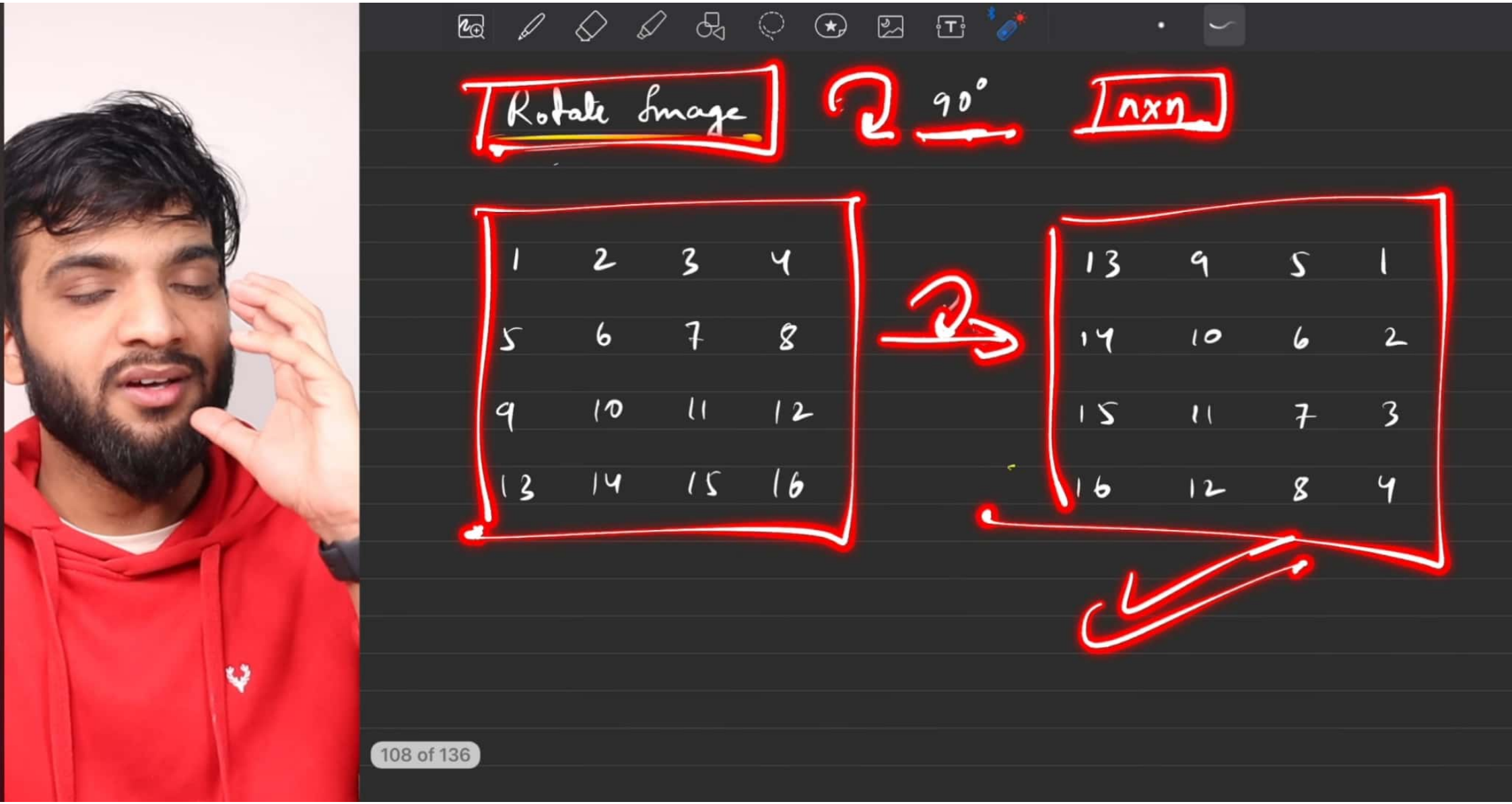
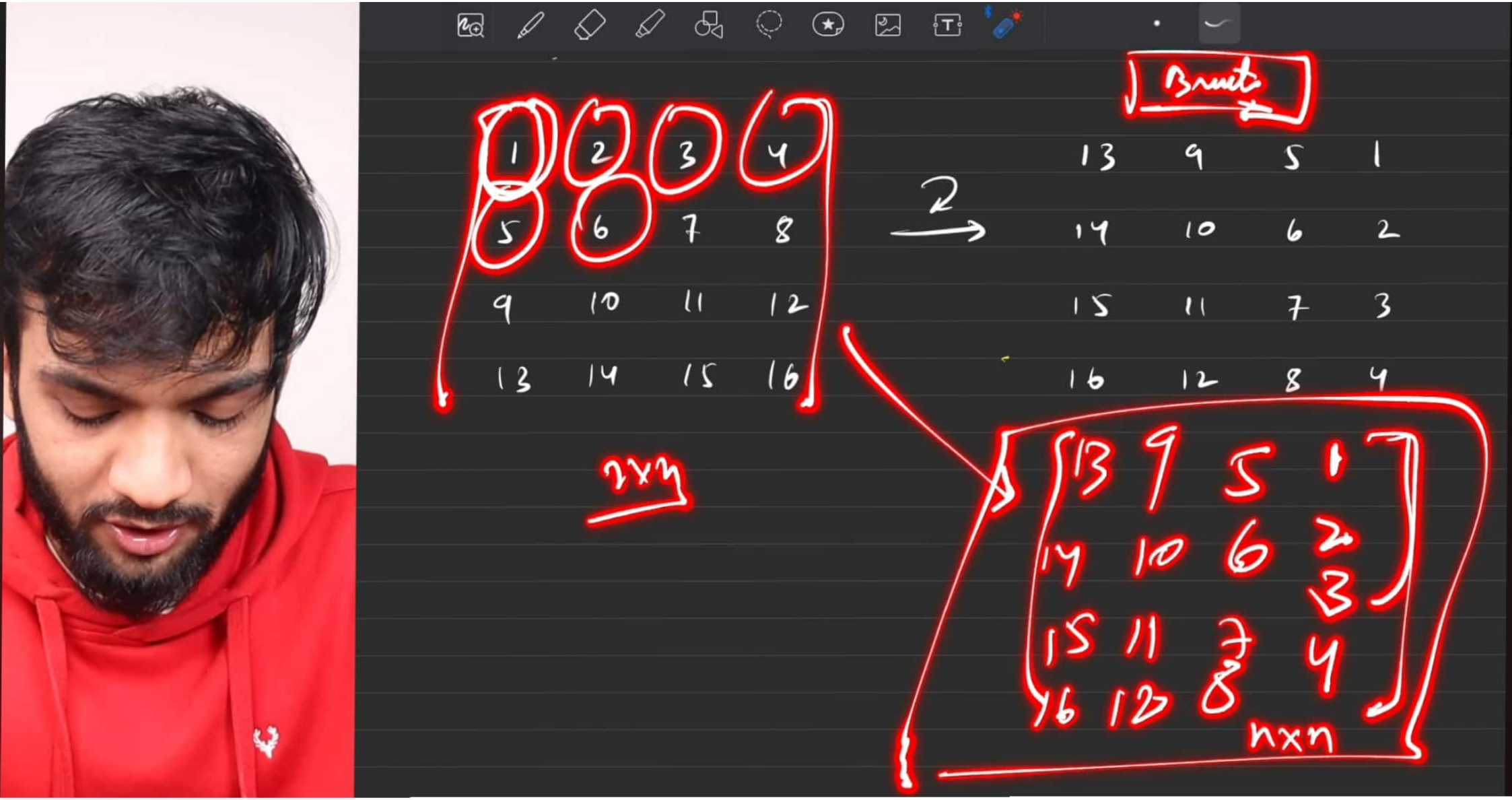
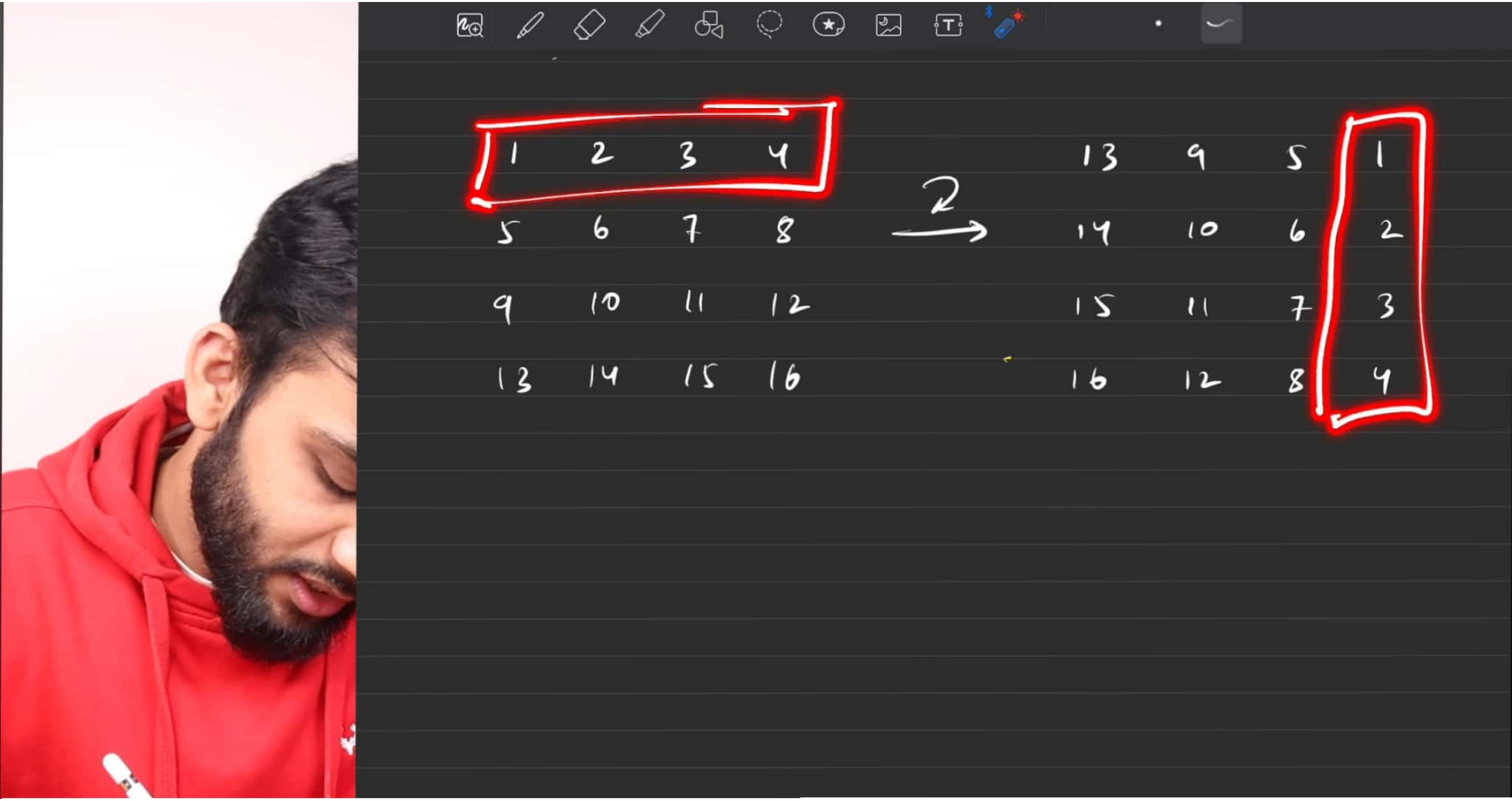
