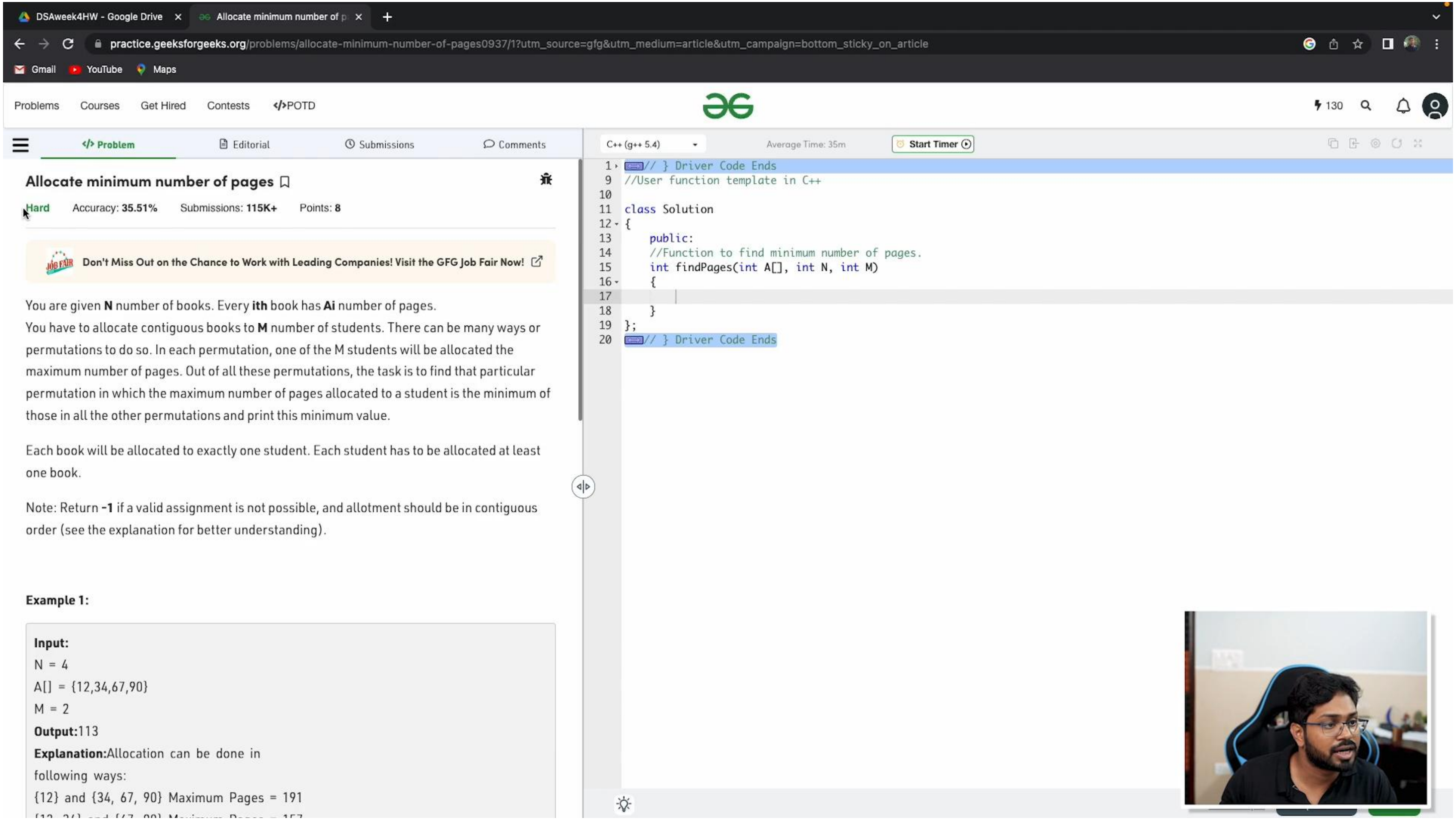


DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
C++ (g++ 5.4)
Average Time: 35m
Start Timer

Allocate minimum number of pages

Hard Accuracy: **35.51%** Submissions: **115K+** Points: **8**

Don't Miss Out on the Chance to Work with Leading Companies! Visit the GFG Job Fair Now!

You are given **N** number of books. Every **ith** book has **Ai** number of pages.
You have to allocate contiguous books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is the minimum of those in all the other permutations and print this minimum value.

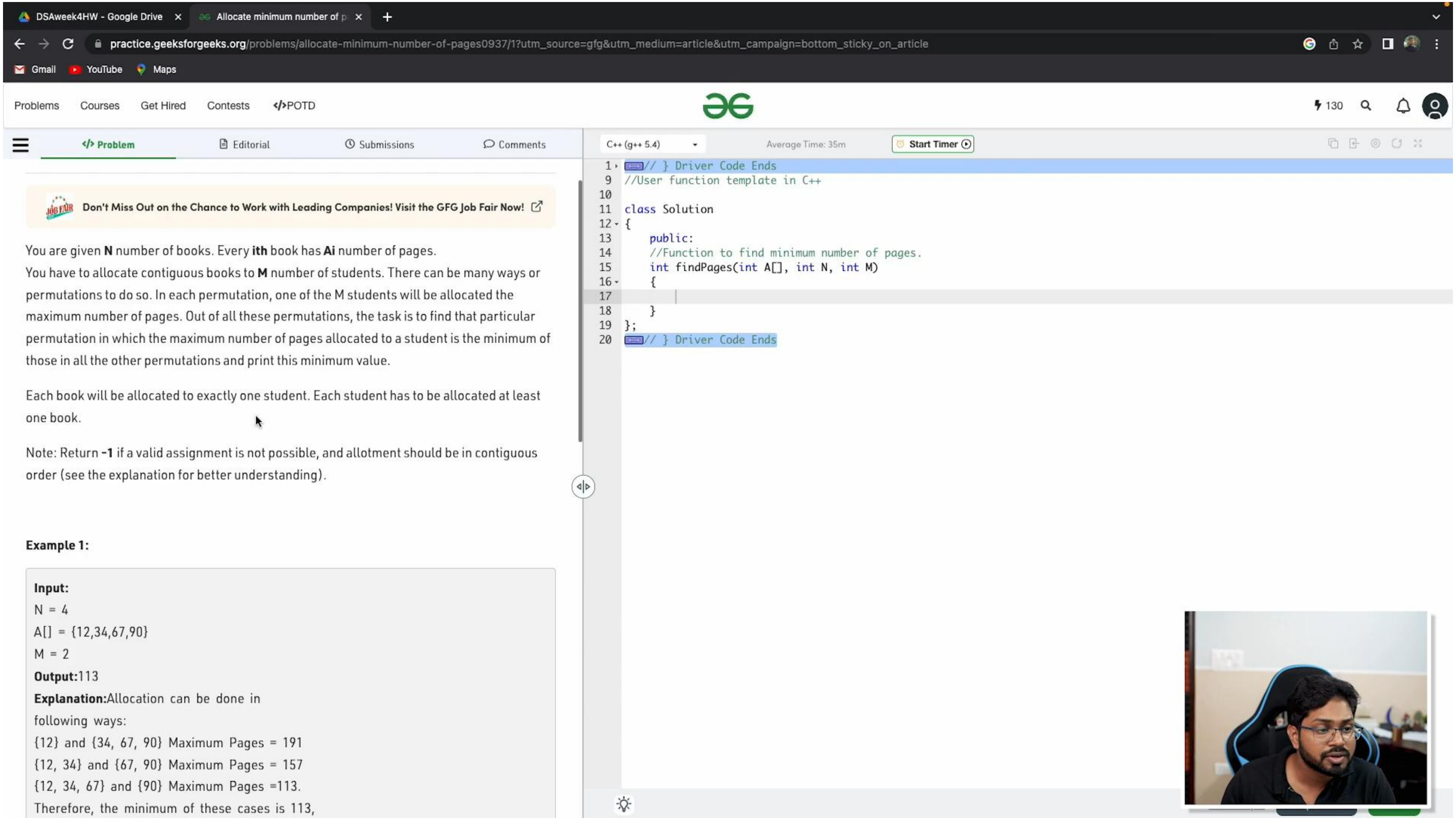
Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

```
Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation:Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
```

```
// } Driver Code Ends
//User function template in C++

class Solution
{
    public:
    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        
    }
};
// } Driver Code Ends
```


DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
Don't Miss Out on the Chance to Work with Leading Companies! Visit the GFG Job Fair Now!
C++ (g++ 5.4)
Average Time: 35m
Start Timer

You are given **N** number of books. Every **ith** book has **Ai** number of pages.
You have to allocate contiguous books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is the minimum of those in all the other permutations and print this minimum value.

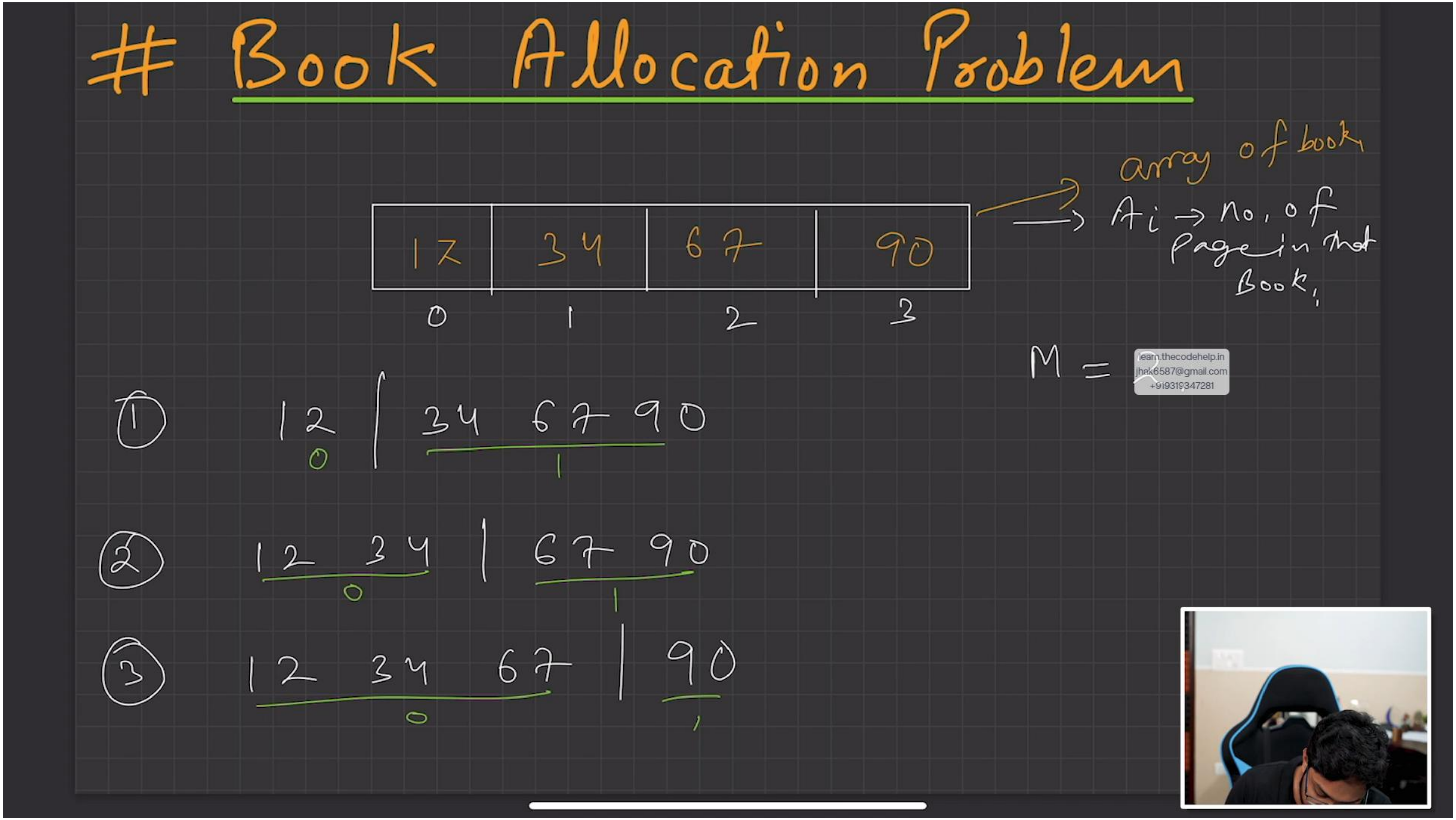
Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

```
Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation:Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
{12, 34, 67} and {90} Maximum Pages =113.
Therefore, the minimum of these cases is 113,
```

```cpp
// } Driver Code Ends
//User function template in C++

class Solution
{
    public:
    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        
    }
};
// } Driver Code Ends
```



Book Allocation Problem

12	34	67	90
0	1	2	3

array of book

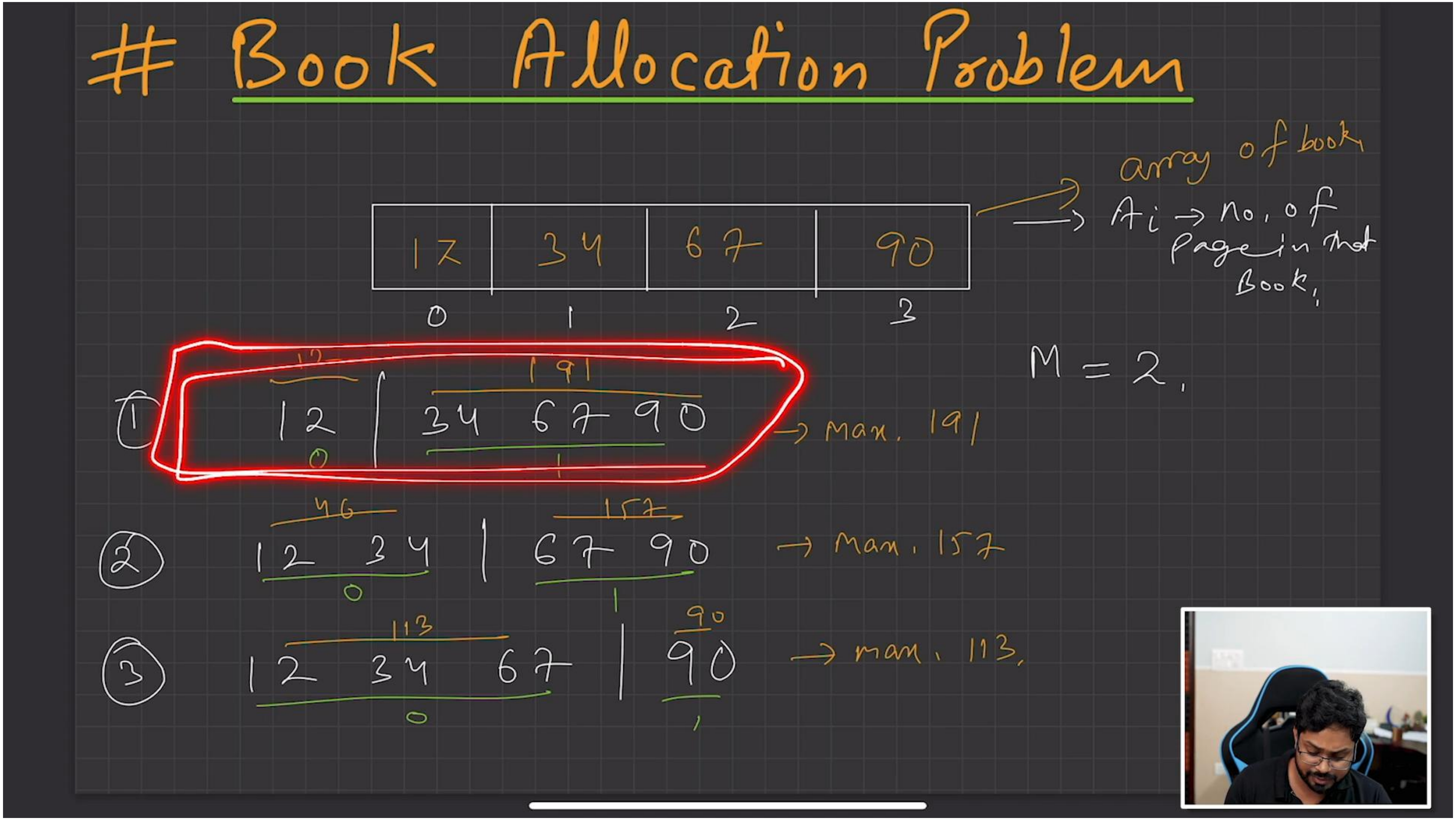
→ A_i → no. of page in that Book.

