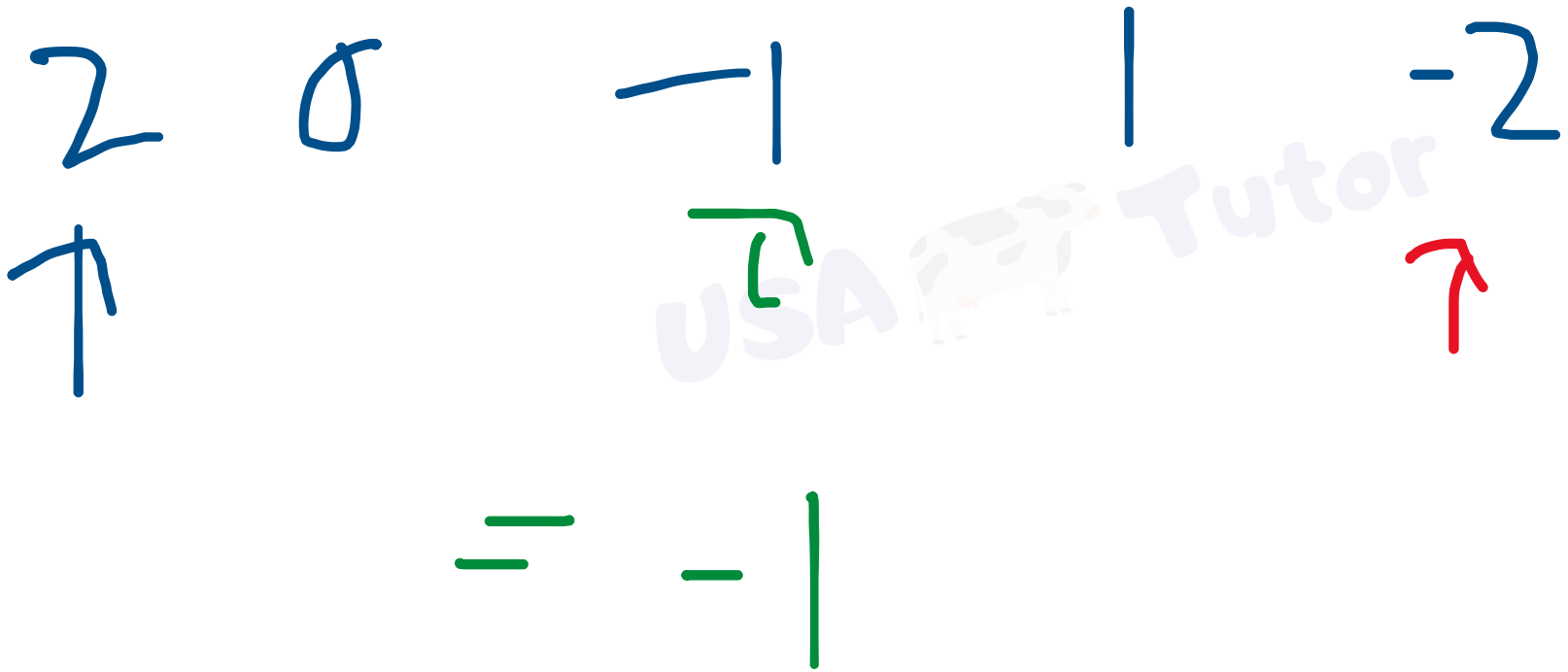
Query 1



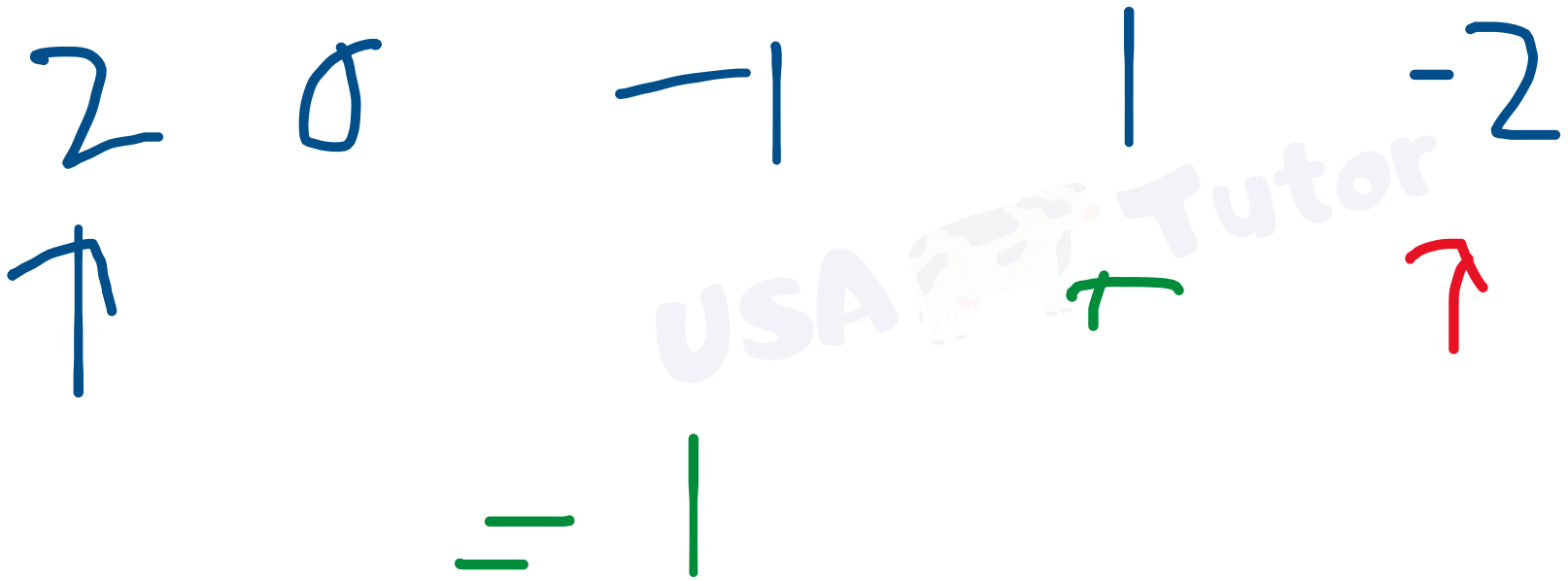
Query 1



Query 1

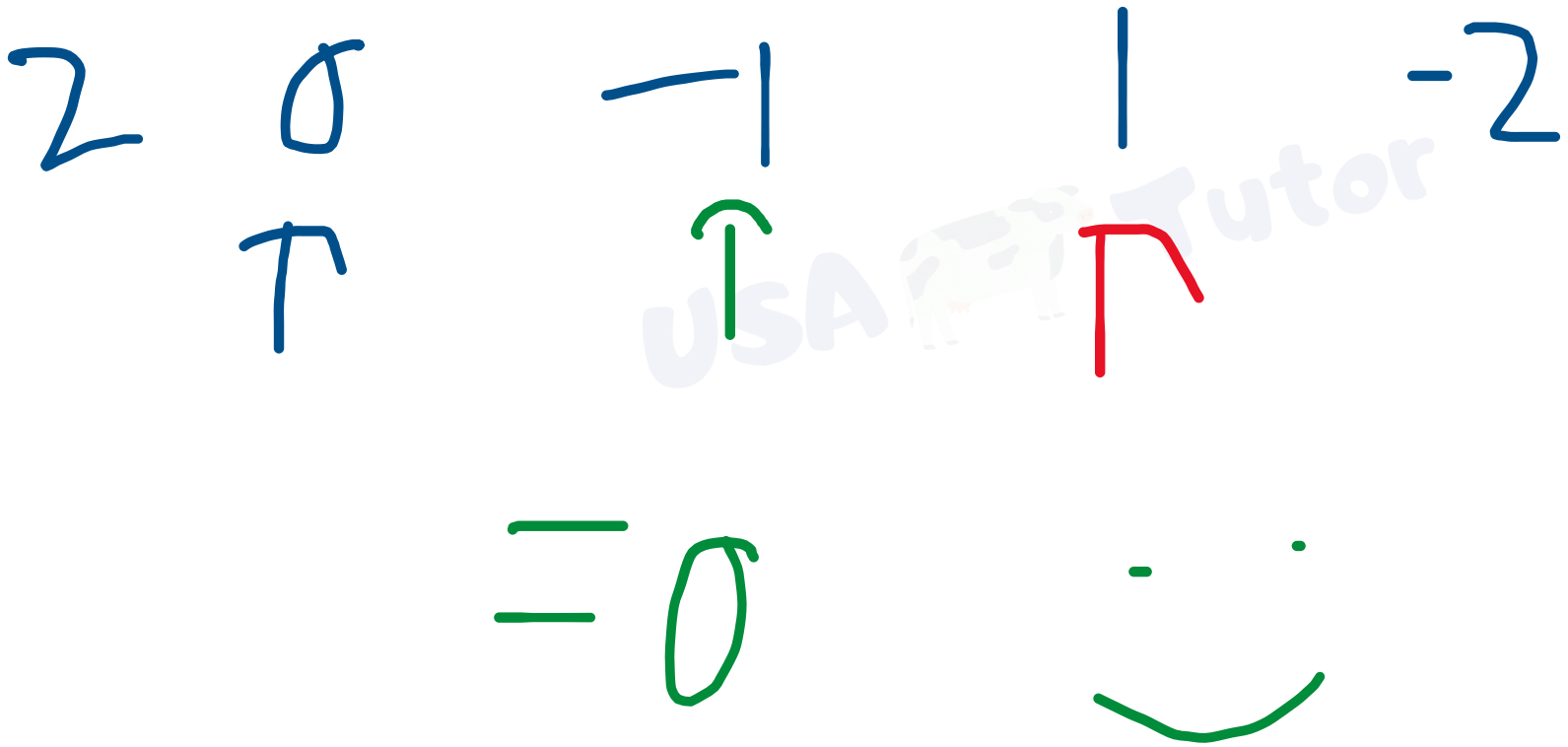


Query 1

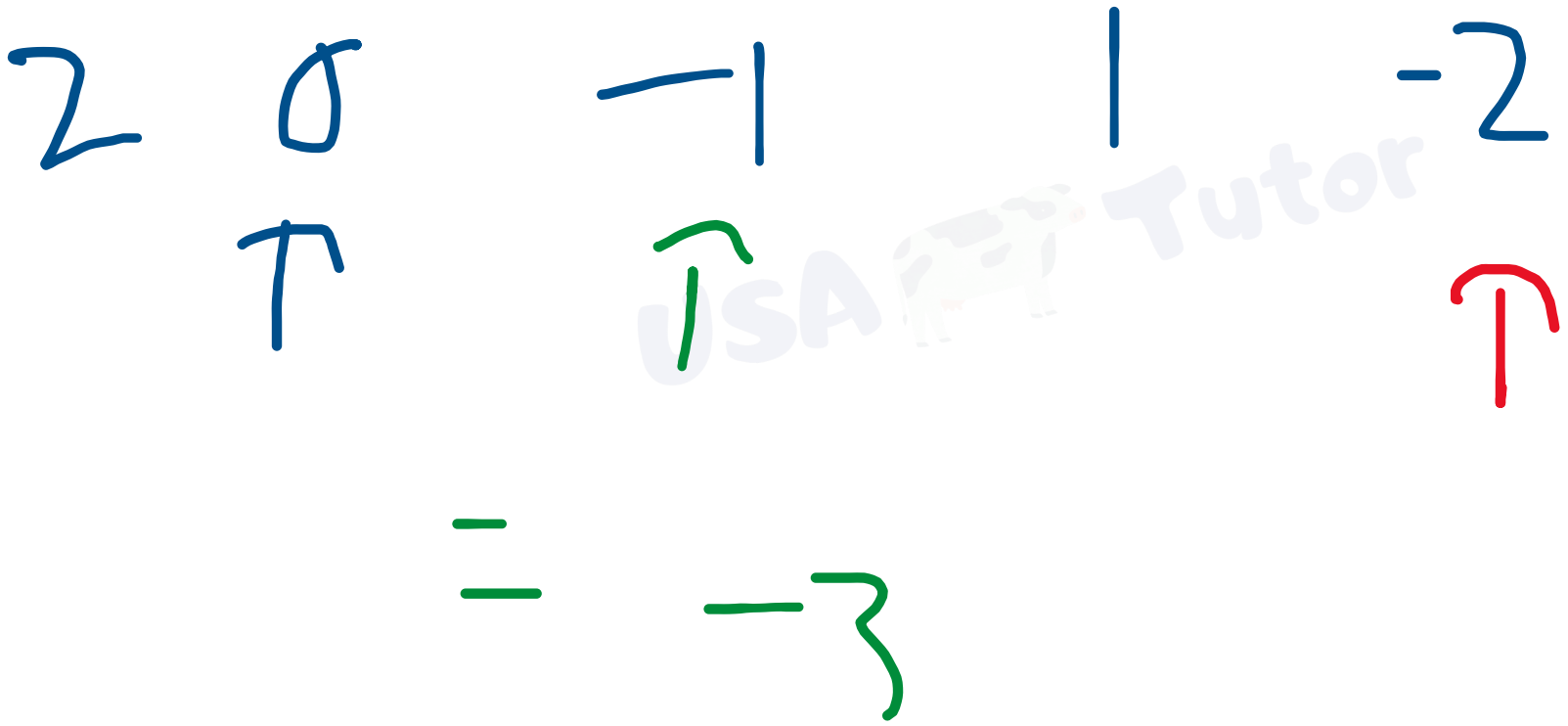


Query 1

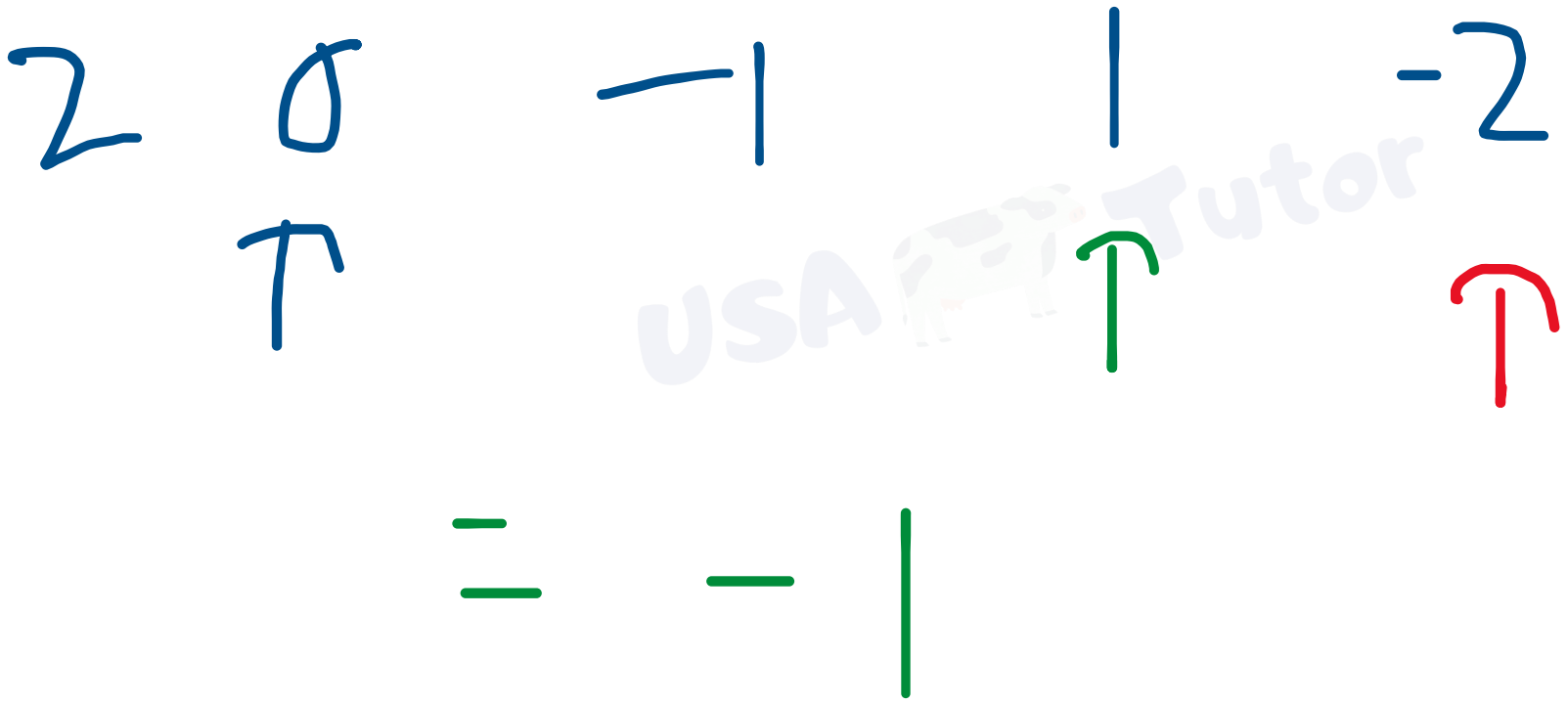
Query 1