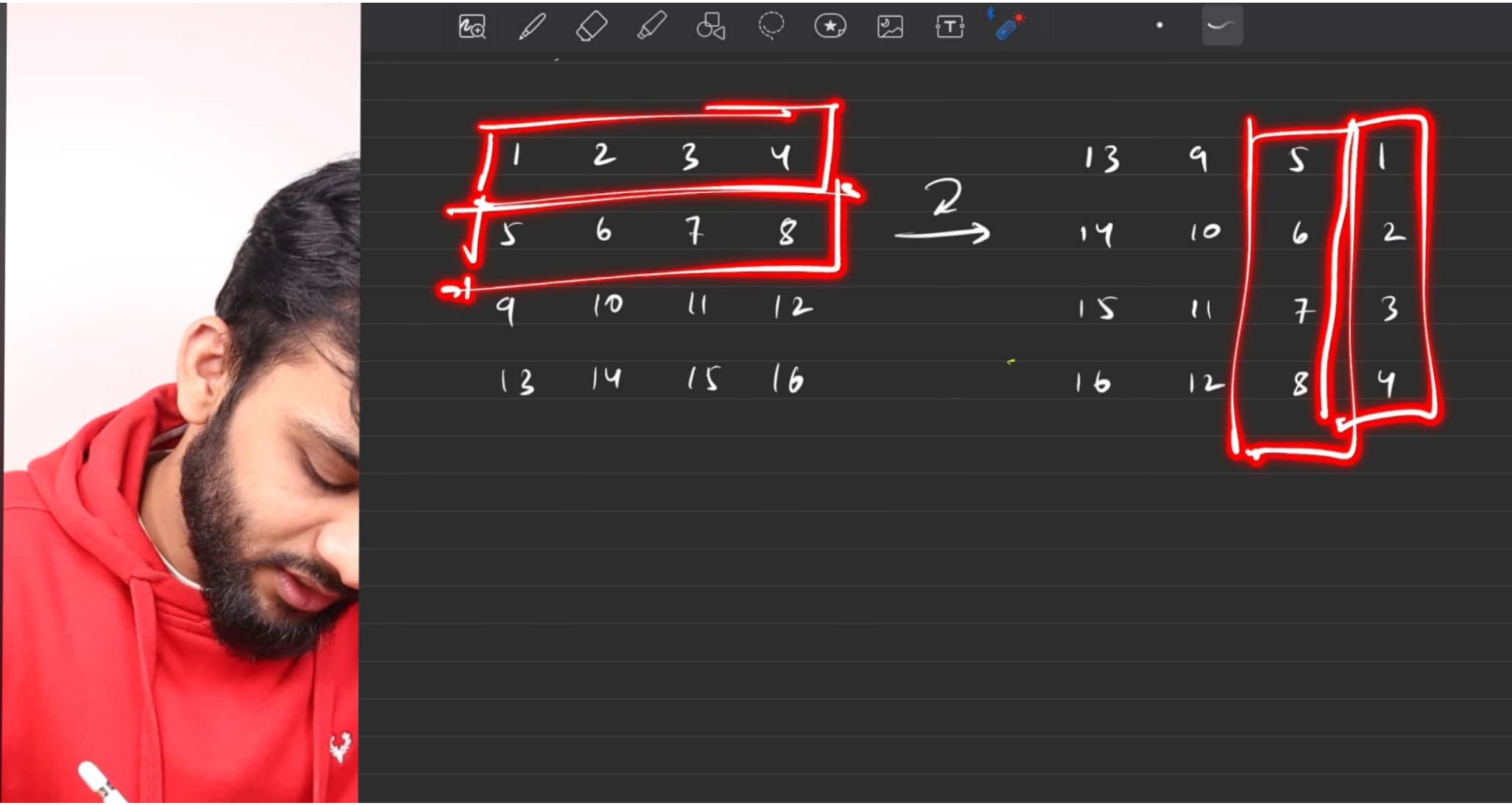
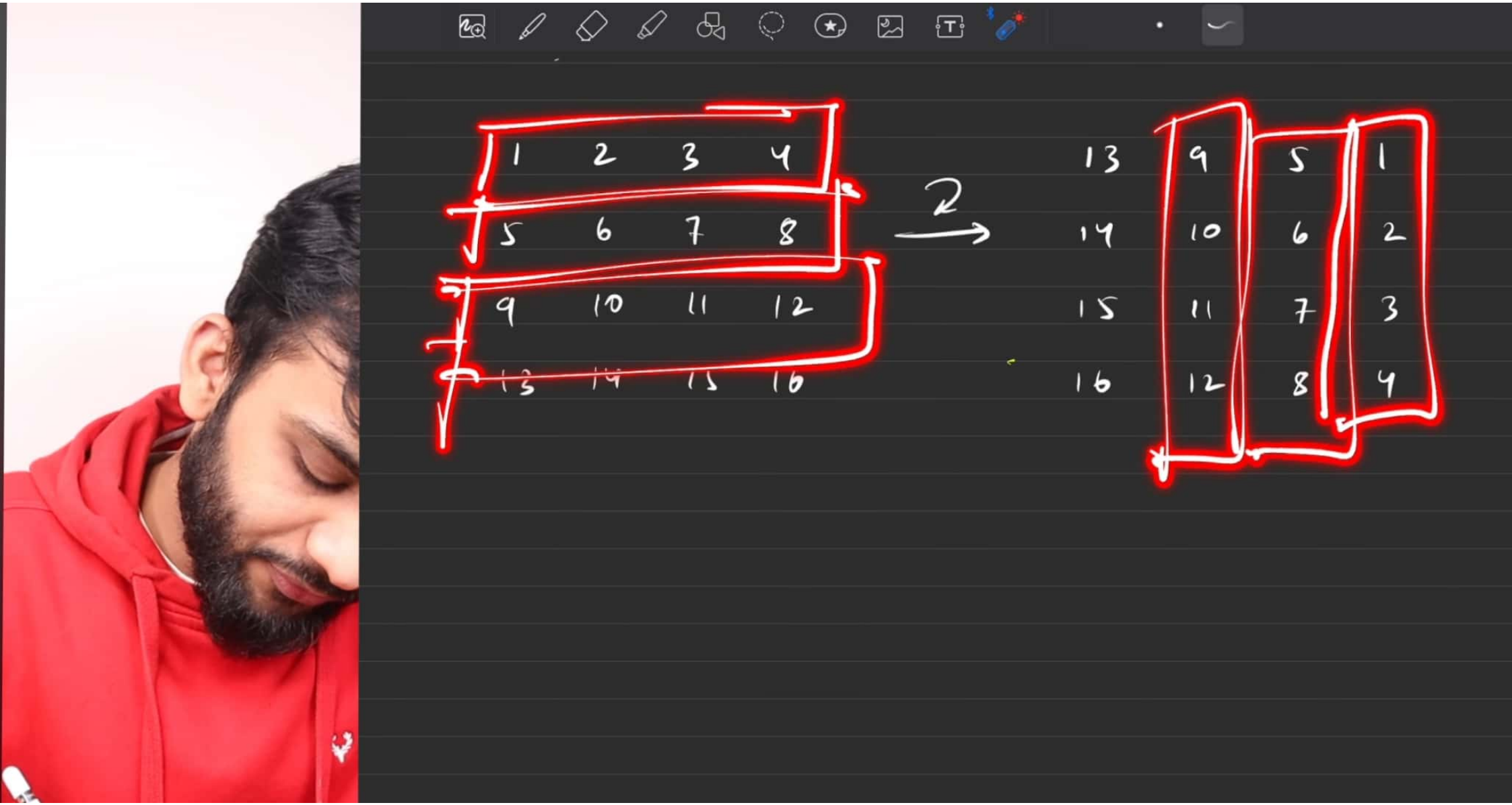
Rotate Image by 90 degree