M = 2

① $\underbrace{12}_{0} \mid \underbrace{34\ 67\ 90}_{1}$

② $\underbrace{12\ 34}_{0} \mid \underbrace{67\ 90}_{1}$

③ $\underbrace{12\ 34\ 67}_{0} \mid \underbrace{90}_{1}$



Book Allocation Problem

12	34	67	90
0	1	2	3

→ array of book

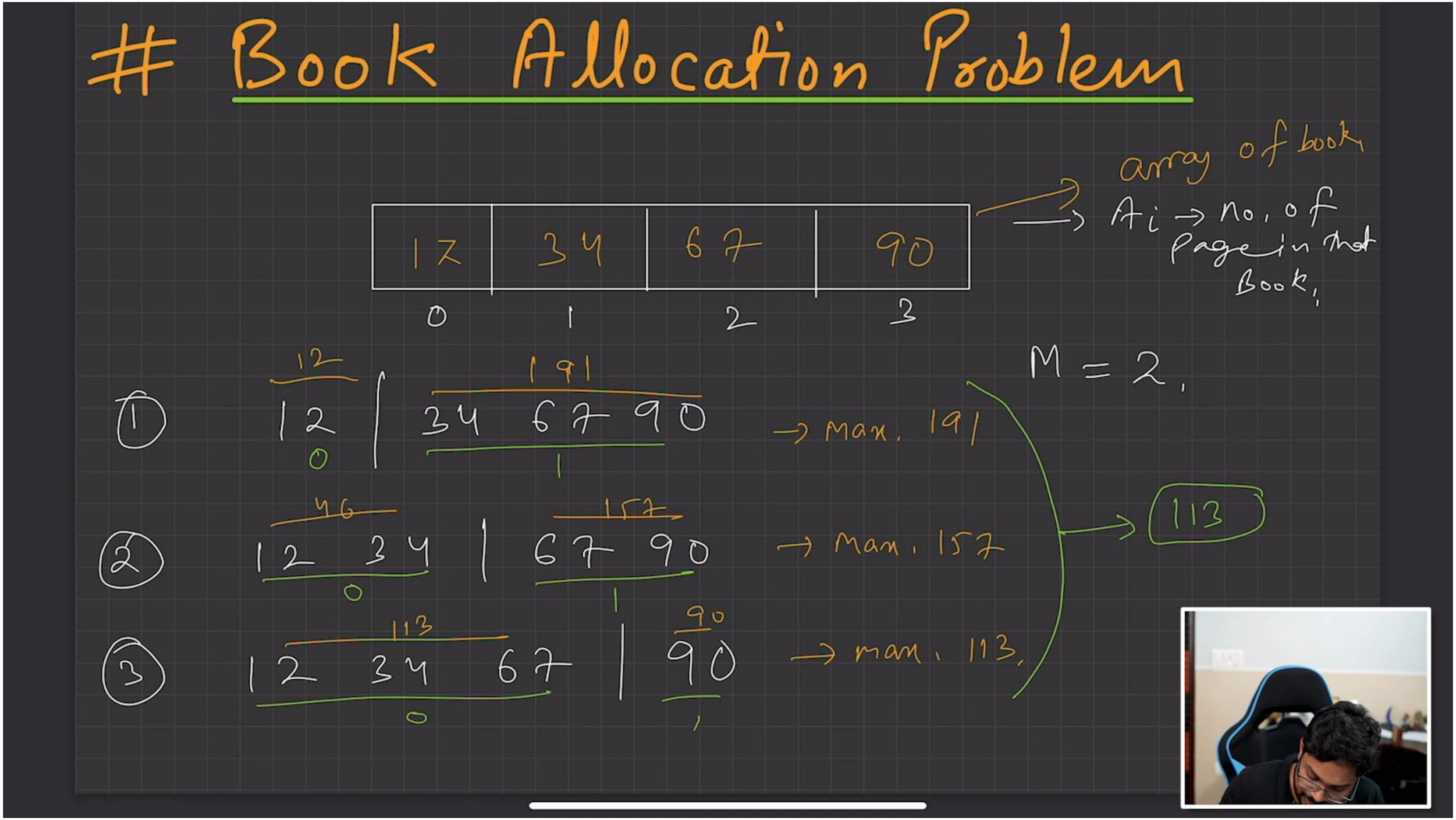
→ A_i → no. of page in that Book.

M = 2.

① $\overset{12}{\underset{0}{12}} \mid \overset{191}{\underset{1}{34\ 67\ 90}}$ → max. 191

② $\overset{46}{\underset{0}{12\ 34}} \mid \overset{157}{\underset{1}{67\ 90}}$ → max. 157

③ $\overset{113}{\underset{0}{12\ 34\ 67}} \mid \overset{90}{\underset{1}{90}}$ → max. 113.



Book Allocation Problem

12	34	67	90
0	1	2	3

array of book,

Ai → no. of page in that Book.

M = 2.

① $\overset{12}{12}$ | $\overset{191}{34\ 67\ 90}$ → max. 191

0 | 1

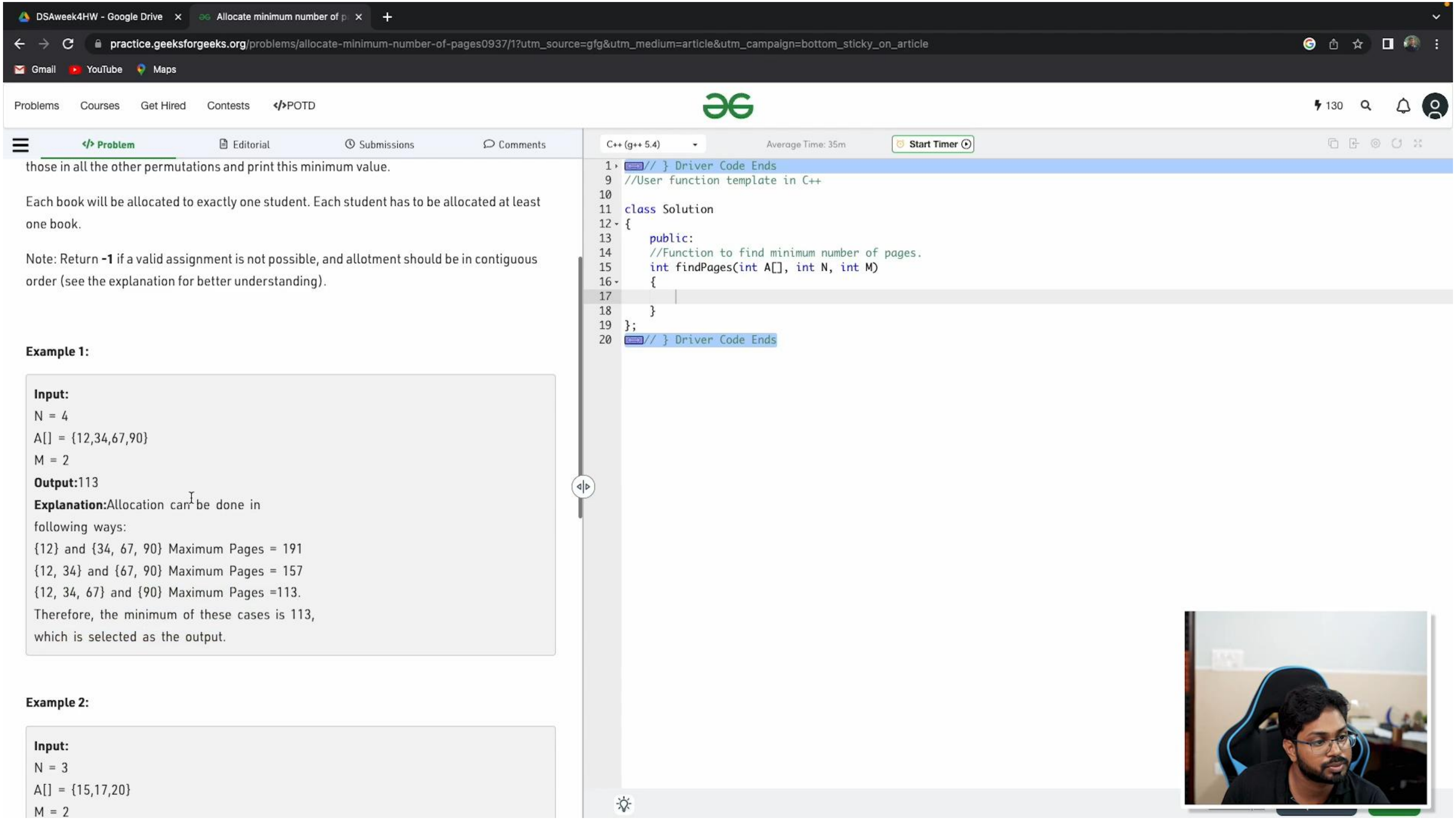
② $\overset{46}{12\ 34}$ | $\overset{157}{67\ 90}$ → max. 157

0 | 1

③ $\overset{113}{12\ 34\ 67}$ | $\overset{90}{90}$ → max. 113.

0 | 1

→ $\boxed{113}$


DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
C++ (g++ 5.4)
Average Time: 35m
Start Timer

those in all the other permutations and print this minimum value.

Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

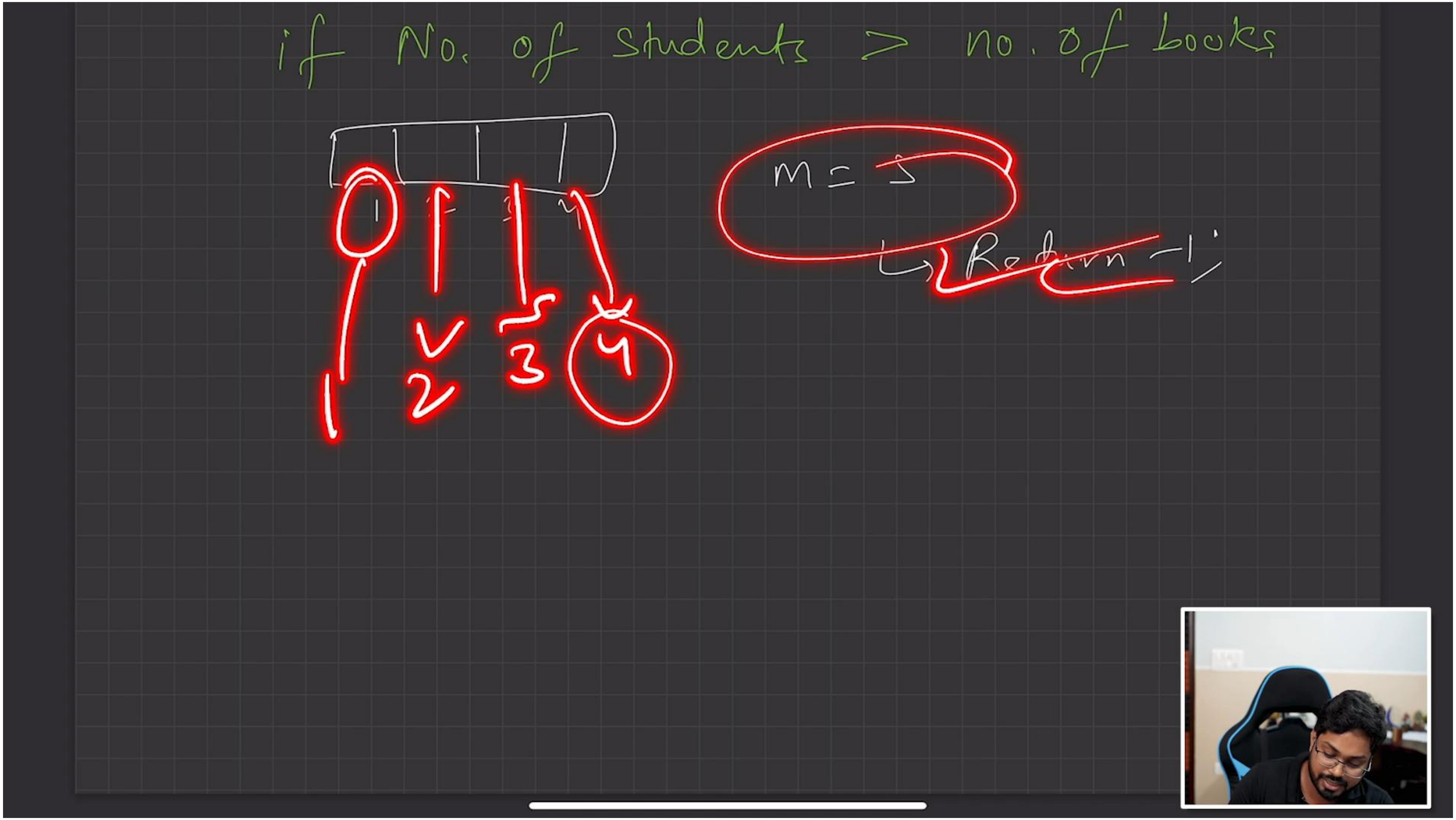
Example 1:

Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation: Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
{12, 34, 67} and {90} Maximum Pages =113.
Therefore, the minimum of these cases is 113,
which is selected as the output.

