Searching and Sorting Class - III

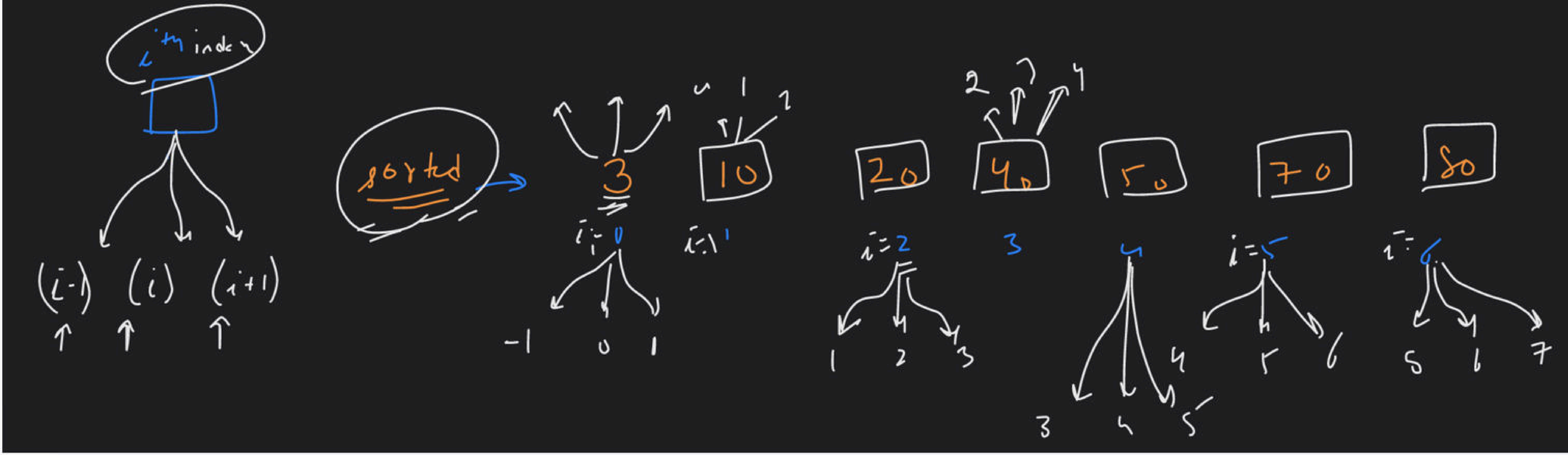
Special class

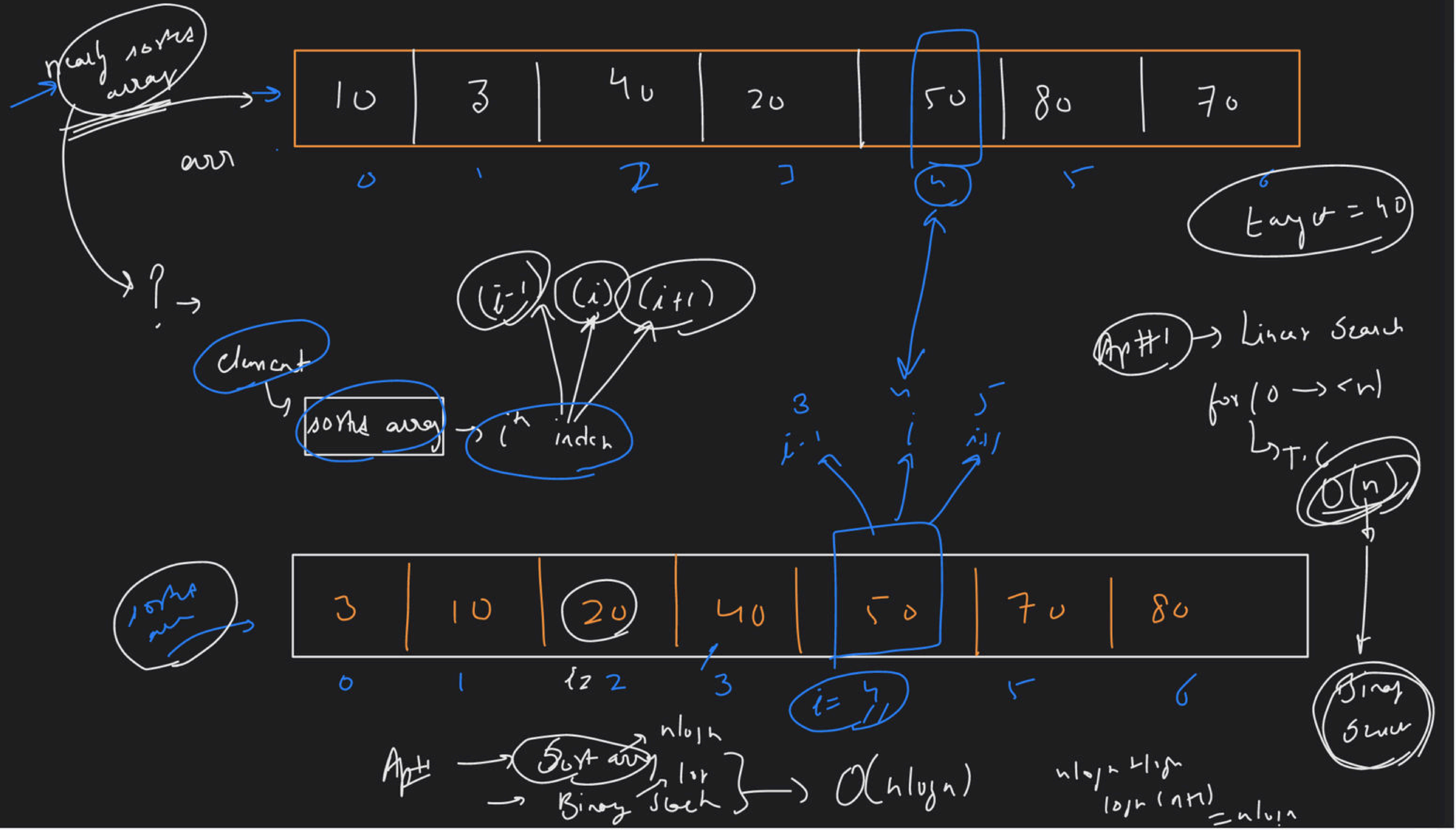
Love Babbar • Feb 19, 2023

Binary Search in a nearly Sorted Array :-

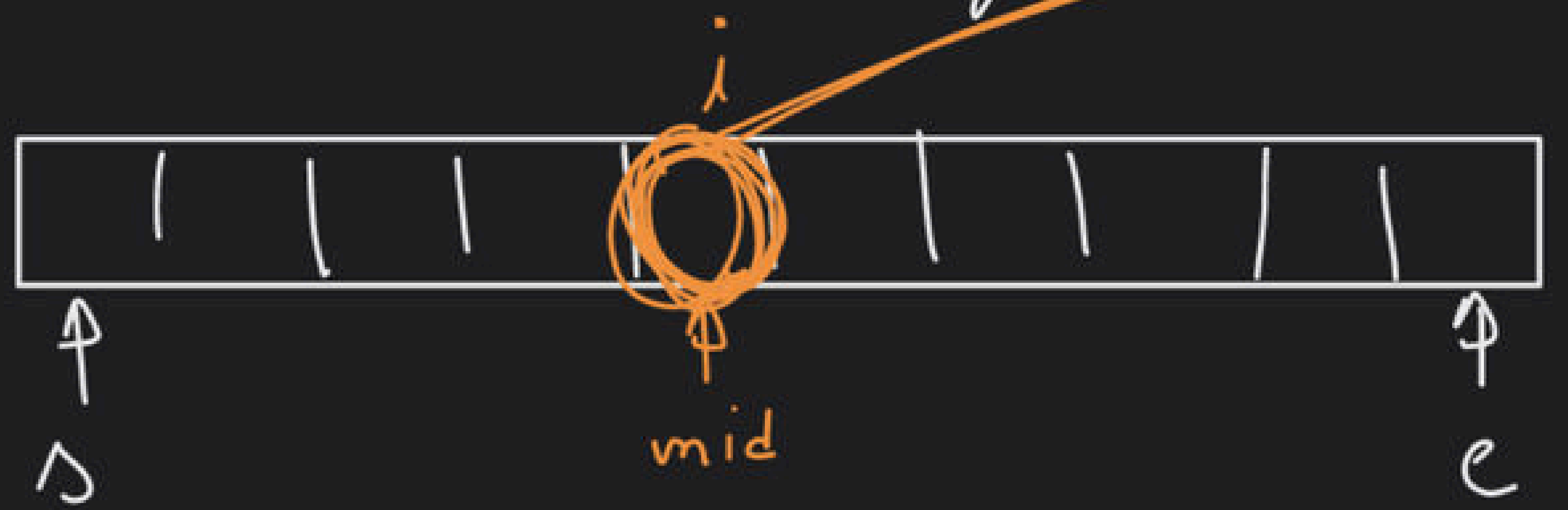
arr

10	3	40	20	50	80	70
0	1	2	3	4	5	6