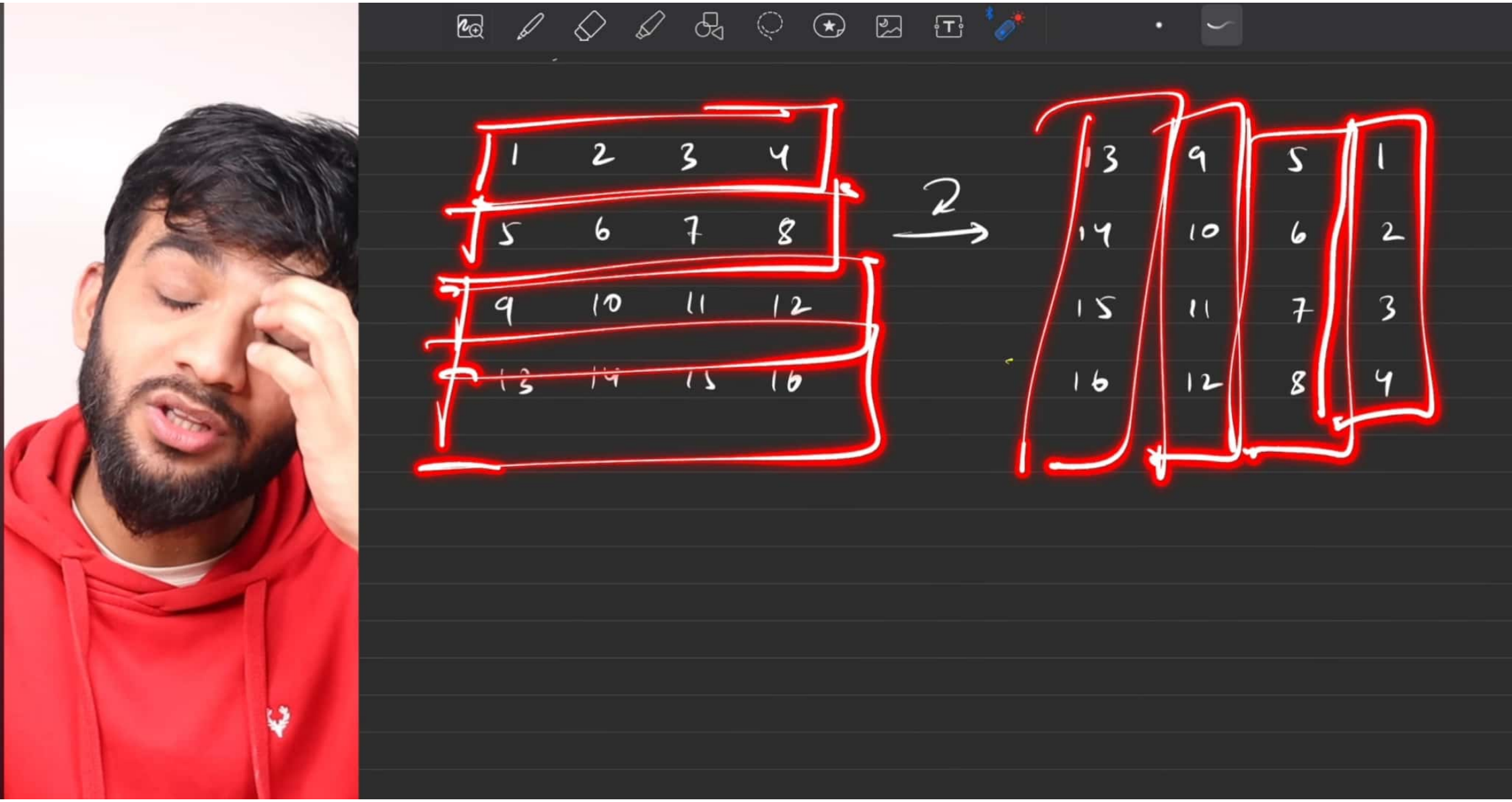

Rotate Image
90°
n x n
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4
108 of 136


Brute
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4
2x4
nxn


1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4


1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4


1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4


1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
2
13 9 5 1
14 10 6 2
15 11 7 3
16 12 8 4

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16

2 →

	0	1	2	3
0	13	9	5	1
1	14	10	6	2
2	15	11	7	3
3	16	12	8	4

[0][0] → [0][3]

[0][1] → [1][3]

[0][2] → [2][3]

[0][3] → [3][3]

[1][0] → [0][2]

[1][1] → [1][2]

[1][2] → [2][2]

[1][3] → [3][2]

2	9	10	11	12
3	13	14	15	16

i j

[0] [0] → [0] [3]

[0] [1] → [1] [3]

[0] [2] → [2] [3]

[0] [3] → [3] [3]

2	15	11	7	3
3	16	12	8	4

i j

[1] [0] → [0] [2]

[1] [1] → [1] [2]

[1] [2] → [2] [2]

[1] [3] → [3] [2]

$n = 4$

$0 \rightarrow 3$

$i \rightarrow (n-1)$

$0 \rightarrow (n-1) - 0$

$n - 1 = 3$

$1 \rightarrow 2$

$j \rightarrow (n-1) - i$

$1 \rightarrow (n-1) - 1$

$1 \rightarrow n - 2$

2

$(i) \rightarrow (n-1) - i$

3 13 14 15 16

i j j (n-1)-i

[0] [0] → [0] [3]

[0] [1] → [1] [3]

[0] [2] → [2] [3]

[0] [3] → [3] [3]

n=4

0 → 3

i → (n-1)

0 → (n-1)-0

3 16 12 8 4

j j

[1] [0] → [0] [2]

[1] [1] → [1] [2]

[1] [2] → [2] [2]

[1] [3] → [3] [2]

1 → 2

i → (n-1)-i

ans[n][n]

TC → O(n²)

SC → O(n²)

for (i = 0 → n)

{

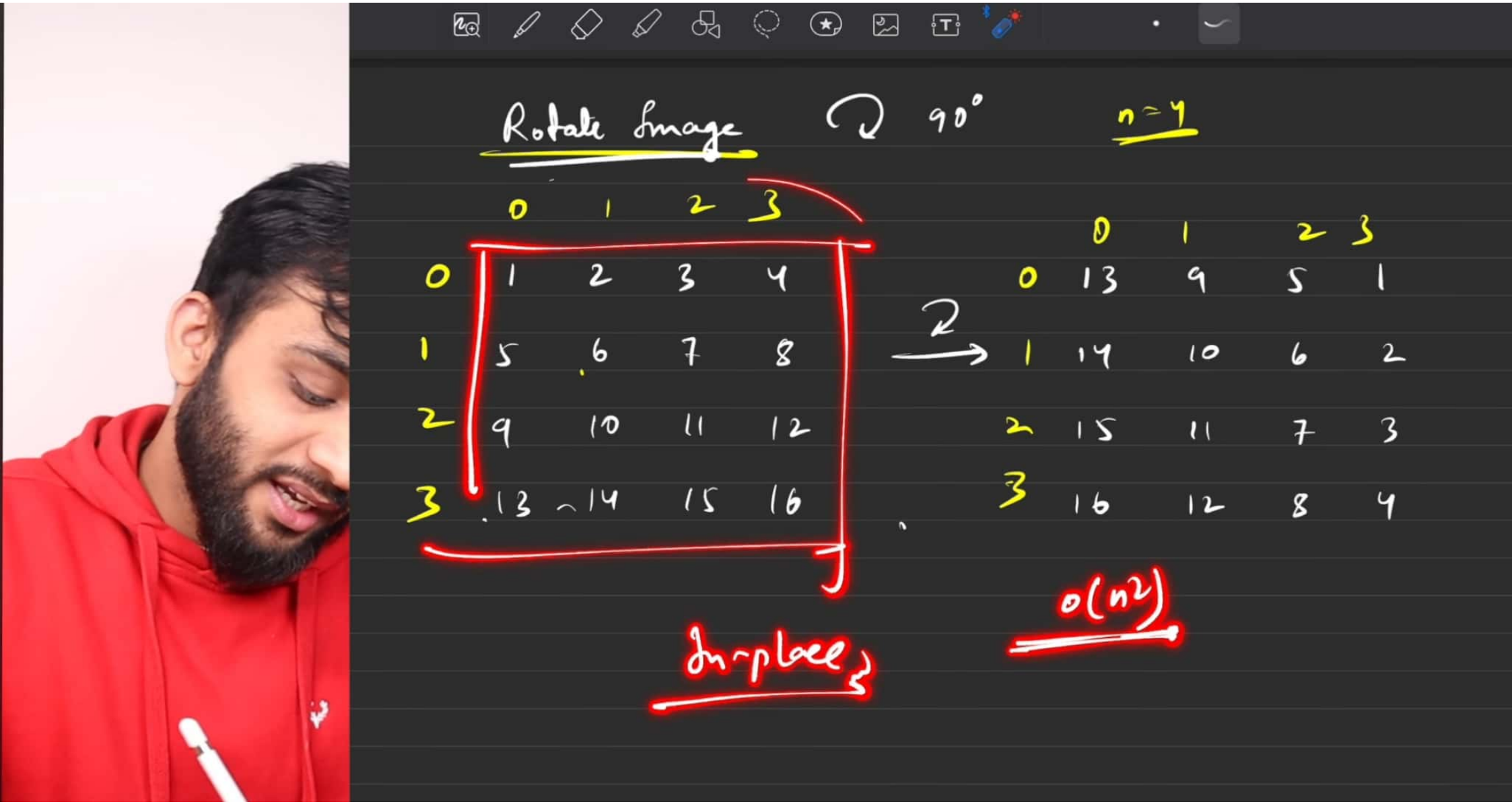
for (j = 0 → n)