Example 2:

Input:
N = 3
A[] = {15,17,20}
M = 2

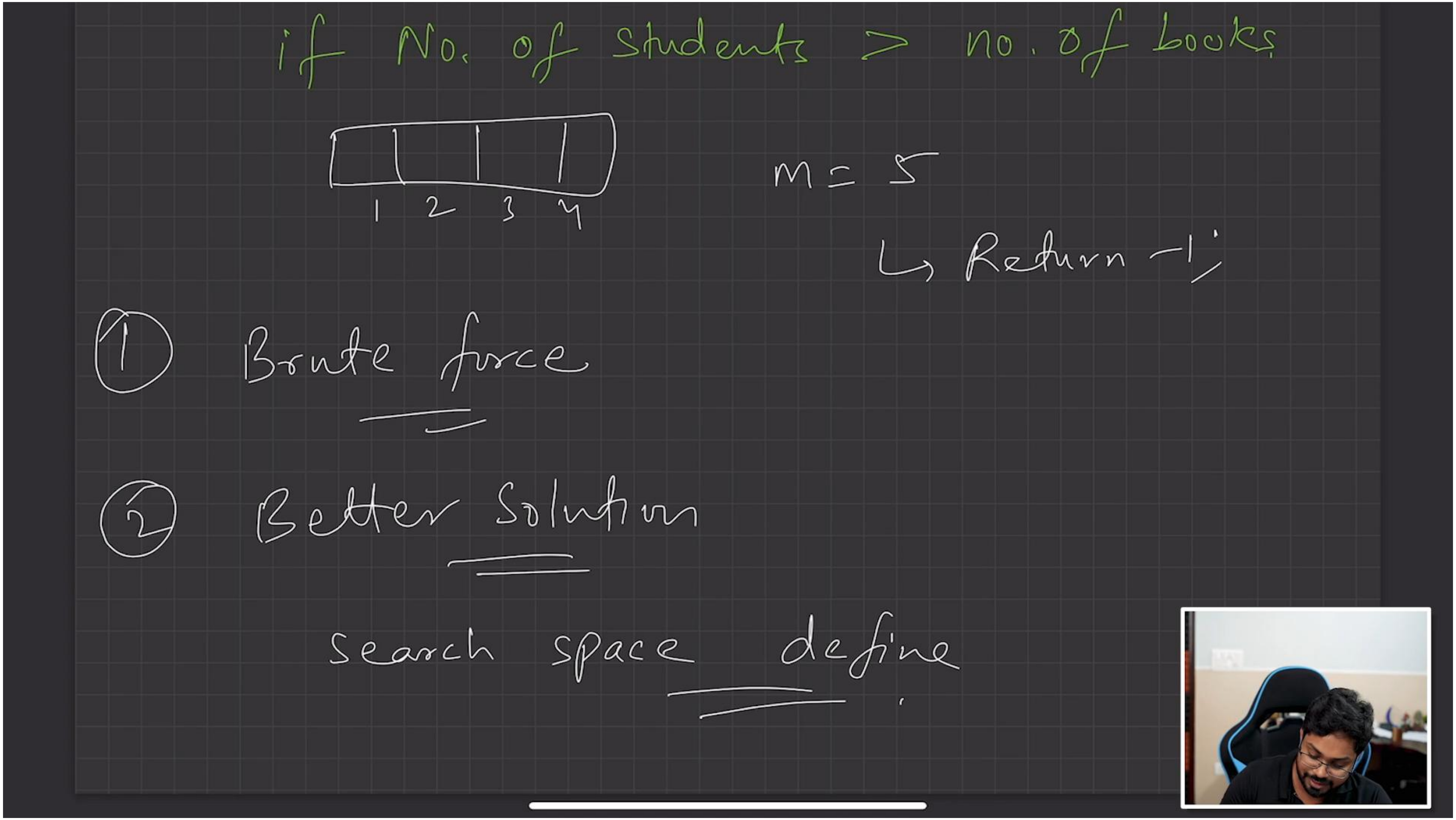
```
// } Driver Code Ends
//User function template in C++

class Solution
{
    public:
    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        
    }
};
// } Driver Code Ends
```



if No. of students > no. of books

m = 5

↳ Return -1;



if No. of students > no. of books

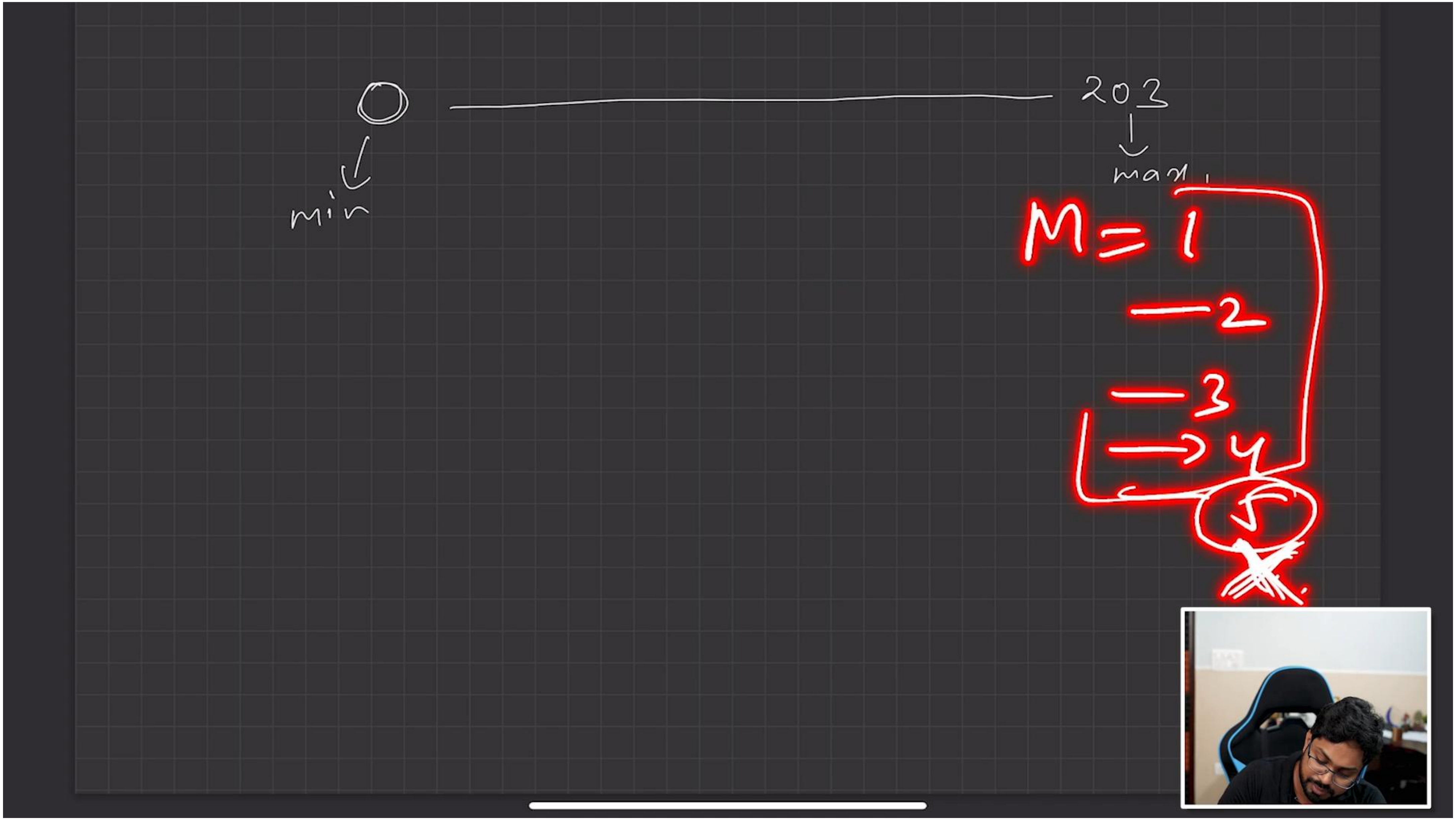
m = 5

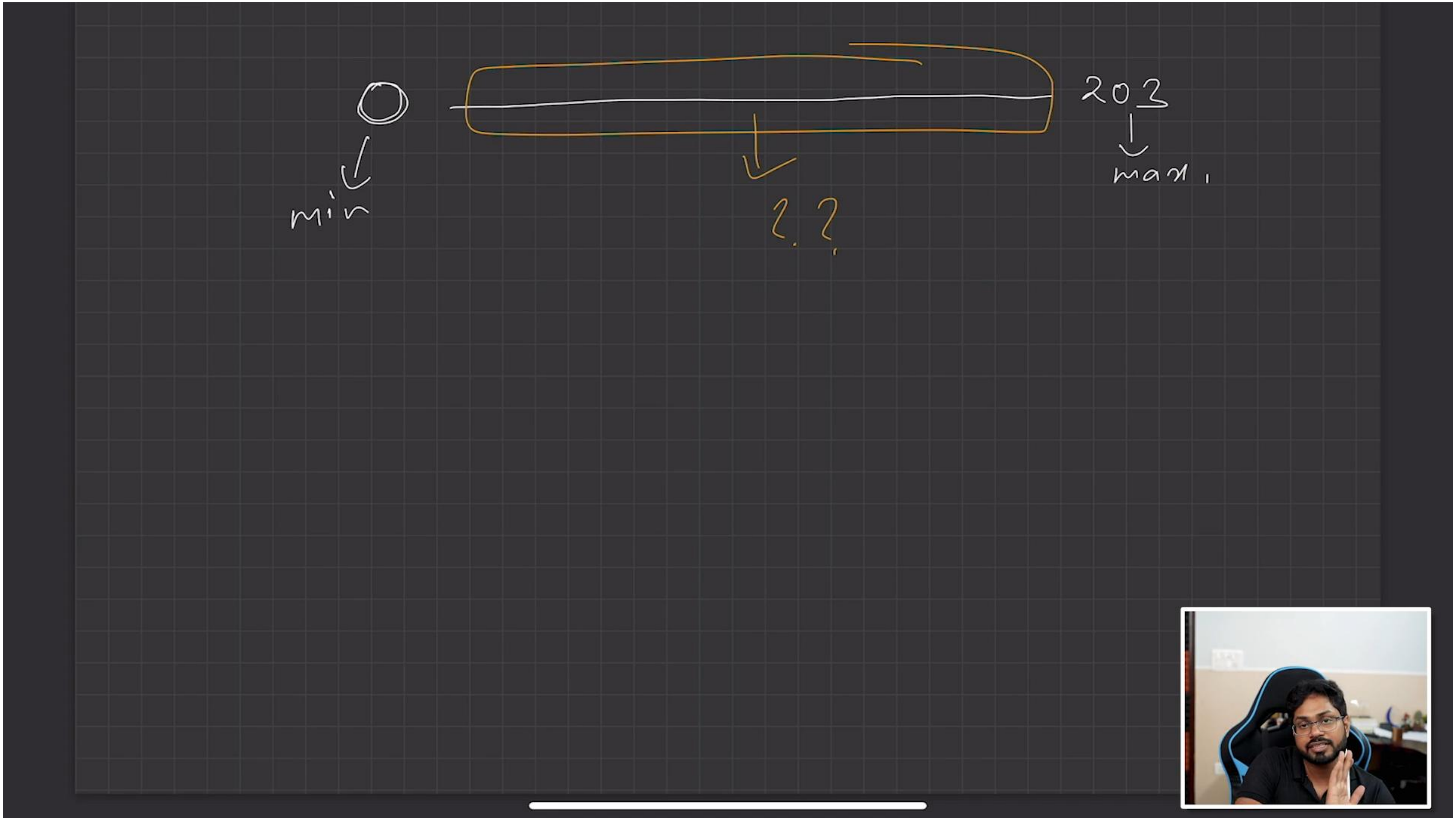
↳ Return -1;

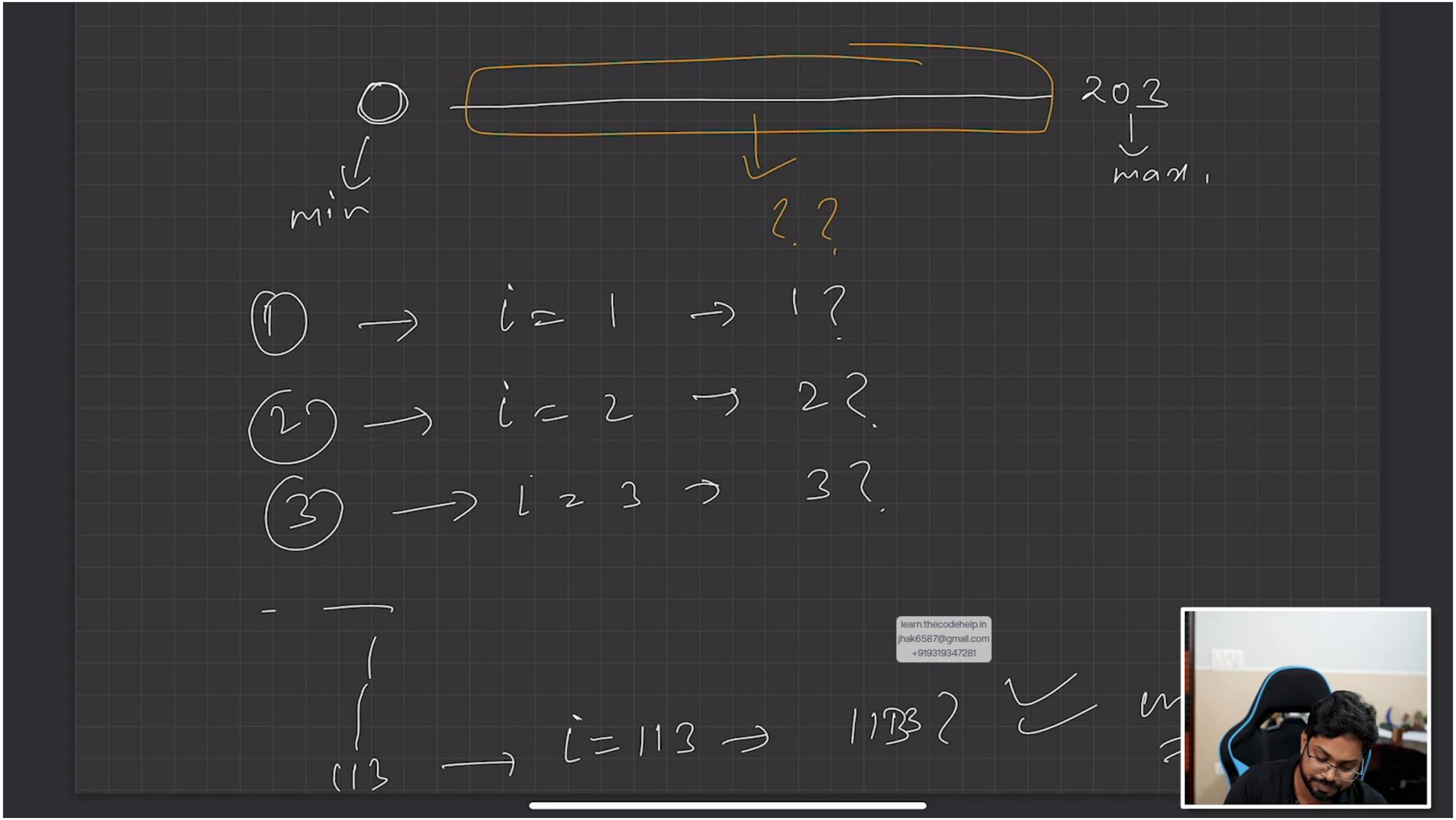
① Brute force

② Better Solution

search space define


0
203
min
max
M = 1
2
3
4
5
X


0
min
??
203
max.



0 → min

203 → max.

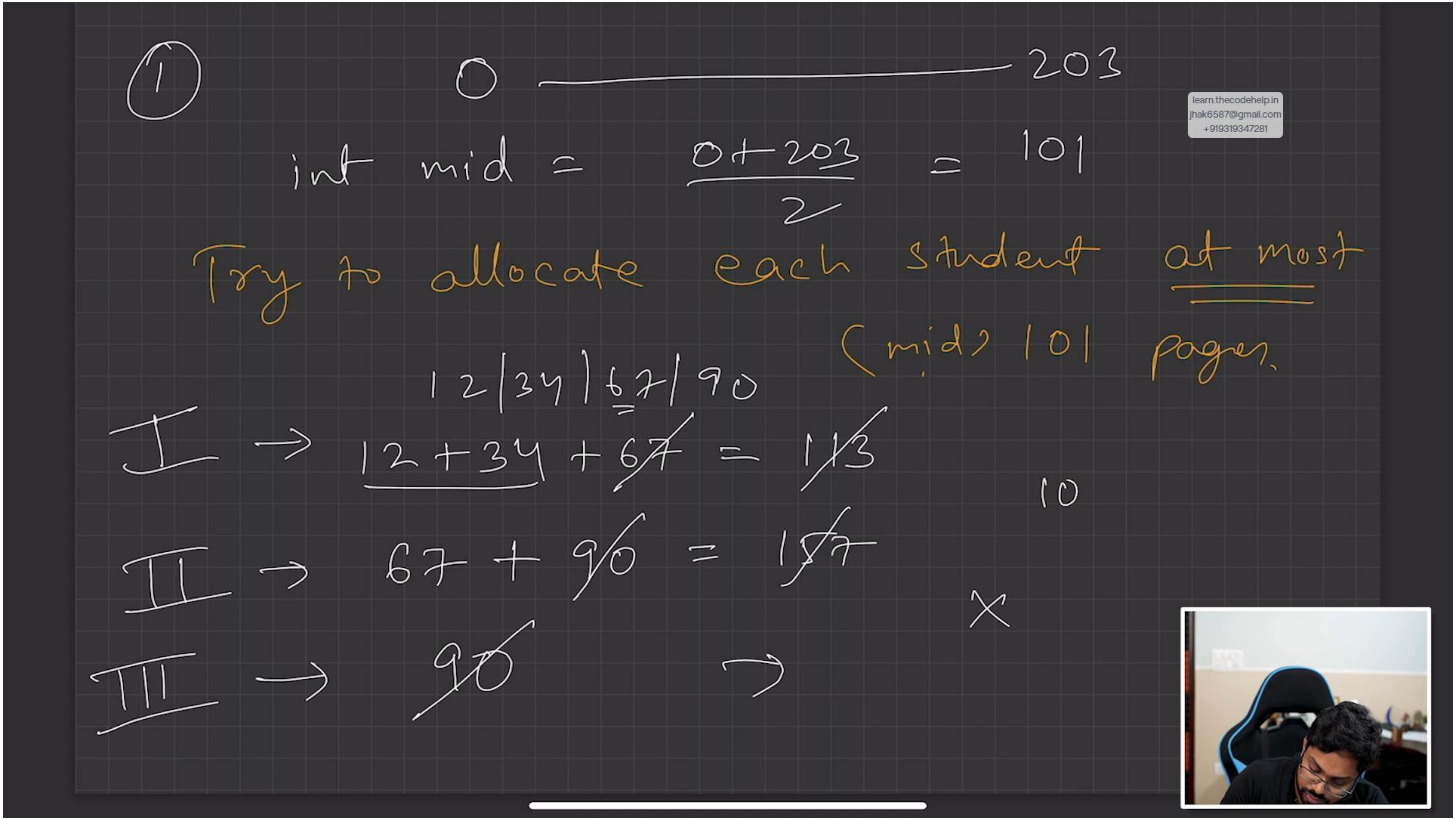
??