Query 1

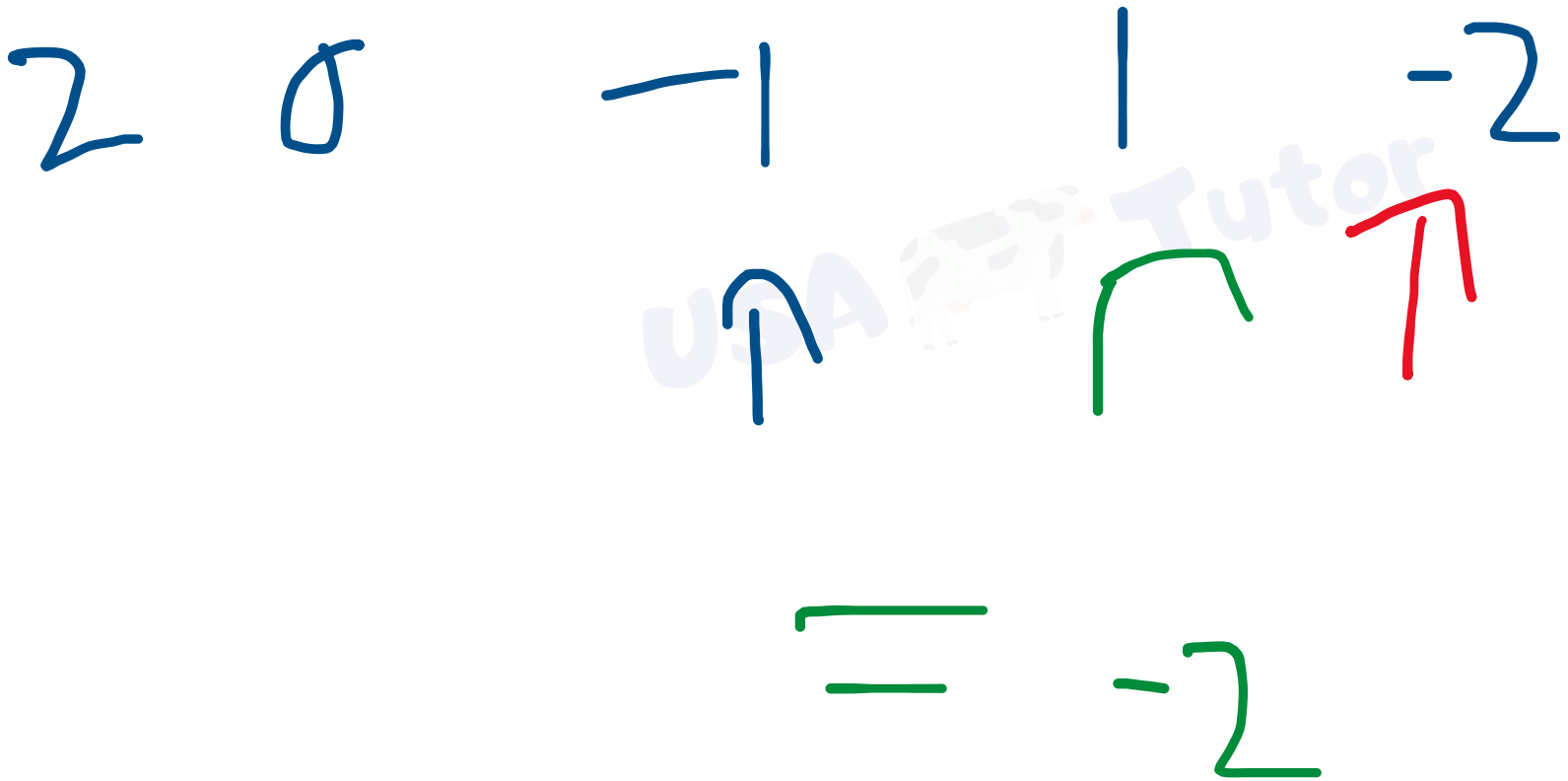


Query 1



Query 1

Query 1



How can we save time?

Previous Solution works in Q * N^3 time

That is a lot

The range (1,5) also includes (2,5) and (3,5) and (4,5)

The range (1,5) also includes (1,4) and (1,3) and (1,2)

Prefix summing for (1,4) to (1,5)

You can just say prefix[i][j] += prefix[i][j-1]

Very simple

Kicking out the Q factor

Let answer[a][b] be the answer to your query, a b

Then you can answer queries in O(1) time

Kicking out the Last N^3 to N^2

For sweeping in between, you can store a map of numbers

Then, you can query if a number exists in your map, then add the amount of times that it showed up before

Kicking out the Last N^3 to N^2

If you are at i=0 and j= 5

you came from i=0 and j=4

This means you can store all of the numbers that you saw from i=0 j=4, and add in j=5 into your map

Map stuff

We reset our map at every new i (starting point)

We do not add in the starting point into our map, or the ending point

And you query for –(arr[i] + arr[j]) in your map