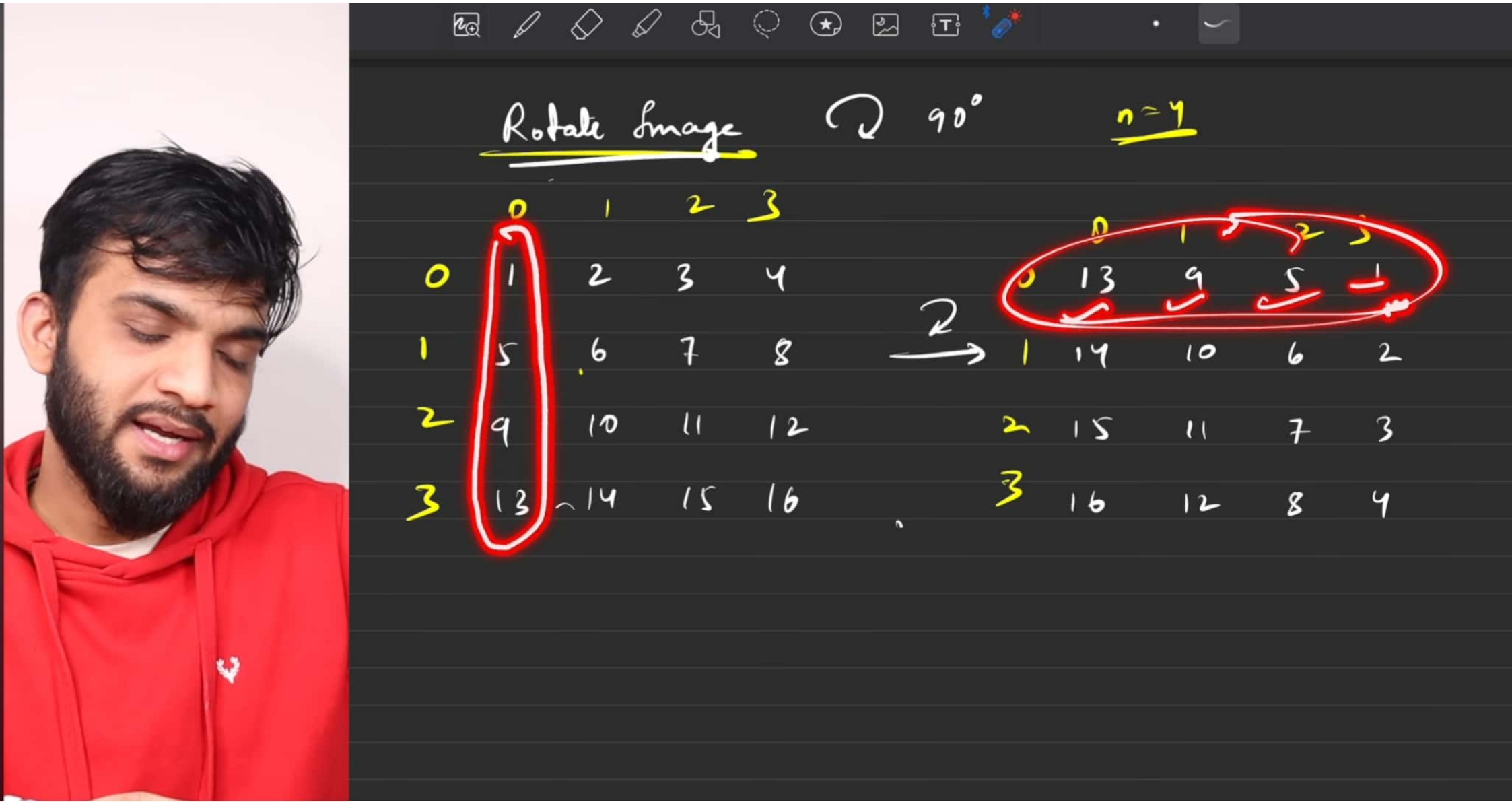
{

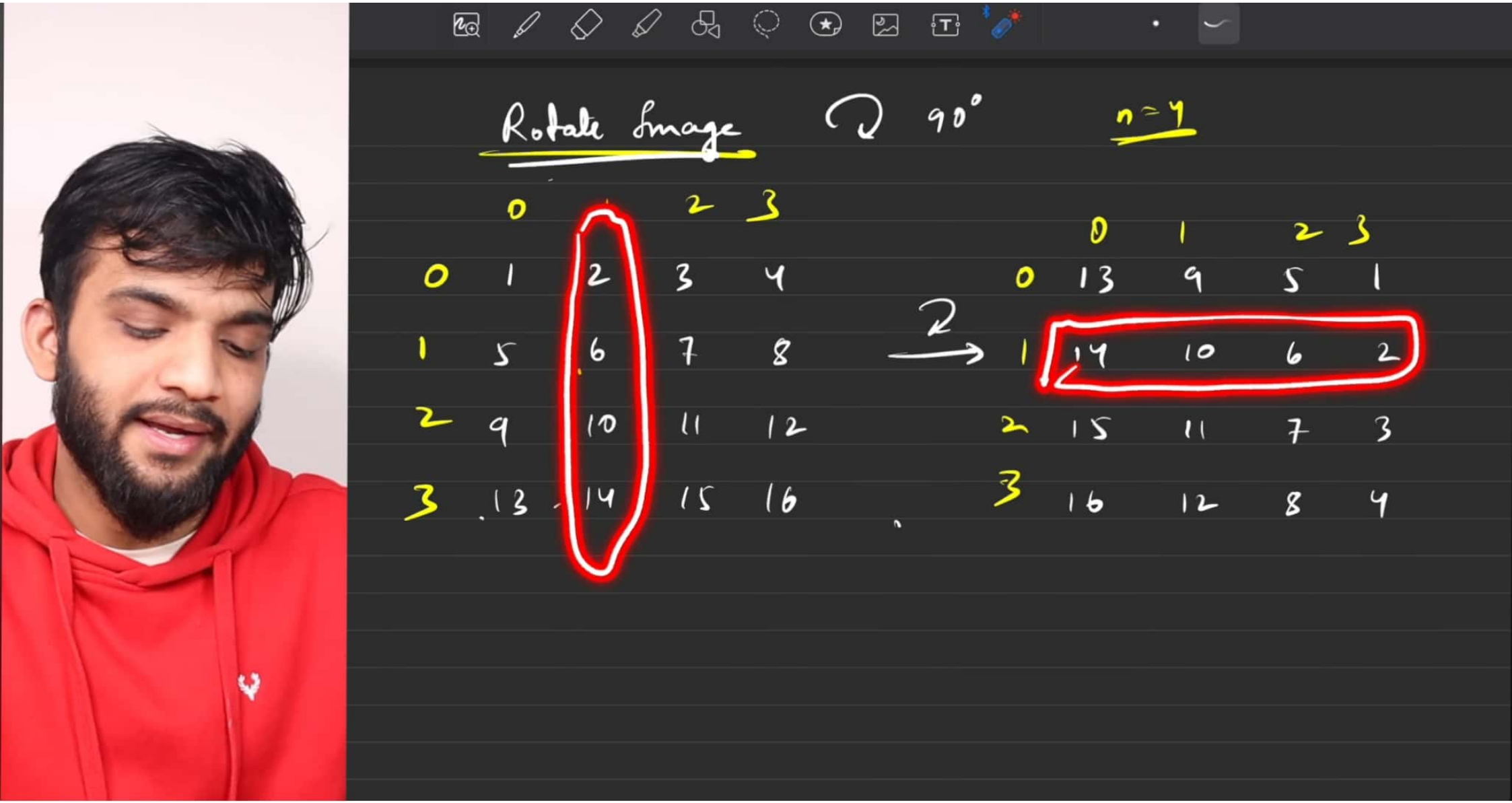
ans[j][n-1-i] = matrix[i][j]

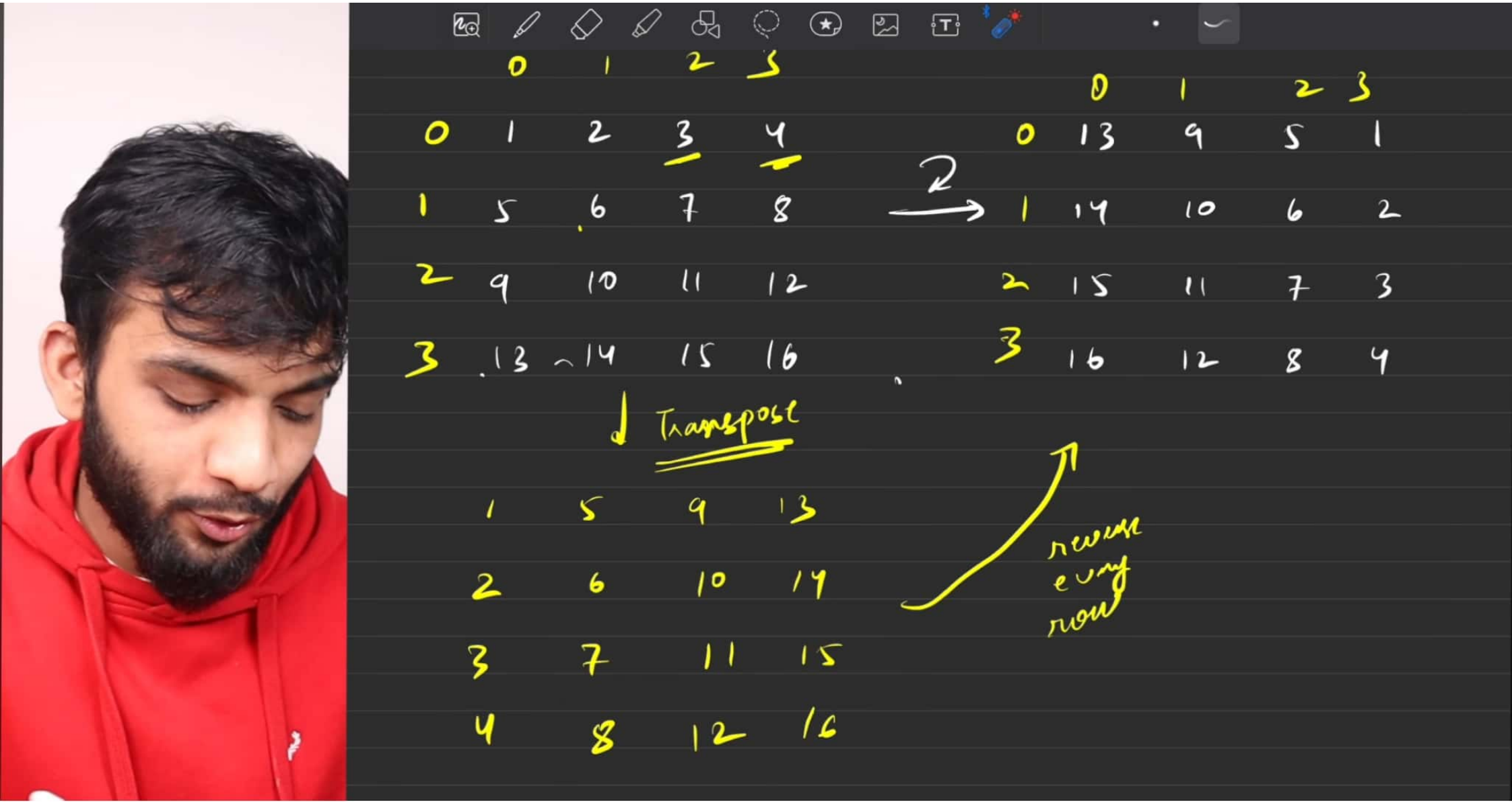
}

}


Rotate Image
90°
n = 4
0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
2
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4
In-place
O(n2)


Rotate Image
90°
n = 4
0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4


Rotate Image
90°
n = 4
0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
2
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4


0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
2
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4
Transpose
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16
reverse every row