① → i = 1 → 1?

② → i = 2 → 2?

③ → i = 3 → 3?

113 → i = 113 → 113? ✓



① 0 ——— 203

$$\text{int mid} = \frac{0+203}{2} = 101$$

Try to allocate each student at most (mid) 101 pages.

12 | 34 | 67 | 90

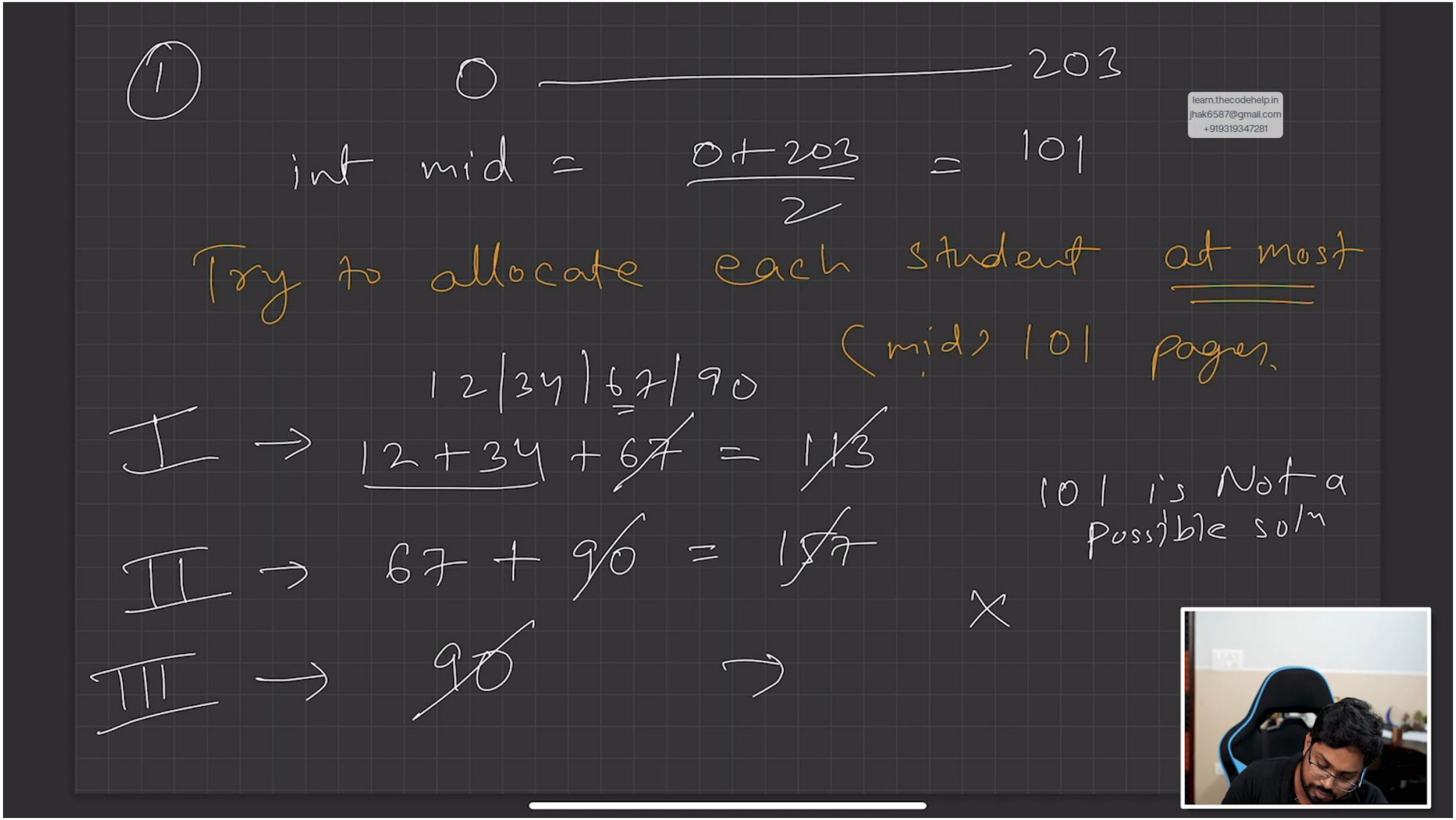
I → $\underline{12+34} + \cancel{67} = \cancel{113}$

10

II → $67 + \cancel{90} = \cancel{157}$

×

III → $\cancel{90}$ →



① 0 ———————— 203

int mid = $\frac{0+203}{2}$ = 101

Try to allocate each student at most (mid) 101 pages.

12 | 34 | 67 | 90

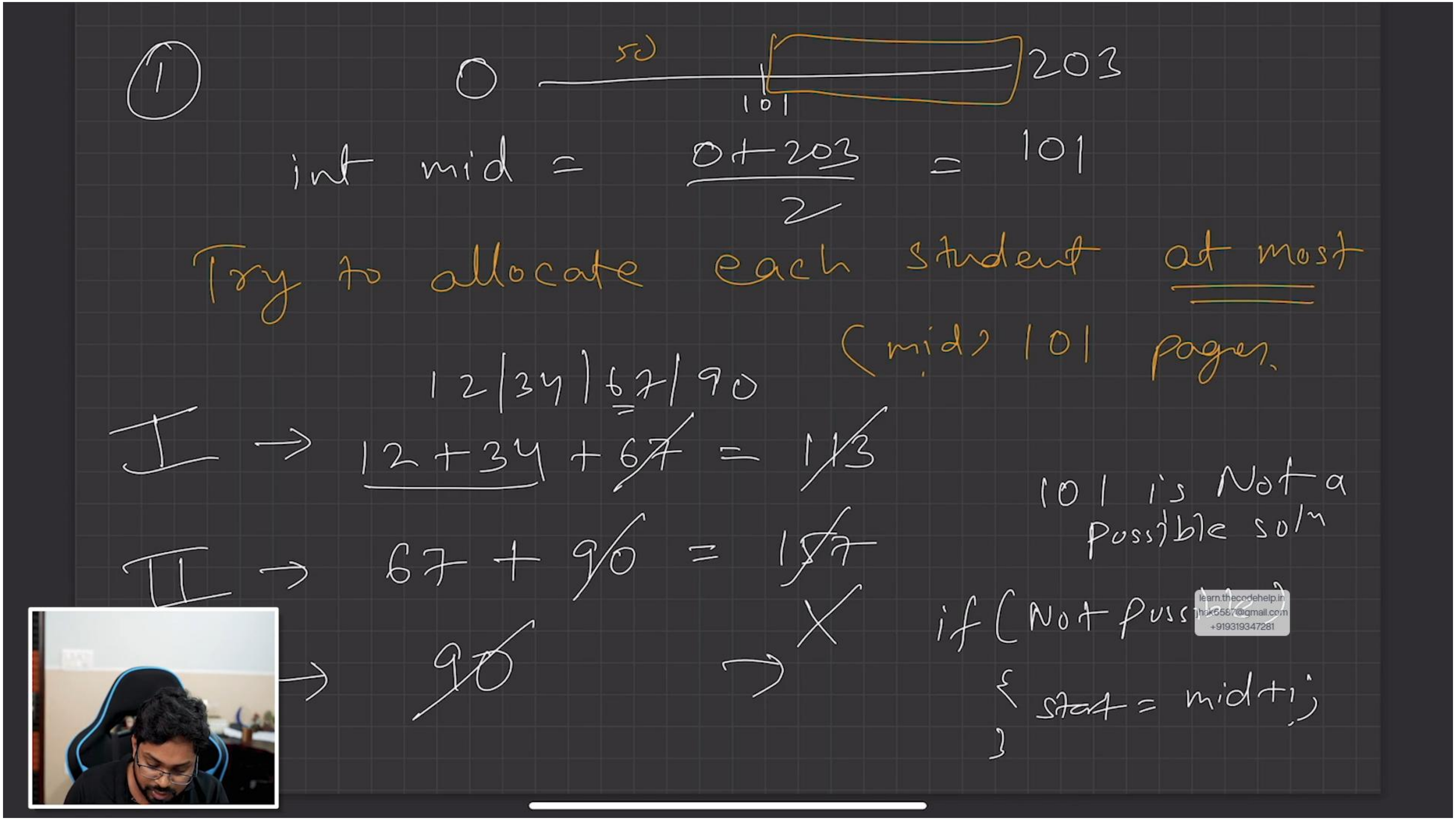
I → $\underline{12+34}+\cancel{67} = \cancel{113}$

II → $67+\cancel{90} = \cancel{157}$

III → $\cancel{90}$ →

×

101 is Not a possible solⁿ



int mid = $\frac{0+203}{2}$ = 101

Try to allocate each student at most (mid) 101 pages.

12 | 34 | 67 | 90

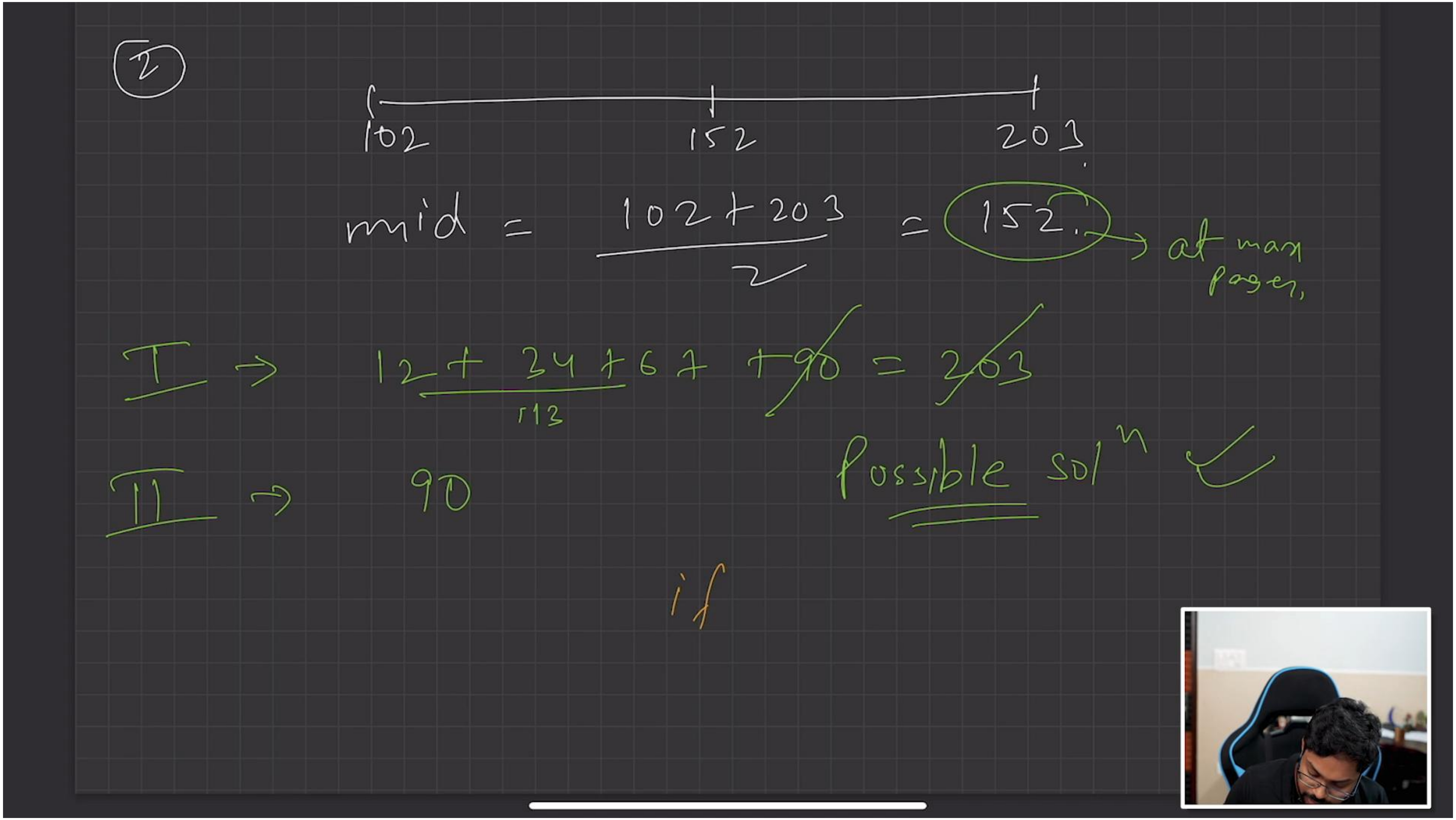
I → $12+34+\cancel{67} = \cancel{113}$

II → $67 + \cancel{90} = \cancel{157}$

→ $\cancel{90}$ → X

101 is Not a possible soln

```
if(Not possible)
{ start = mid+1;
}
```



$\boxed{2}$

102 ——— 152 ——— 203

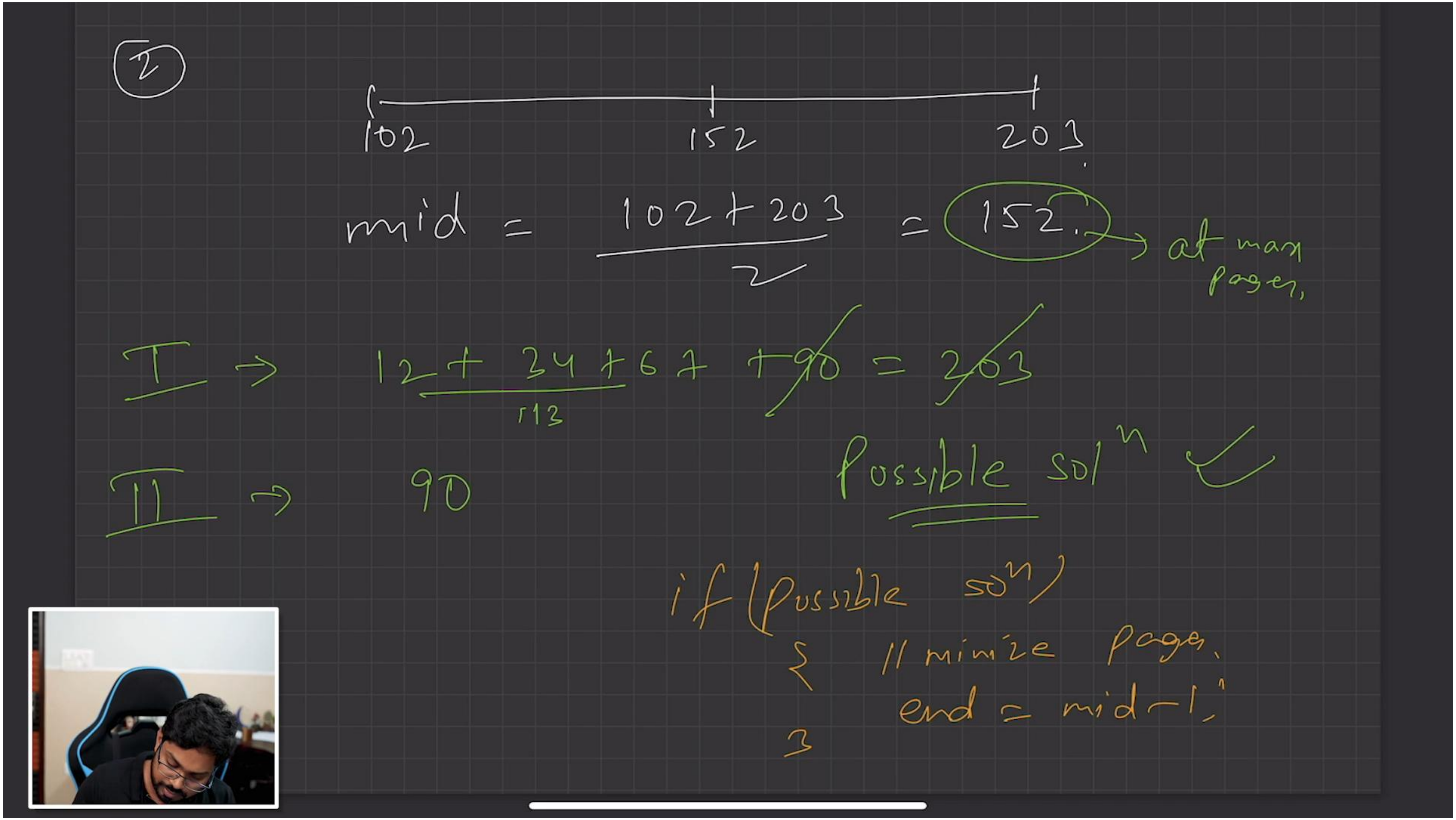
$$\text{mid} = \frac{102+203}{2} = \boxed{152} \rightarrow \text{at max pages.}$$

$$\text{I} \rightarrow \underbrace{12 + 34 + 67}_{113} + \cancel{90} = \cancel{203}$$

$$\text{II} \rightarrow 90$$

Possible solⁿ ✓

if



$\boxed{2}$

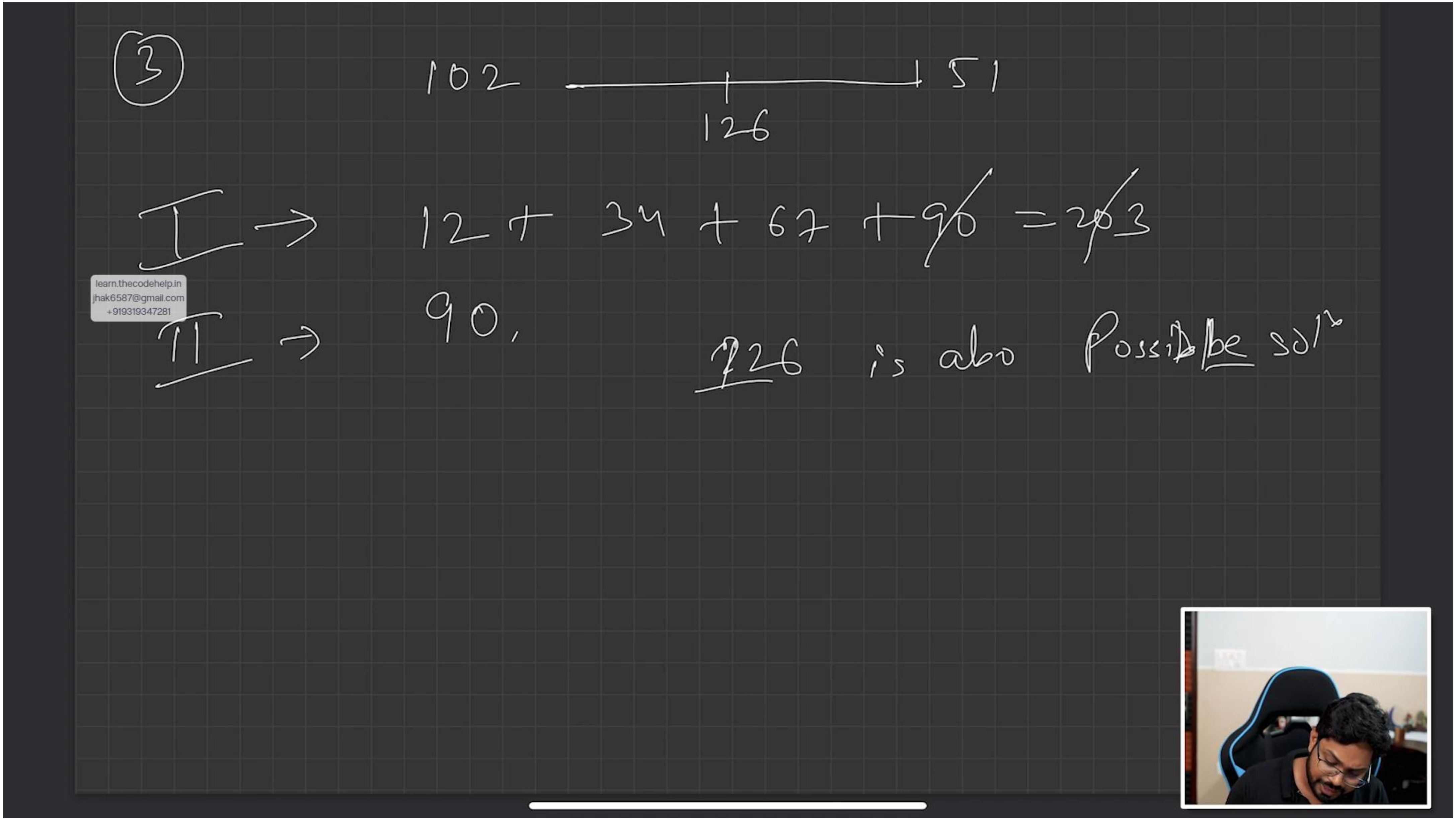
$$mid = \frac{102+203}{2} = \boxed{152.} \rightarrow \text{at max pages.}$$