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16

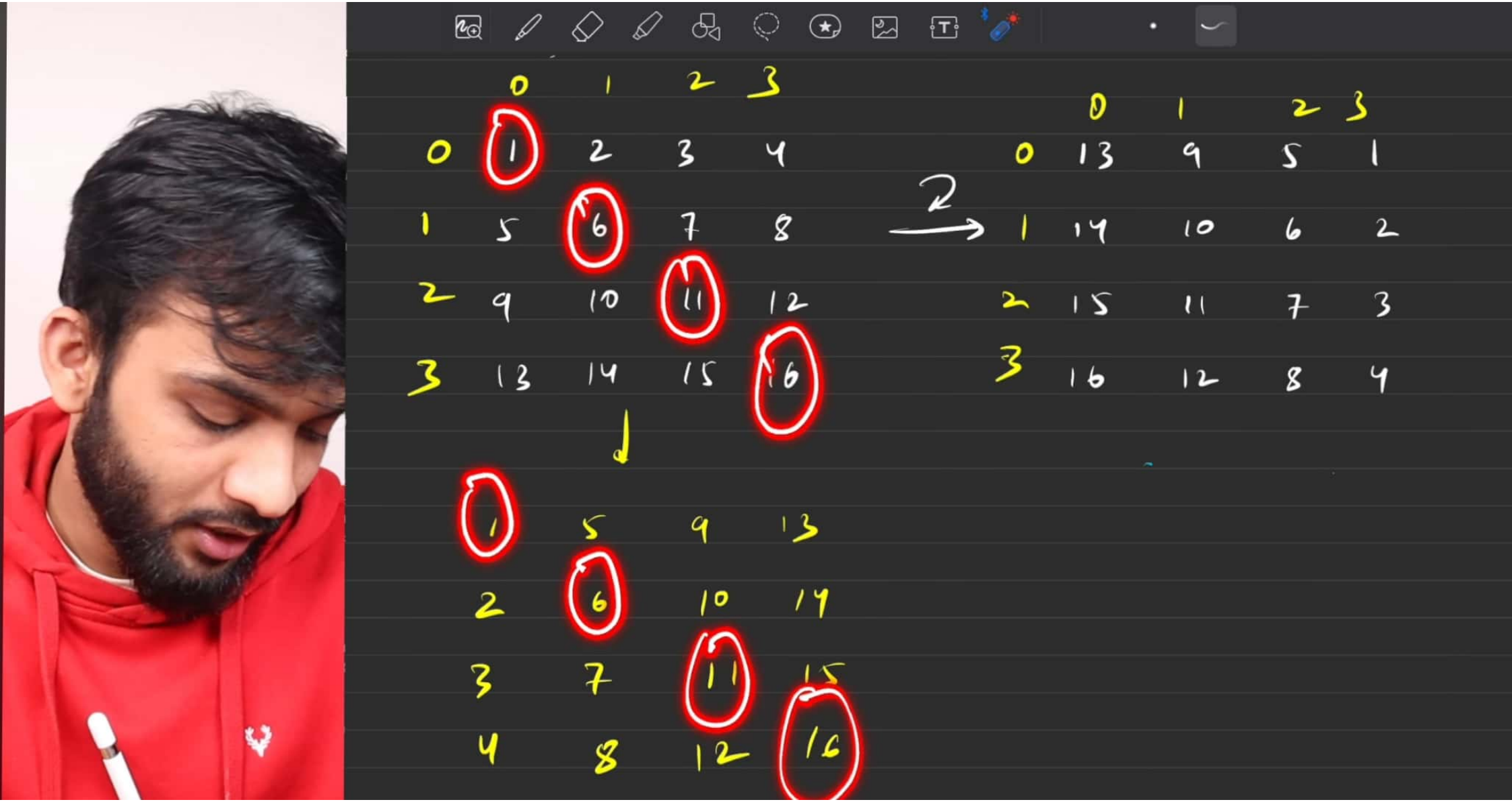
↓

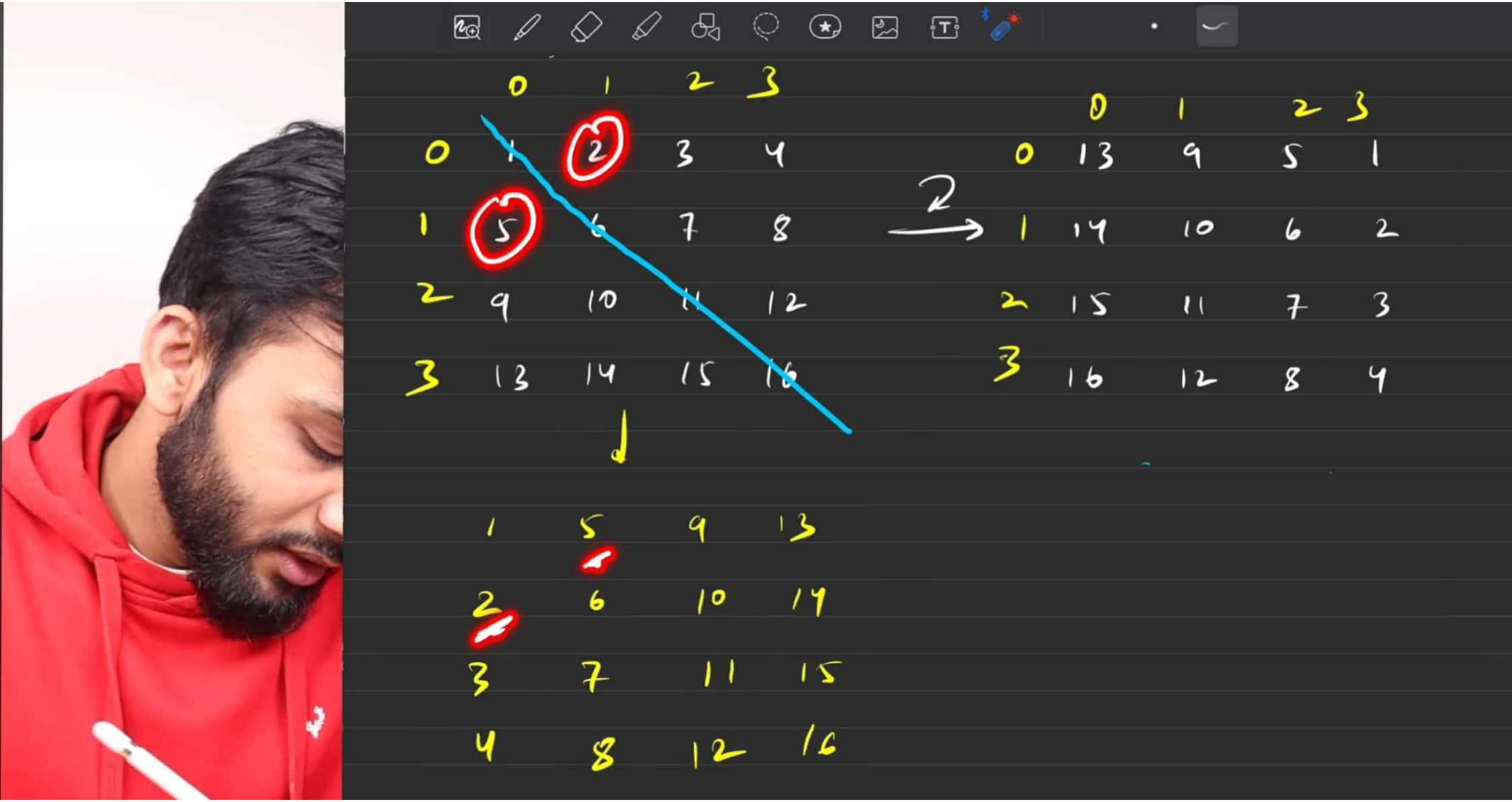
	1	5	9	13
	2	6	10	14
	3	7	11	15
	4	8	12	16

2 →

	0	1	2	3
0	13	9	5	1
1	14	10	6	2
2	15	11	7	3
3	16	12	8	4

1. Transpose
2. Ro


0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
2
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16


0 1 2 3
0 1 2 3 4
1 5 6 7 8
2 9 10 11 12
3 13 14 15 16
2
0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16

	1	5	9	13
2	6	10	14	
3	7	11	15	
4	8	12	16	

2 →

	0	1	2	3
0	13	9	5	1
1	14	10	6	2
2	15	11	7	3
3	16	12	8	4

{0} {1} → {1} {0}

{0} {2} → {2} {0}

{0} {3} → {3} {0}

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16

2 →

	0	1	2	3
0	13	9	5	1
1	14	10	6	2
2	15	11	7	3
3	16	12	8	4

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

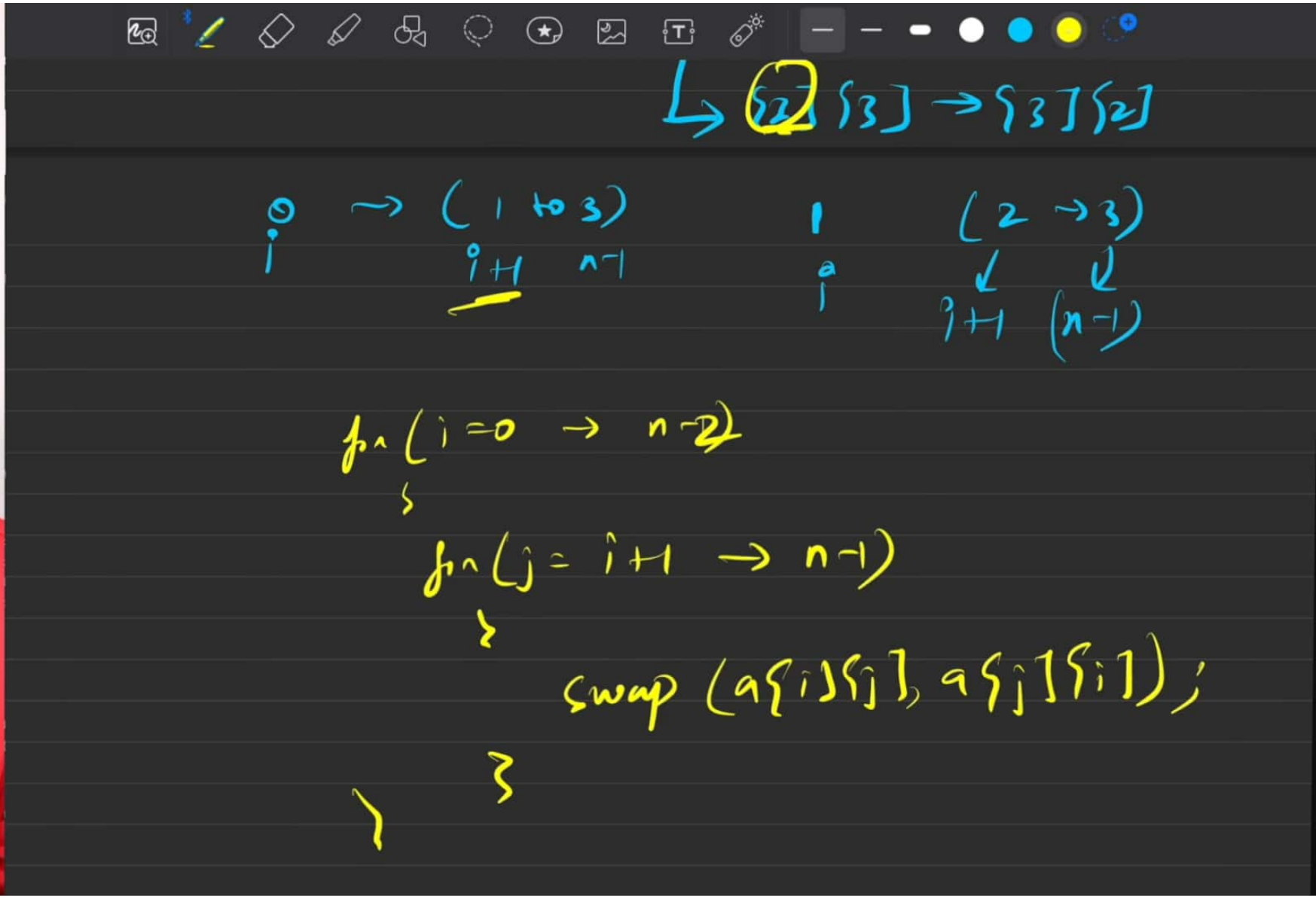
[0][1] → [1][0]

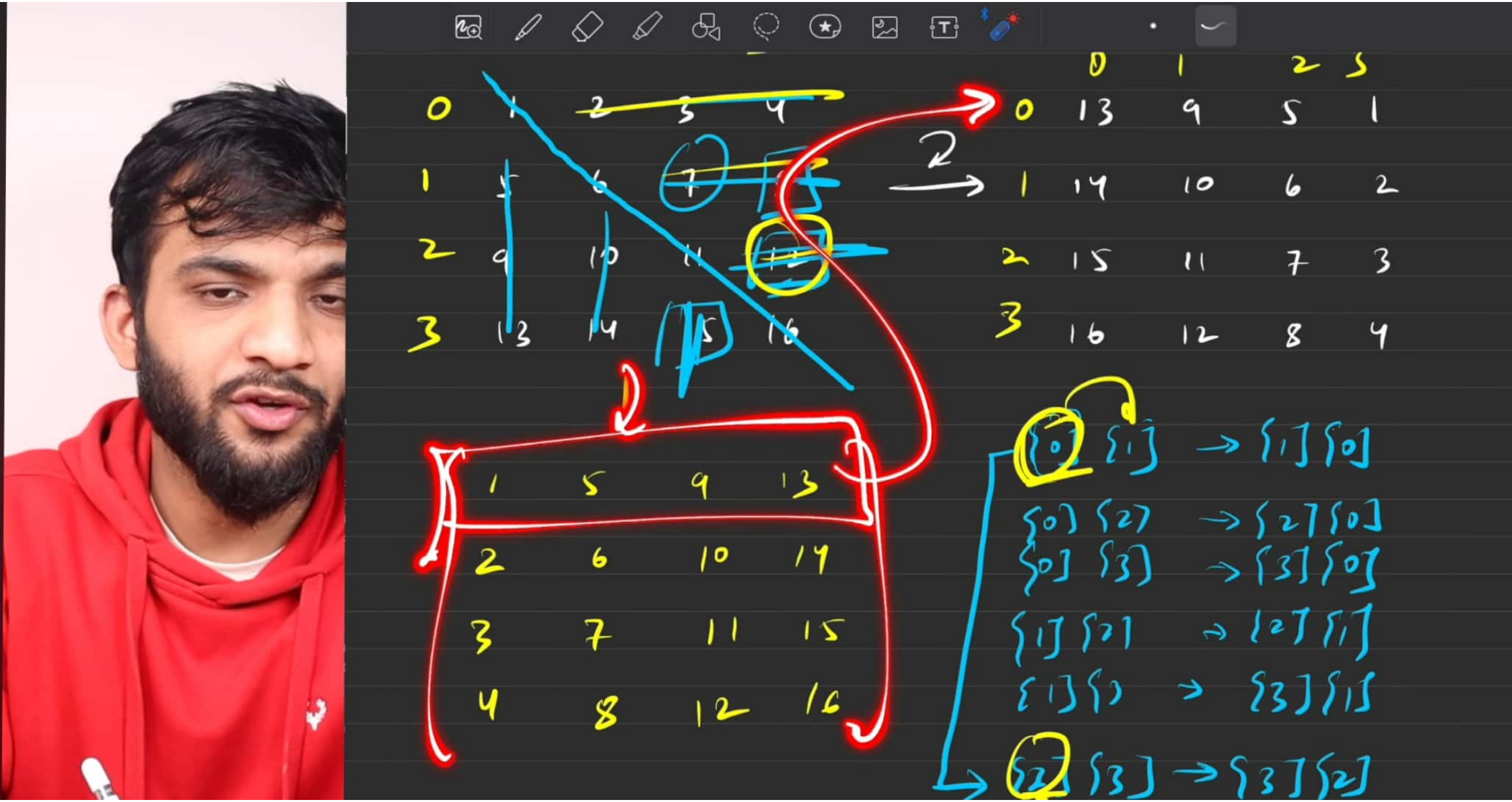
[0][2] → [2][0]

[0][3] → [3][0]

[1][0] → [0][1]

[1][0] → [3][1]


[2] [3] → [3] [2]
i → (i+1 to n-1)
i → (2 → 3)
(i+1) (n-1)
fn(i=0 → n-2)
fn(j = i+1 → n-1)
swap(a[i],a[j], a[j],a[i]);


0 1 2 3
0 13 9 5 1
1 14 10 6 2
2 15 11 7 3
3 16 12 8 4
2
1 5 9 13
2 6 10 14
3 7 11 15
4 8 12 16
[0][1] → [1][0]
[0][2] → [2][0]
[0][3] → [3][0]
[1][2] → [2][1]
[1][3] → [3][1]
[2][3] → [3][2]

codestudio POWERED BY CODING NINJAS

Guided Paths | Contests | Interview Prep | Practice | Resources | Problem of the day

Topics | Problem | Submissions | Solution | New

Current Submission Report

Correct Answer
few secs ago

Test Cases EXP: 11/11 | 40/40

Penalty 0% | C++

Did you find these test cases useful?

Previous Submissions All languages

Status	Test cases	EXP	Penalty	Langua
✓	11/11	+ 40	0%	C++
✓	11/11	+ 40	0%	C++

Prev | Next

```cpp
#include<bits/stdc++.h>
void rotateMatrix(vector<vector<int>> &mat){
    int n = mat.size();
    // transpose
    // O(N / 2 * N / 2)
    for(int i = 0;i<n-1;i++) {
        for (int j = i + 1; j < n; j++) {
            swap(mat[i][j], mat[j][i]);
        }
    }
    // reverse
    // O(N * N / 2)
    for(int i = 0;i<n;i++) {
        // row is mat[i]
        reverse(mat[i].begin(), mat[i].end());
    }
}
```

View hints | Last saved on 5:17:40 AM

Merge Overlapping Intervals

Merge Overlapping Subintervals

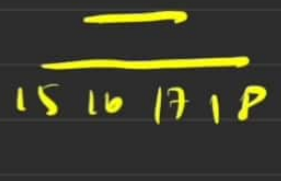
(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

{1 6} {8 11} {15, 18}

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

{1 6} {8 11} {15, 18}



(1, 6)

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3)

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

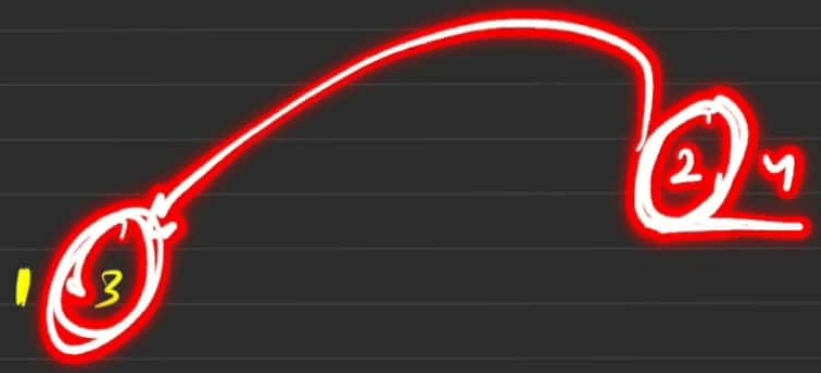
(1, 3)

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)



Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

1, 4

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

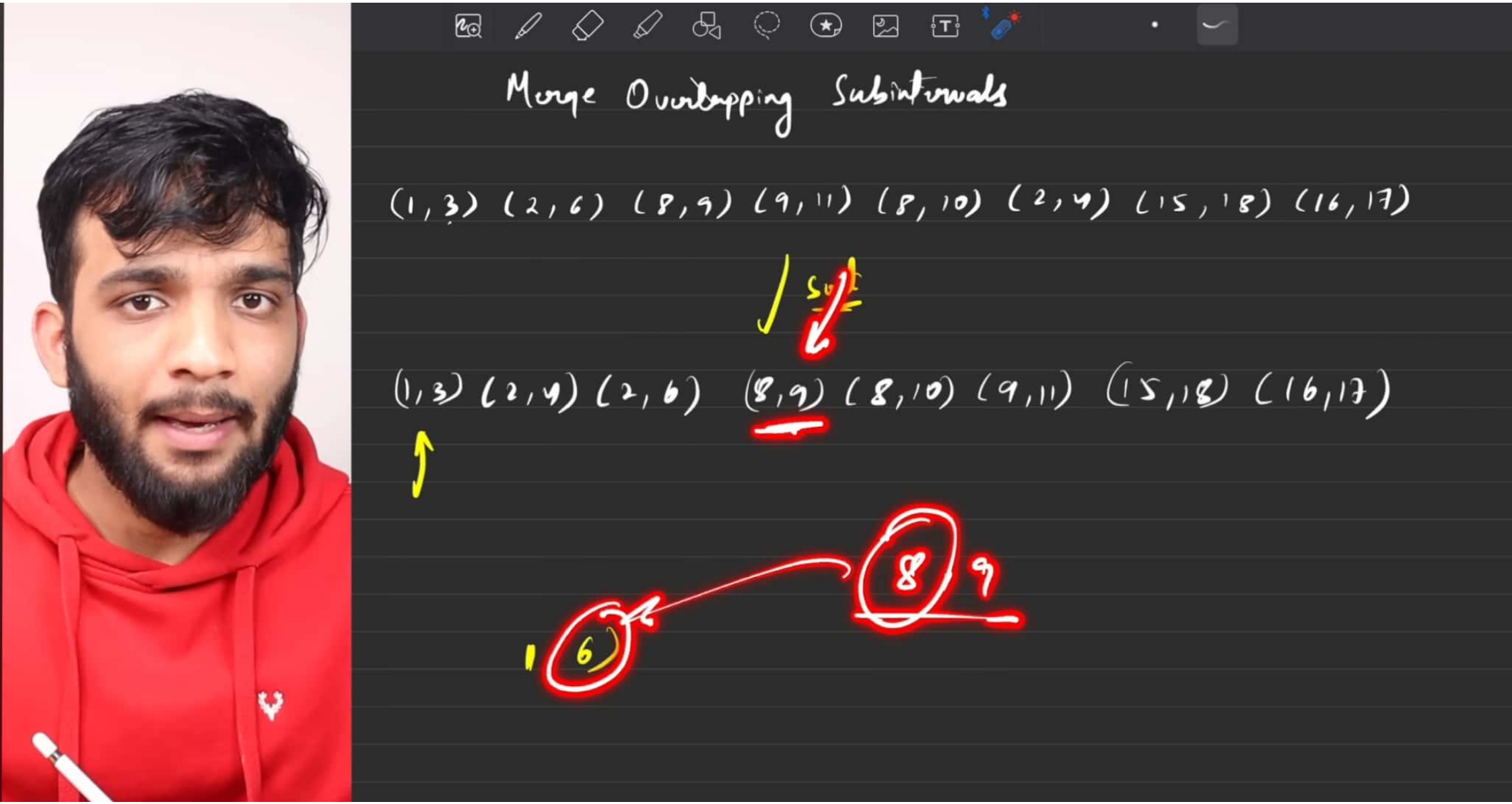
Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