$$I \rightarrow \underbrace{12+34+67}_{113} \cancel{+90} = \cancel{203}$$

$$II \rightarrow 90$$

Possible sol^n ✓

```
if(possible so^n)
{  // minize page.
   end = mid-1;
}
```



③

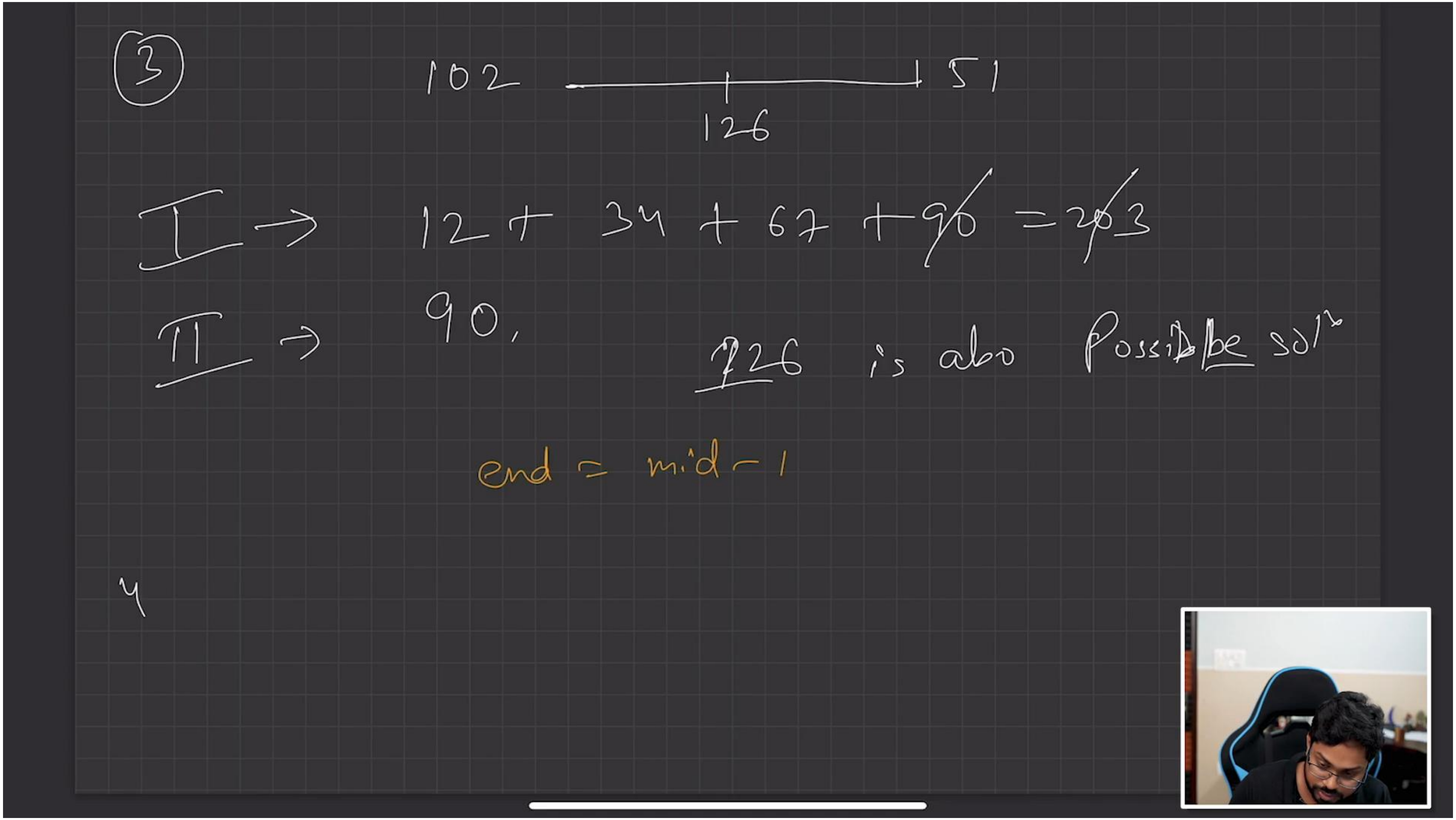
102 —|— 51

126

I → 12 + 34 + 67 + ~~90~~ = 2~~0~~3

II → 90,

226 is also Possible sol^n



3

102 ——|—— 51

126

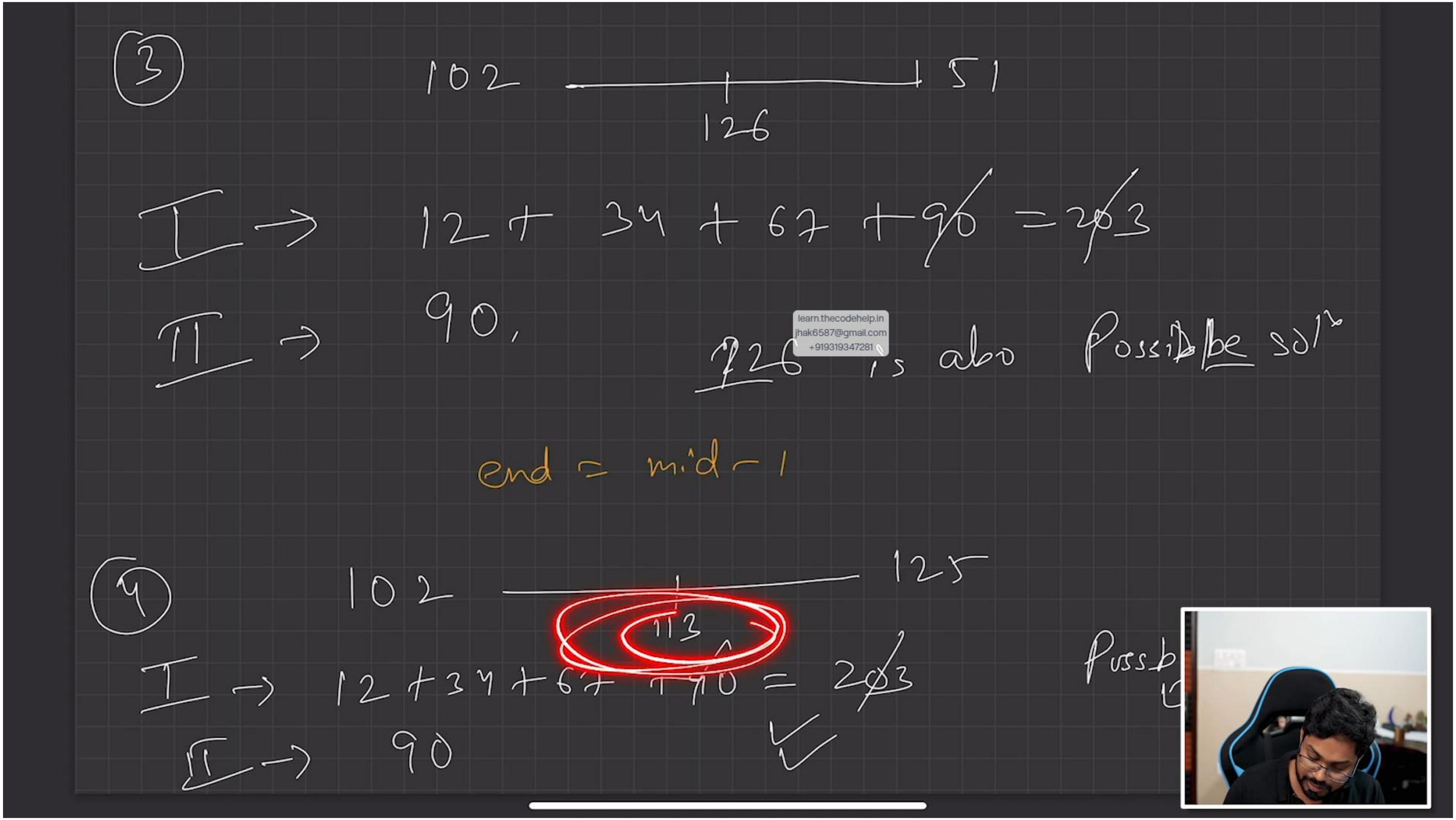
I → 12 + 34 + 67 + ~~90~~ = 2~~0~~3

II → 90,

126 is also Possible sol^n

end = mid - 1

4



③ 102 —————— 51

126

I → 12 + 34 + 67 + 90 = 203

II → 90,

126 is also Possible sol^n

end = mid - 1

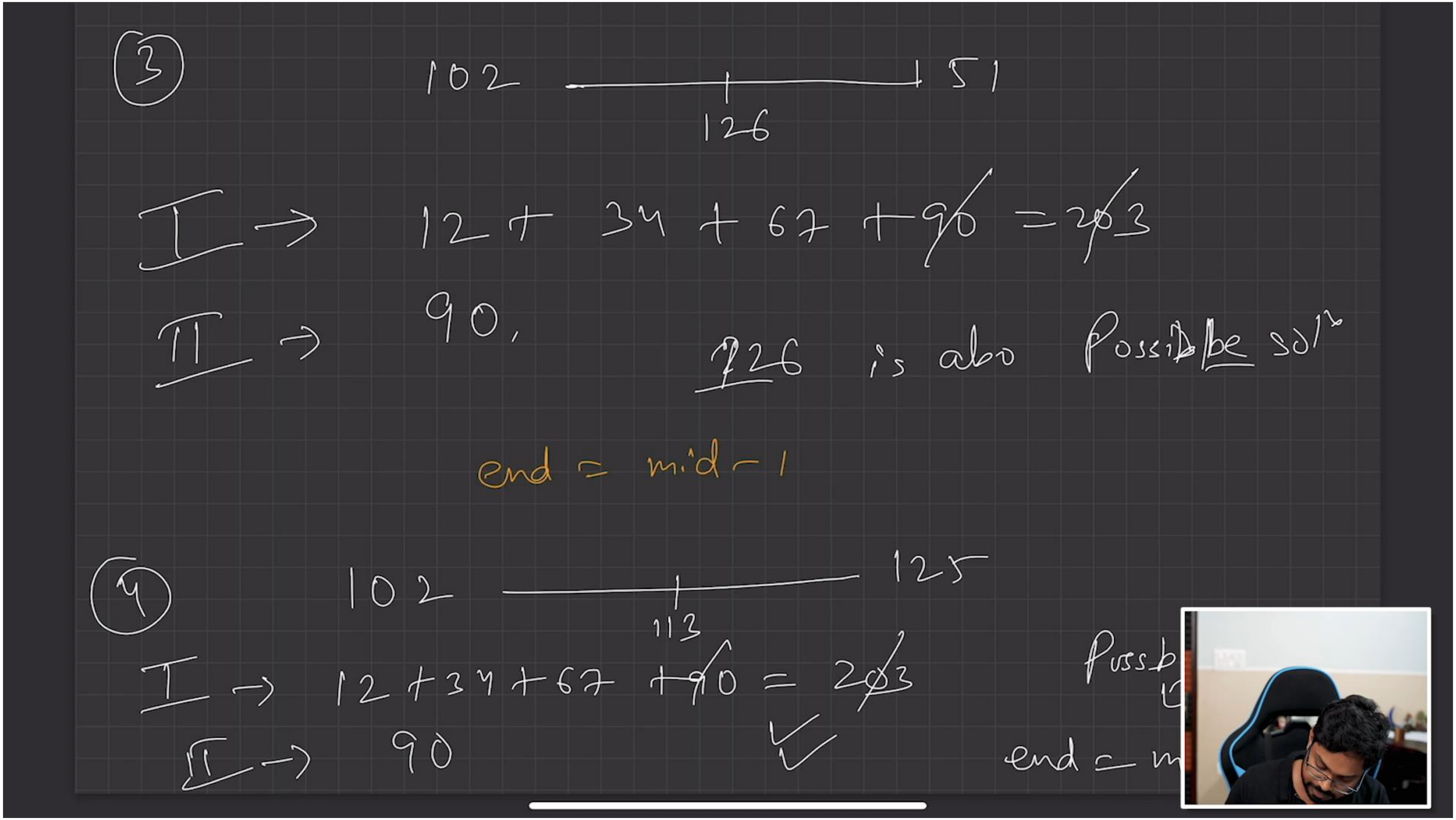
④ 102 —————— 125

113

I → 12 + 34 + 67 + 90 = 203 ✓✓

II → 90

Possib



I → $12 + 34 + 67 + \cancel{90} = 2\cancel{0}3$

II → 90,

126 is also Possible sol^n

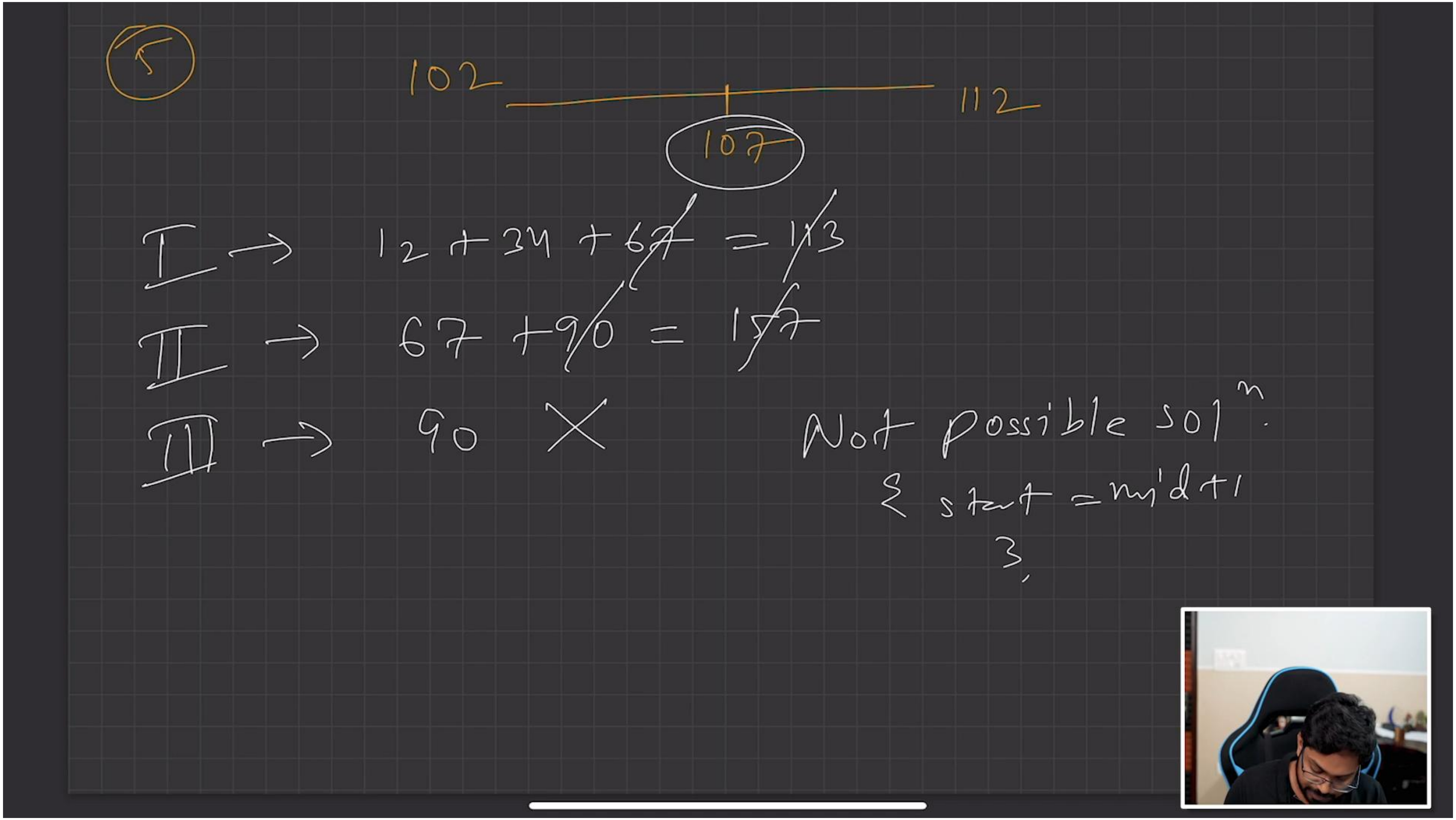
end = mid - 1

I → $12 + 34 + 67 + \cancel{90} = 2\cancel{0}3$ ✓✓

II → 90

Possib

end = m



I → 12 + 34 + ~~67~~ = ~~113~~

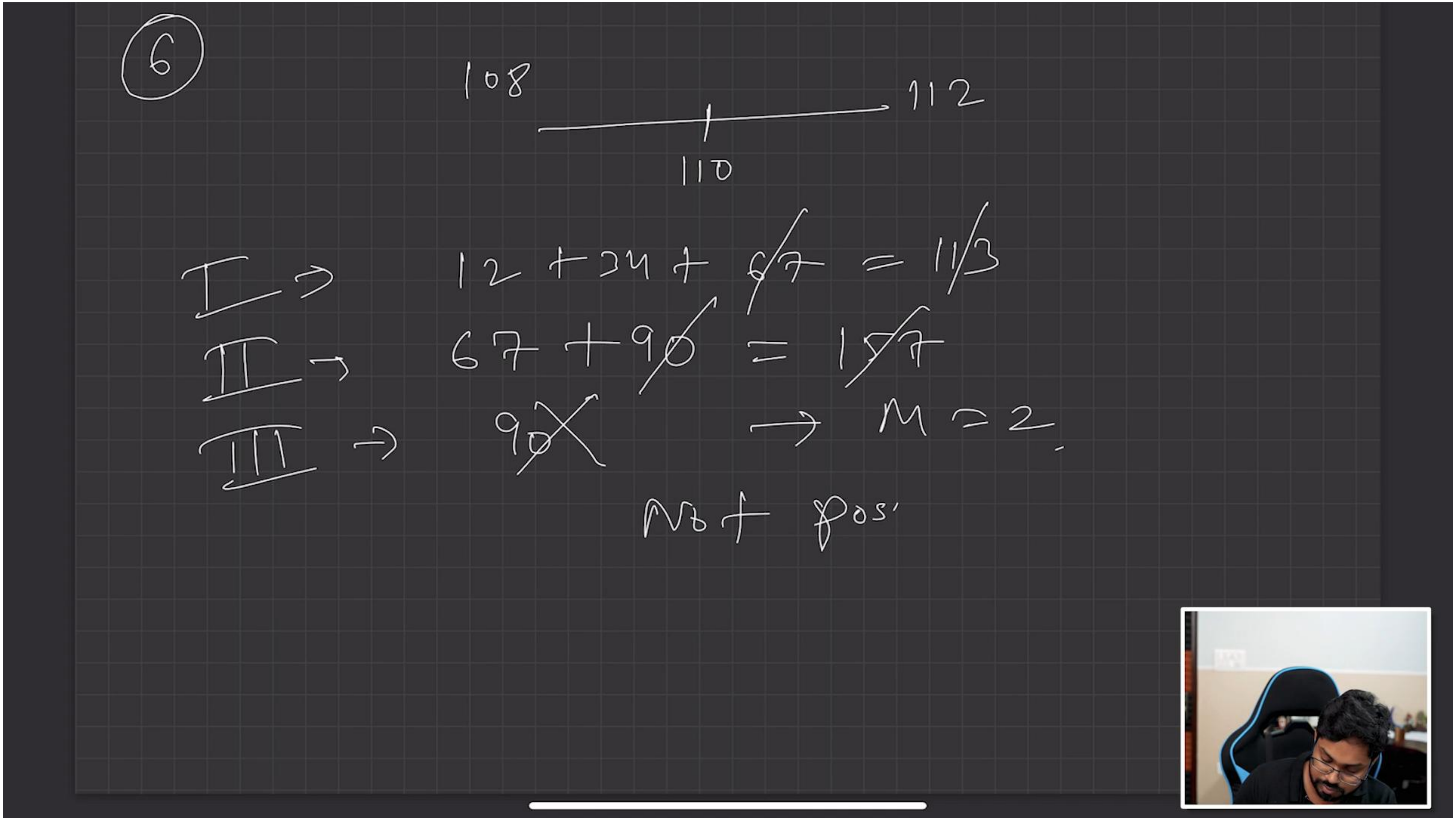
II → 67 + ~~90~~ = ~~157~~

III → 90 ✗

Not possible sol^n.

{ start = mid+1

3,



6

108 112

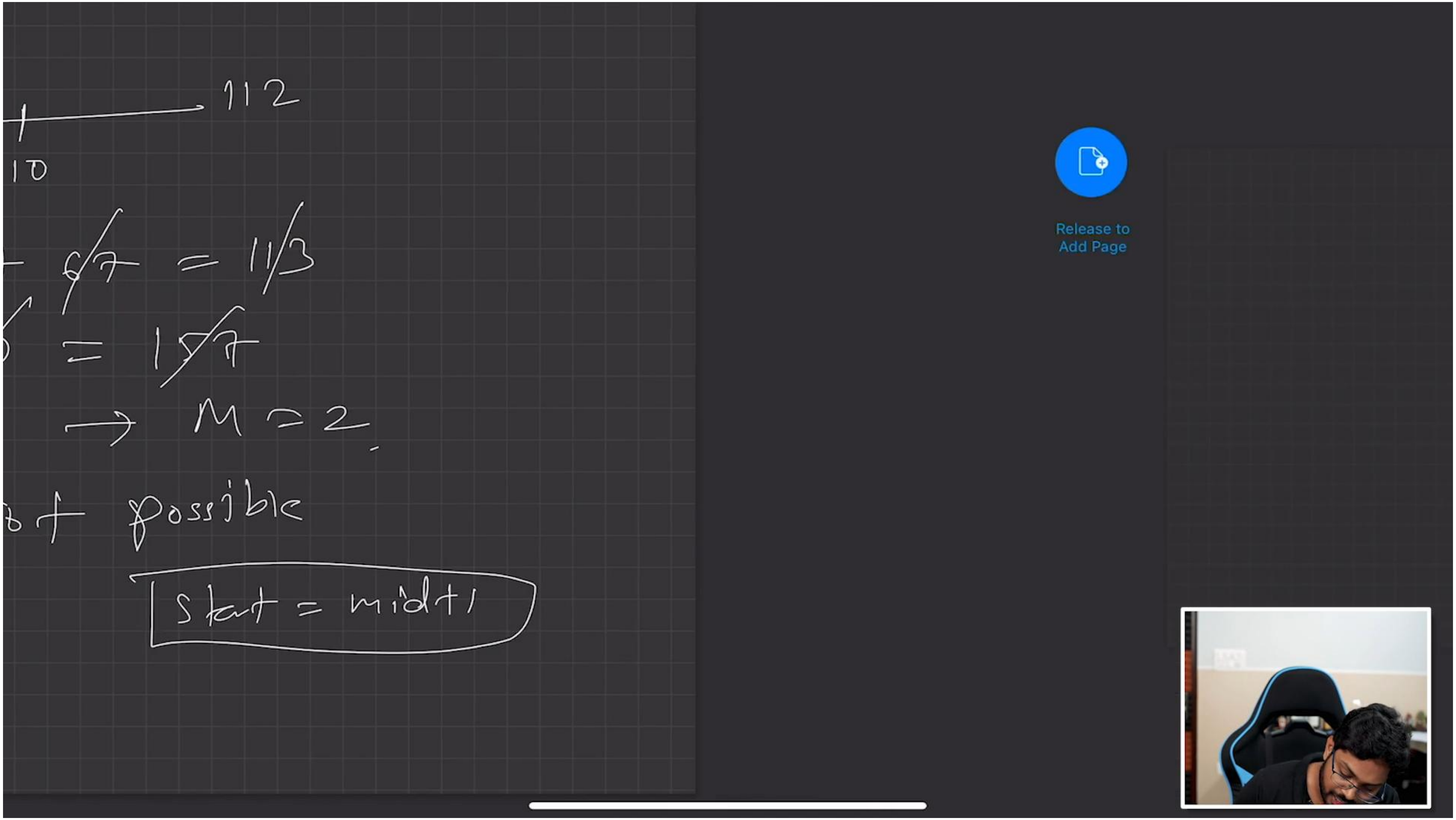
110

I → 12 + 34 + ~~67~~ = ~~113~~

II → 67 + ~~90~~ = ~~157~~

III → ~~90~~ → M = 2.