(1, 6)


Merge Overlapping Subintervals
(1,3) (2,6) (8,9) (9,11) (8,10) (2,4) (15,18) (16,17)
(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)
8
9
6

Merge Overlapping Subintervals

(1,3) (2,6) (8,9) (9,11) (8,10) (2,4) (15,18) (16,17)

sort

(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)

2

(1,6) (8,11)

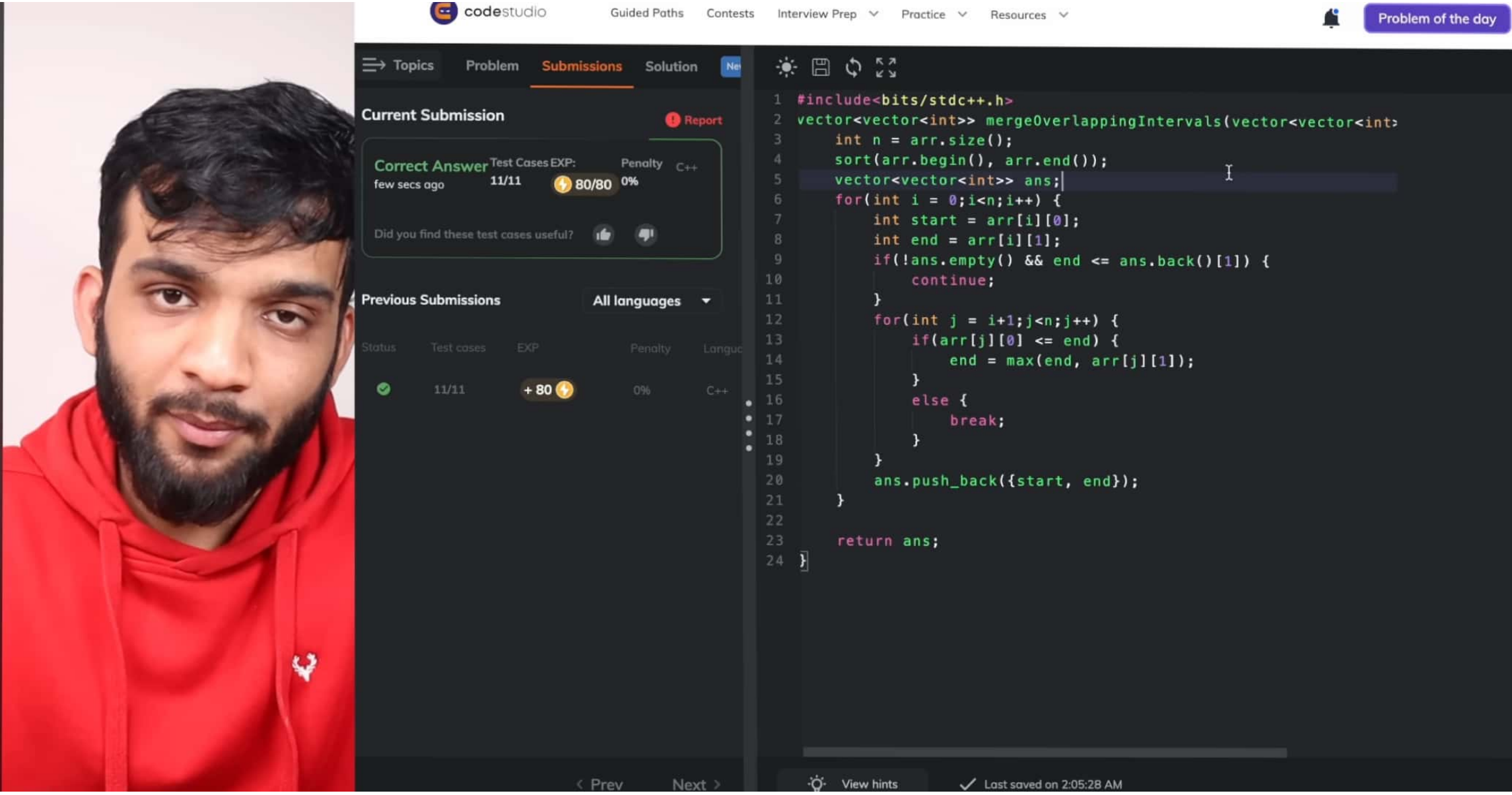
Merge Overlapping Subintervals

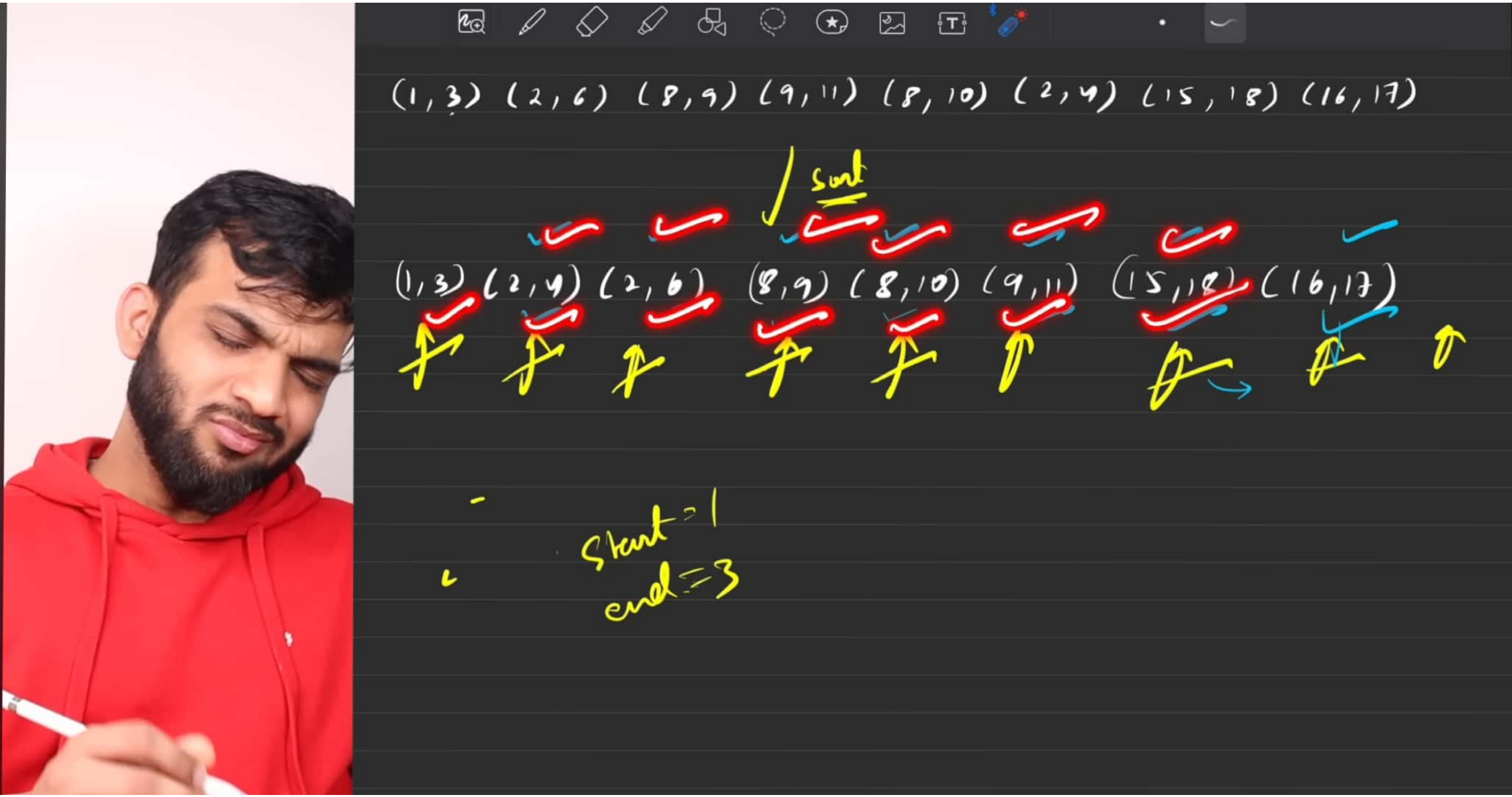
(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

[(1, 6) (8, 11) (15, 18)]


codestudio
Guided Paths
Contests
Interview Prep
Practice
Resources
Problem of the day
Topics
Problem
Submissions
Solution
Current Submission
Report
Correct Answer
few secs ago
Test Cases
11/11
EXP:
80/80
Penalty
0%
C++
Did you find these test cases useful?
Previous Submissions
All languages
Status
Test cases
EXP
Penalty
Langua
11/11
+ 80
0%
C++
Prev
Next
#include<bits/stdc++.h>
vector<vector<int>> mergeOverlappingIntervals(vector<vector<int>
int n = arr.size();
sort(arr.begin(), arr.end());
vector<vector<int>> ans;
for(int i = 0;i<n;i++) {
int start = arr[i][0];
int end = arr[i][1];
if(!ans.empty() && end <= ans.back()[1]) {
continue;
}
for(int j = i+1;j<n;j++) {
if(arr[j][0] <= end) {
end = max(end, arr[j][1]);
}
else {
break;
}
}
ans.push_back({start, end});
}
return ans;
}
View hints
Last saved on 2:05:28 AM


(1,3) (2,6) (8,9) (9,11) (8,10) (2,4) (15,18) (16,17)
sort
(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)
start=1
end=3