Not pos'



112

10

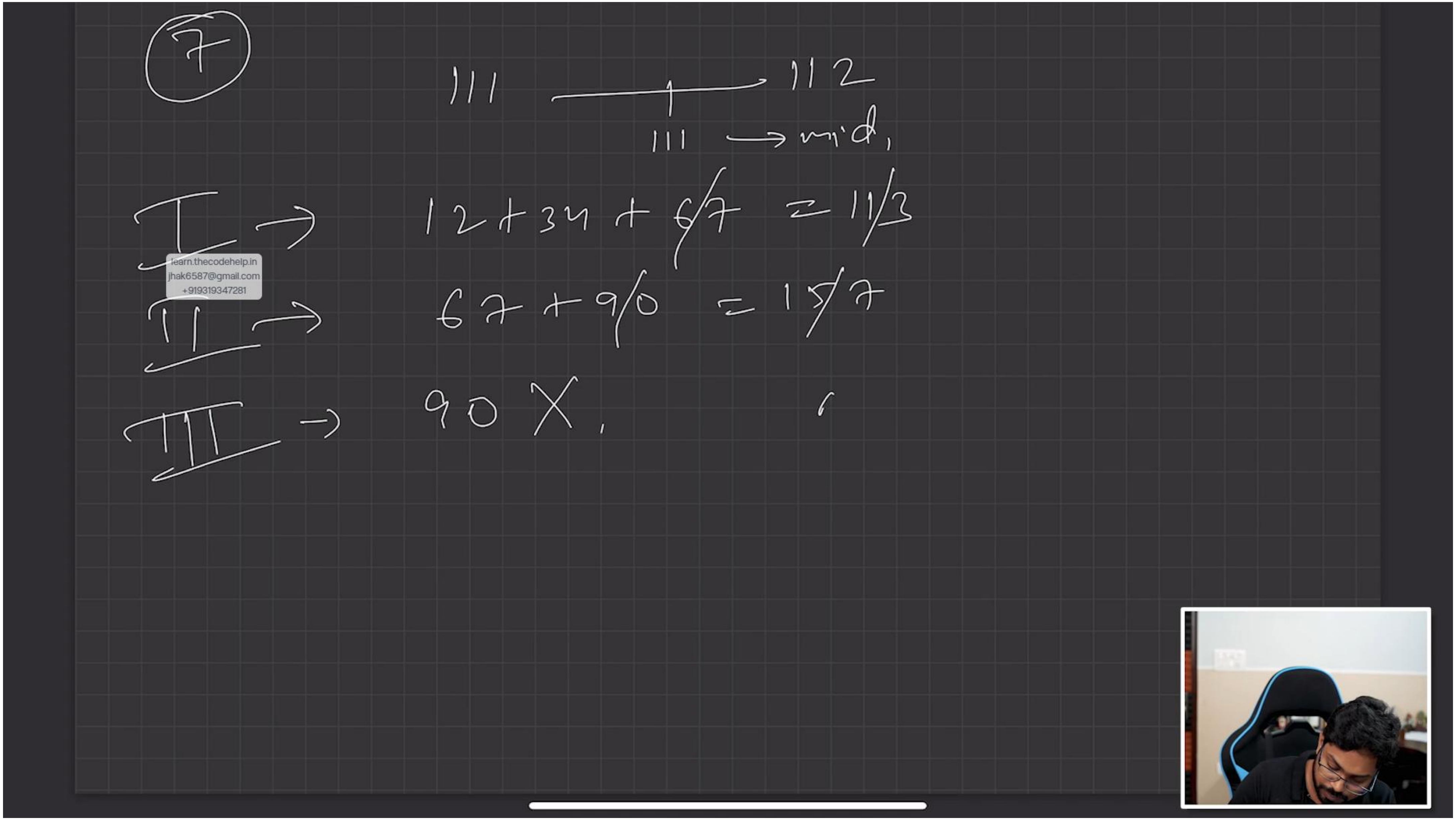
$\cancel{6}\cancel{7} = \cancel{11}\cancel{3}$

$= 1\cancel{5}\cancel{7}$

→ M = 2

t possible

$\boxed{\text{start} = \text{mid}+1}$



7

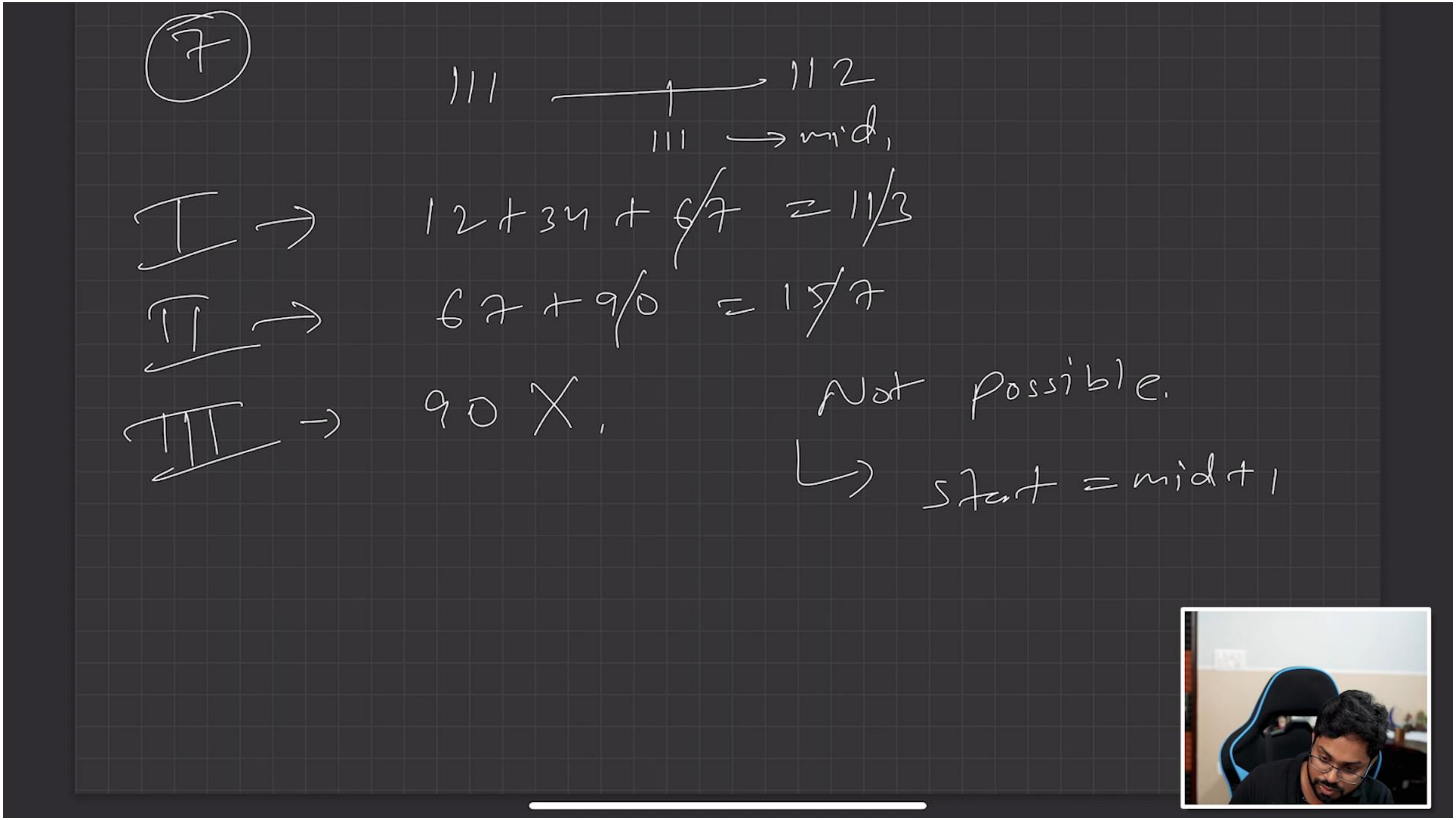
111 ——+——→ 112

111 → mid,

I → 12 + 34 + 6/7 = 11/3

II → 67 + 9/0 = 15/7

III → 90 X,



$\textcircled{7}$

111 ——+——→ 112

111 → mid,

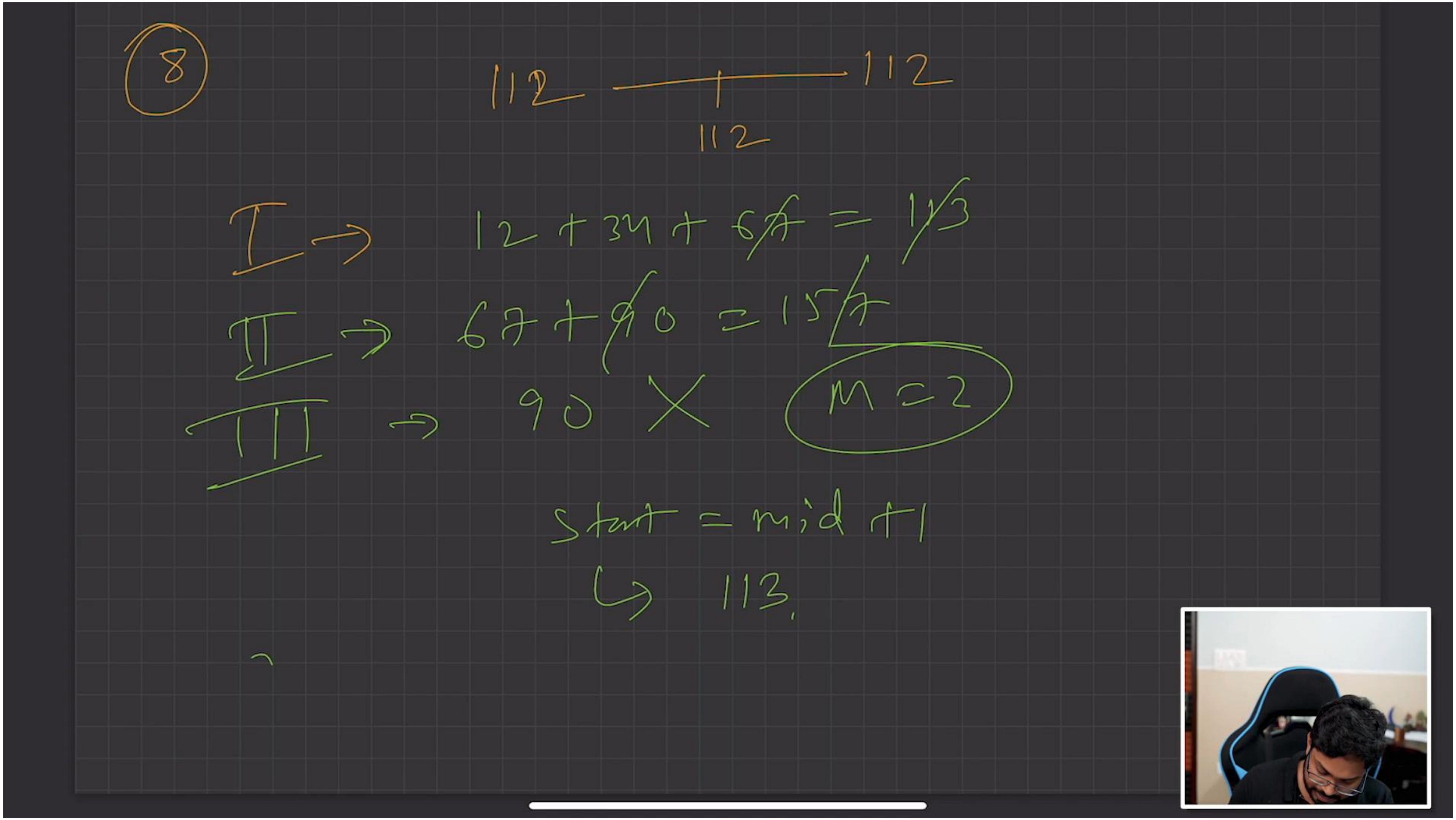
I → 12 + 34 + 6/7 = 11/3

II → 67 + 9/0 = 15/7

III → 90 X,

Not possible.

↳ start = mid + 1



8

112 ——┼—— 112

112

$\mathrm{I} \rightarrow 12 + 34 + \cancel{67} = \cancel{113}$

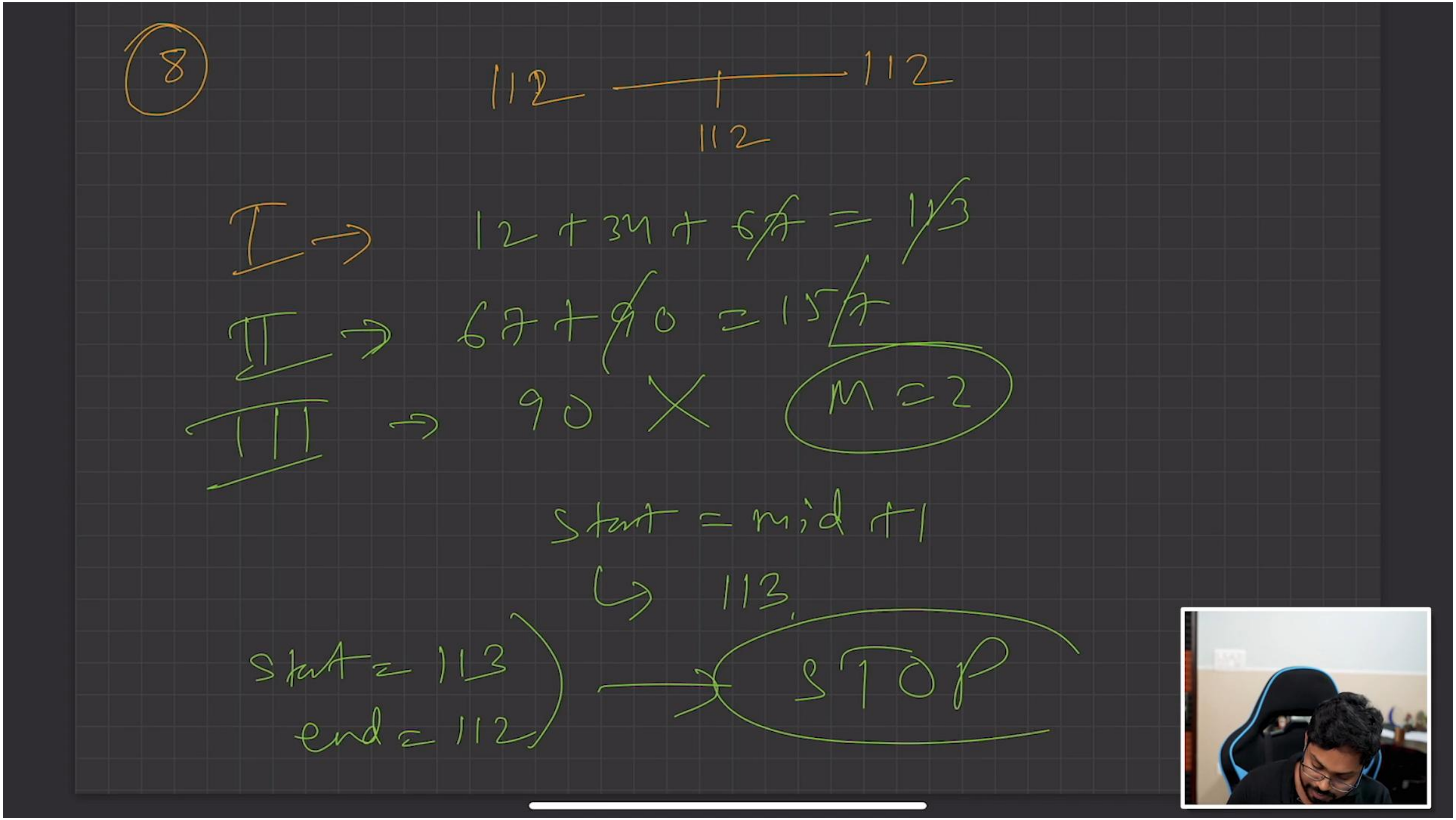
$\mathrm{II} \rightarrow 67 + \cancel{90} = \cancel{157}$

$\mathrm{III} \rightarrow 90$ ✗

$\boxed{M = 2}$

start = mid + 1

↳ 113.



8

112 —— 112

112

I → 12 + 34 + 67 = $\cancel{113}$

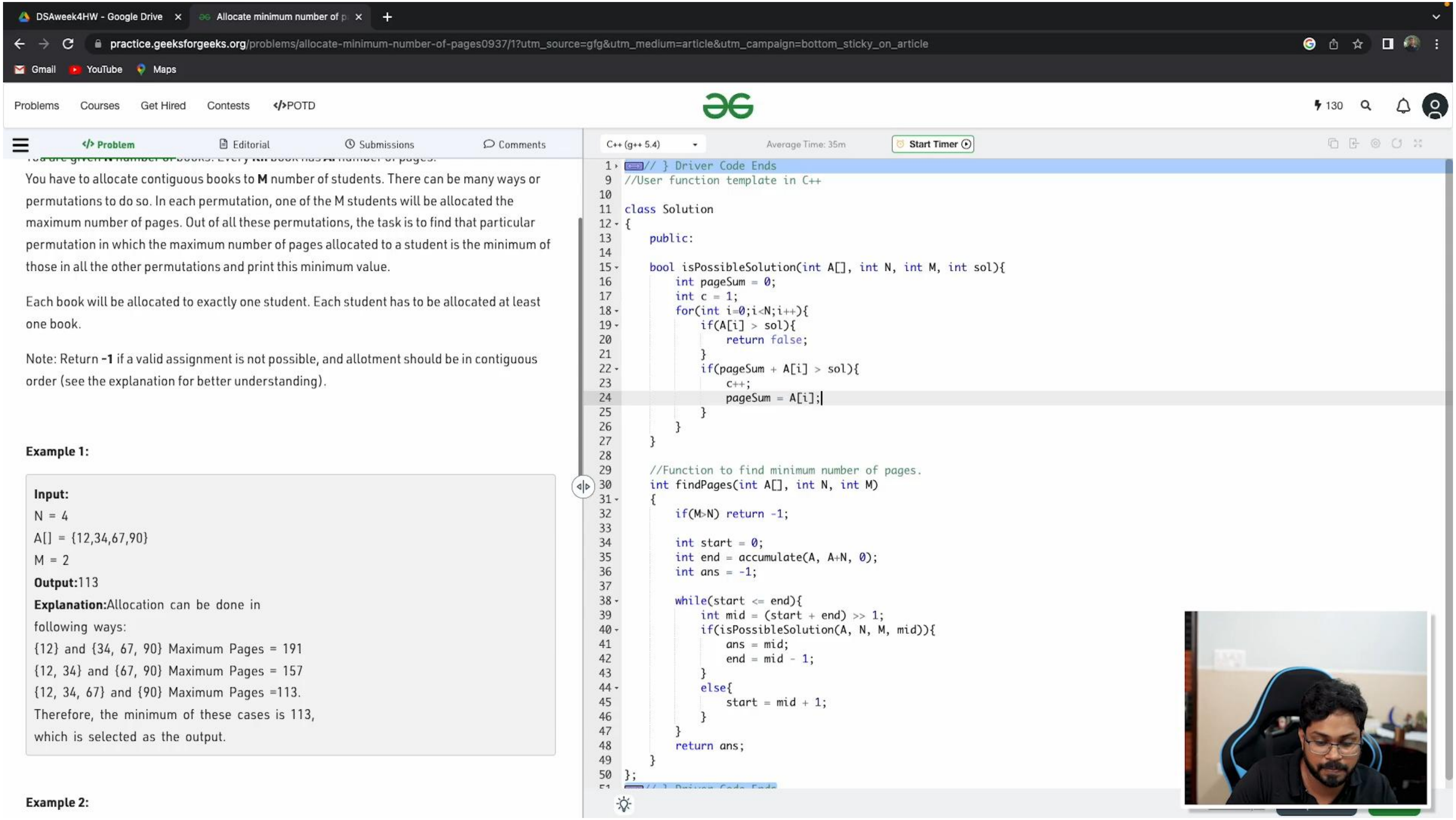
II → 67 + 90 = $\cancel{157}$

III → 90 ✗ M = 2

start = mid + 1

↳ 113

start = 113
end = 112 → STOP



DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
C++ (g++ 5.4)
Average Time: 35m
Start Timer

You are given N number of books. Every ith book has Ai number of pages.

You have to allocate contiguous books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is the minimum of those in all the other permutations and print this minimum value.

Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

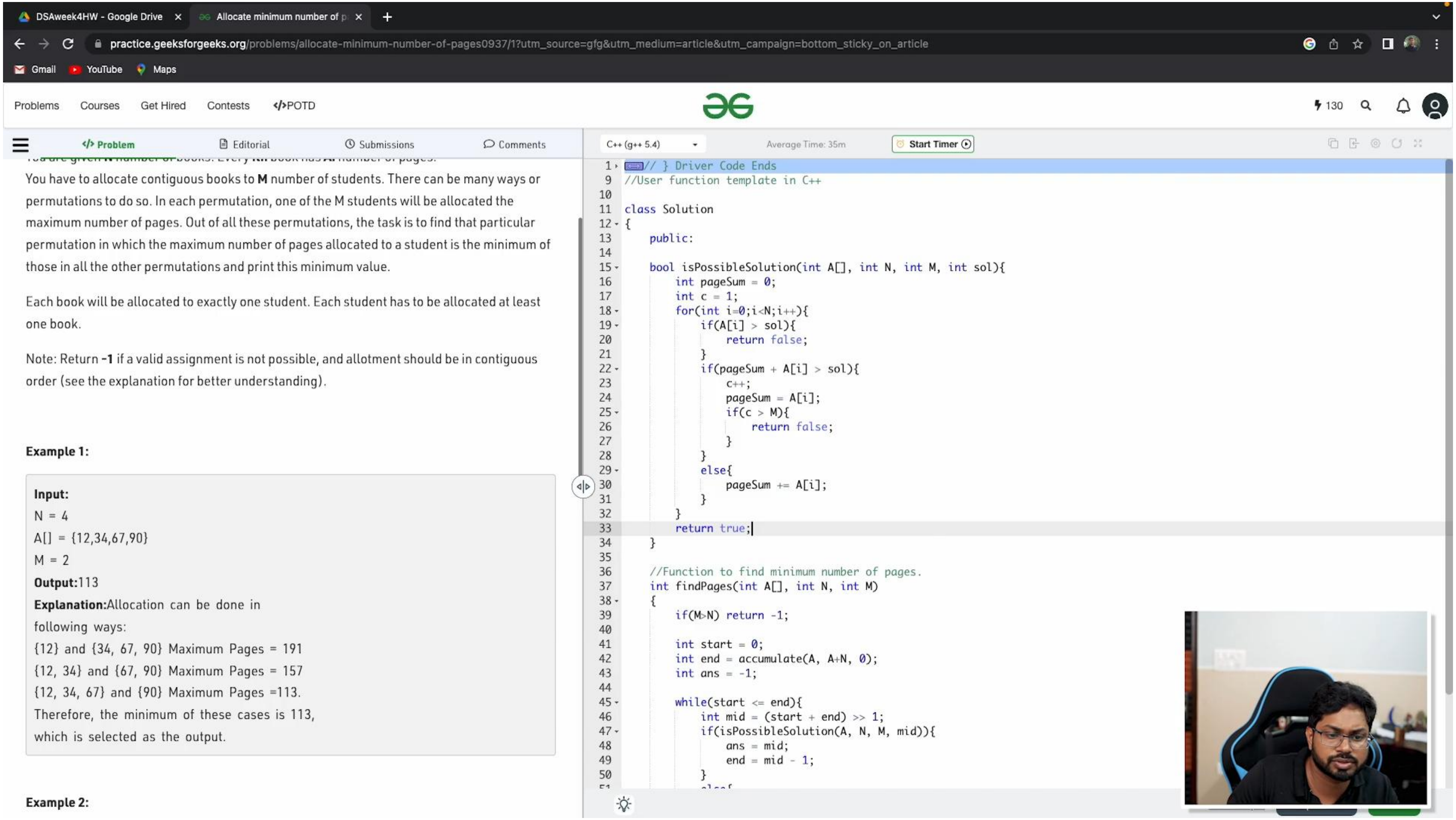
Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation: Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
{12, 34, 67} and {90} Maximum Pages =113.
Therefore, the minimum of these cases is 113,
which is selected as the output.

Example 2:

```
// } Driver Code Ends
//User function template in C++

class Solution
{
    public:

    bool isPossibleSolution(int A[], int N, int M, int sol){
        int pageSum = 0;
        int c = 1;
        for(int i=0;i<N;i++){
            if(A[i] > sol){
                return false;
            }
            if(pageSum + A[i] > sol){
                c++;
                pageSum = A[i];
            }
        }
    }

    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        if(M>N) return -1;

        int start = 0;
        int end = accumulate(A, A+N, 0);
        int ans = -1;

        while(start <= end){
            int mid = (start + end) >> 1;
            if(isPossibleSolution(A, N, M, mid)){
                ans = mid;
                end = mid - 1;
            }
            else{
                start = mid + 1;
            }
        }
        return ans;
    }
};
// } Driver Code Ends
```


DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
C++ (g++ 5.4)
Average Time: 35m
Start Timer

You are given N number of books. Every ith book has Ai number of pages.

You have to allocate contiguous books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is the minimum of those in all the other permutations and print this minimum value.

Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

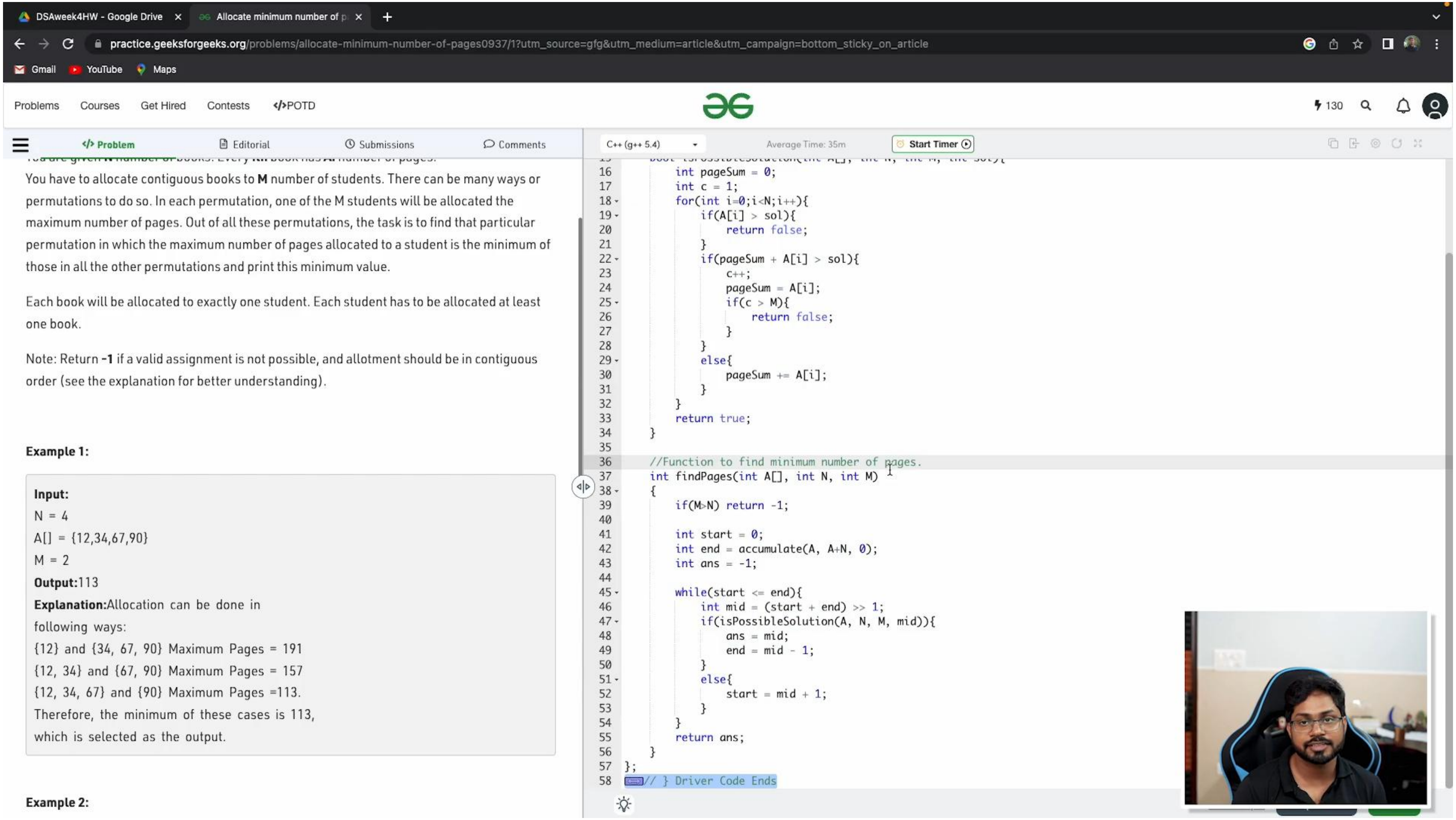
Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

```
Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation:Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
{12, 34, 67} and {90} Maximum Pages =113.
Therefore, the minimum of these cases is 113,
which is selected as the output.
```

Example 2:

```cpp
// } Driver Code Ends
//User function template in C++

class Solution
{
    public:

    bool isPossibleSolution(int A[], int N, int M, int sol){
        int pageSum = 0;
        int c = 1;
        for(int i=0;i<N;i++){
            if(A[i] > sol){
                return false;
            }
            if(pageSum + A[i] > sol){
                c++;
                pageSum = A[i];
                if(c > M){
                    return false;
                }
            }
            else{
                pageSum += A[i];
            }
        }
        return true;
    }

    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        if(M>N) return -1;

        int start = 0;
        int end = accumulate(A, A+N, 0);
        int ans = -1;

        while(start <= end){
            int mid = (start + end) >> 1;
            if(isPossibleSolution(A, N, M, mid)){
                ans = mid;
                end = mid - 1;
            }
            else{
```


DSAweek4HW - Google Drive
Allocate minimum number of p
practice.geeksforgeeks.org/problems/allocate-minimum-number-of-pages0937/1?utm_source=gfg&utm_medium=article&utm_campaign=bottom_sticky_on_article
Gmail
YouTube
Maps
Problems
Courses
Get Hired
Contests
</>POTD
130
</> Problem
Editorial
Submissions
Comments
C++ (g++ 5.4)
Average Time: 35m
Start Timer

You are given N number of books. Every ith book has Ai number of pages.

You have to allocate contiguous books to **M** number of students. There can be many ways or permutations to do so. In each permutation, one of the M students will be allocated the maximum number of pages. Out of all these permutations, the task is to find that particular permutation in which the maximum number of pages allocated to a student is the minimum of those in all the other permutations and print this minimum value.

Each book will be allocated to exactly one student. Each student has to be allocated at least one book.

Note: Return **-1** if a valid assignment is not possible, and allotment should be in contiguous order (see the explanation for better understanding).

Example 1:

Input:
N = 4
A[] = {12,34,67,90}
M = 2
Output:113
Explanation: Allocation can be done in
following ways:
{12} and {34, 67, 90} Maximum Pages = 191
{12, 34} and {67, 90} Maximum Pages = 157
{12, 34, 67} and {90} Maximum Pages =113.
Therefore, the minimum of these cases is 113,
which is selected as the output.

Example 2:

```
    bool isPossibleSolution(int A[], int N, int M, int sol){
        int pageSum = 0;
        int c = 1;
        for(int i=0;i<N;i++){
            if(A[i] > sol){
                return false;
            }
            if(pageSum + A[i] > sol){
                c++;
                pageSum = A[i];
                if(c > M){
                    return false;
                }
            }
            else{
                pageSum += A[i];
            }
        }
        return true;
    }

    //Function to find minimum number of pages.
    int findPages(int A[], int N, int M)
    {
        if(M>N) return -1;

        int start = 0;
        int end = accumulate(A, A+N, 0);
        int ans = -1;

        while(start <= end){
            int mid = (start + end) >> 1;
            if(isPossibleSolution(A, N, M, mid)){
                ans = mid;
                end = mid - 1;
            }
            else{
                start = mid + 1;
            }
        }
        return ans;
    }
};
// } Driver Code Ends
```