(1,2) (3,4) (6,2)

$N \log W + O(2n)$

$O(W)$

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

(1, 6) (8, 9)

Merge Overlapping Subintervals

(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

(1, 6) (8, 11)

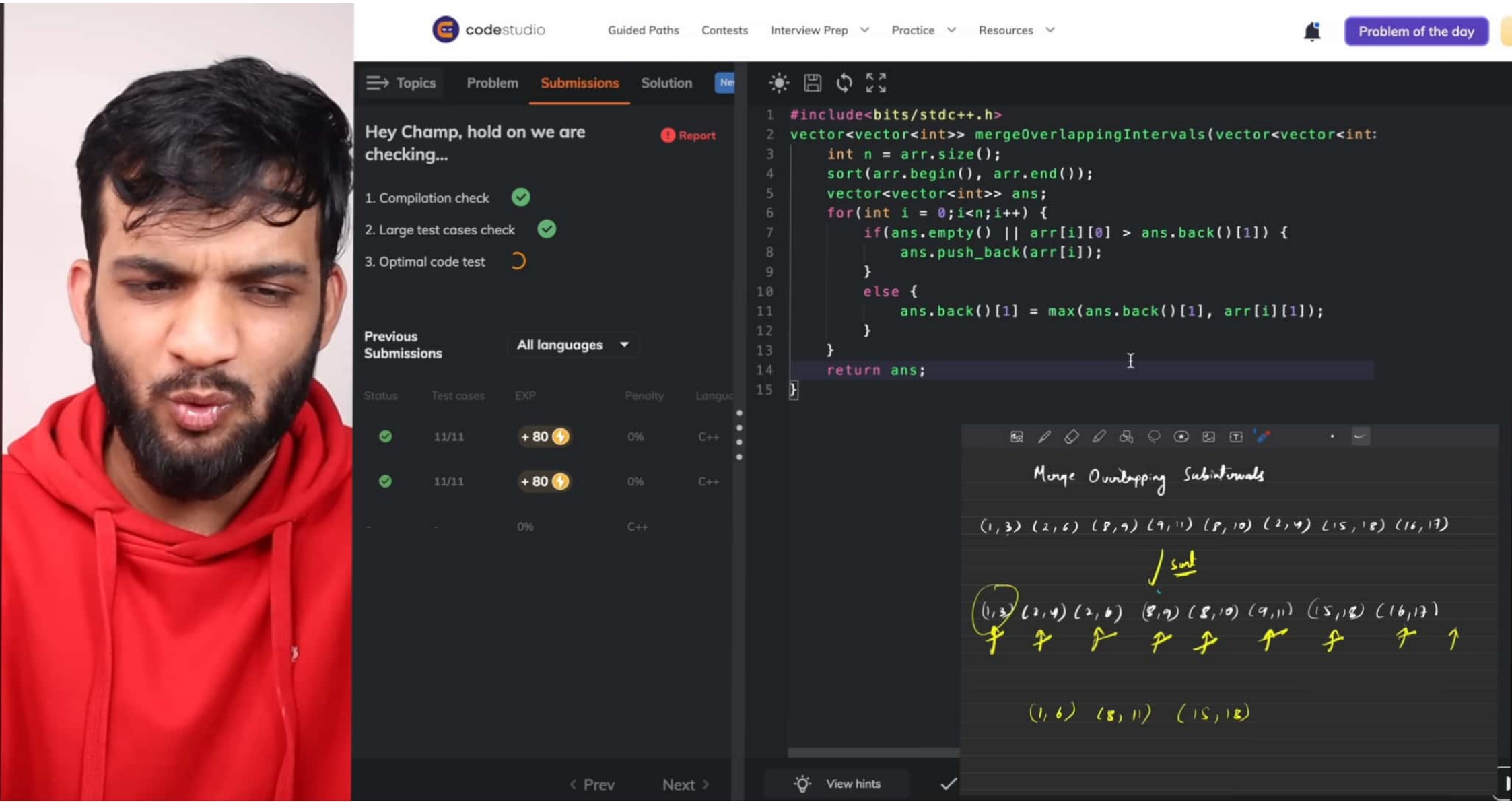
Merge Overlapping Subintervals

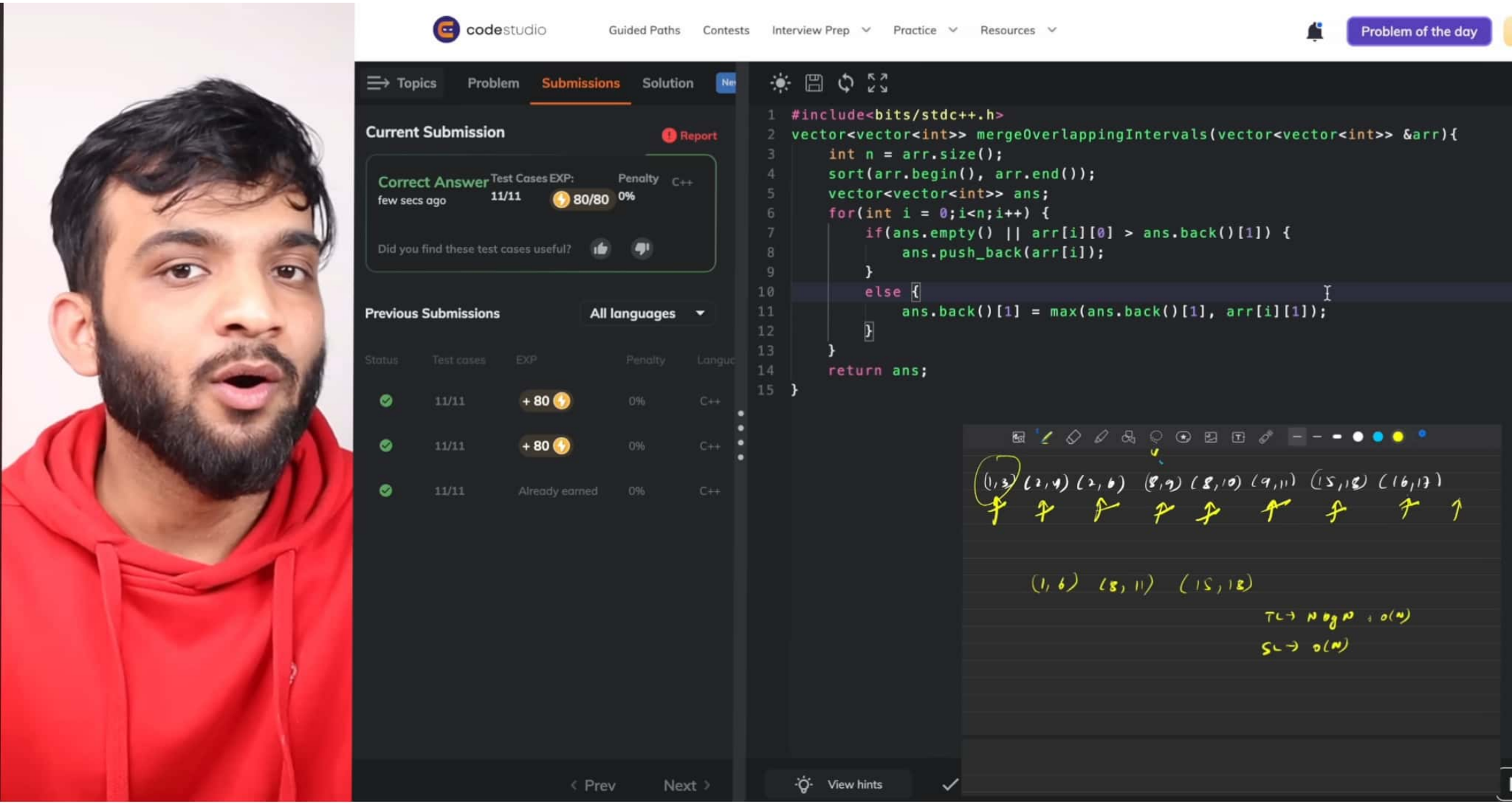
(1, 3) (2, 6) (8, 9) (9, 11) (8, 10) (2, 4) (15, 18) (16, 17)

sort

(1, 3) (2, 4) (2, 6) (8, 9) (8, 10) (9, 11) (15, 18) (16, 17)

(1, 6) (8, 11) (15, 18)


codestudio
Guided Paths
Contests
Interview Prep
Practice
Resources
Problem of the day
Topics
Problem
Submissions
Solution
Hey Champ, hold on we are checking...
Report
1. Compilation check
2. Large test cases check
3. Optimal code test
Previous Submissions
All languages
Status
Test cases
EXP
Penalty
11/11
+80
0%
C++
11/11
+80
0%
C++
0%
C++
Prev
Next
#include<bits/stdc++.h>
vector<vector<int>> mergeOverlappingIntervals(vector<vector<int:
int n = arr.size();
sort(arr.begin(), arr.end());
vector<vector<int>> ans;
for(int i = 0;i<n;i++) {
if(ans.empty() || arr[i][0] > ans.back()[1]) {
ans.push_back(arr[i]);
}
else {
ans.back()[1] = max(ans.back()[1], arr[i][1]);
}
}
return ans;
}
Merge Overlapping Subintervals
(1,3) (2,6) (8,9) (9,11) (8,10) (2,4) (15,18) (16,17)
Sort
(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)
(1,6) (8,11) (15,18)
View hints


codestudio
Guided Paths
Contests
Interview Prep
Practice
Resources
Problem of the day
Topics
Problem
Submissions
Solution
Current Submission
Report
Correct Answer
few secs ago
Test Cases
11/11
EXP:
80/80
Penalty
0%
C++
Did you find these test cases useful?
Previous Submissions
All languages
Status
Test cases
EXP
Penalty
Langua
11/11
+ 80
0%
C++
11/11
+ 80
0%
C++
11/11
Already earned
0%
C++
Prev
Next
View hints
#include<bits/stdc++.h>
vector<vector<int>> mergeOverlappingIntervals(vector<vector<int>> &arr){
int n = arr.size();
sort(arr.begin(), arr.end());
vector<vector<int>> ans;
for(int i = 0;i<n;i++) {
if(ans.empty() || arr[i][0] > ans.back()[1]) {
ans.push_back(arr[i]);
}
else {
ans.back()[1] = max(ans.back()[1], arr[i][1]);
}
}
return ans;
}
(1,3) (2,4) (2,6) (8,9) (8,10) (9,11) (15,18) (16,17)
(1,6) (8,11) (15,18)
TC → N log N : O(N)
SC → O(N)