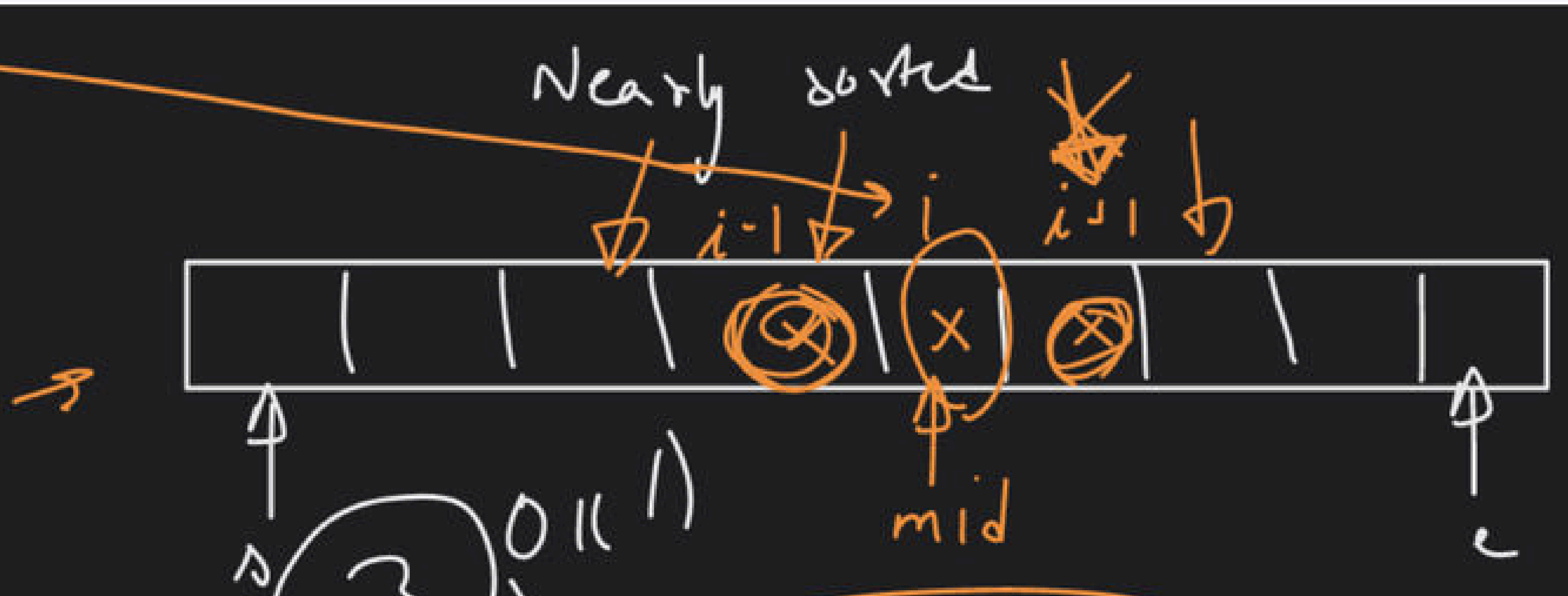



nearly sorted array
arr
10
3
40
20
50
80
70
0
1
2
3
4
5
6
target = 40
?→
element
sorted array
i th index
(i-1)
(i)
(i+1)
3
4
5
i-1
i
i+1
Ap#1 → Linear Search
for (0 → <n)
↳ T.C
O(n)
Binary Search
sorted arr
3
10
20
40
50
70
80
0
1
2
3
i = 4
5
6
Ap#2 → Sort arr → nlogn
for
→ Binary Search
→ O(nlogn)
nlogn + logn
logn (n+1)
= nlogn

Sorted Array



Nearly sorted



3 O(1)

```
if (arr[mid] == target)
    return mid;
```

O(1)

O(1)

```
if (target > arr[mid])
    └→ R.S → s = mid+1;

else
    └→ L.S → e = mid-1;
```

T.C O(logn)

3×O(1)
= O(1)

~~mid-1 >= 0~~

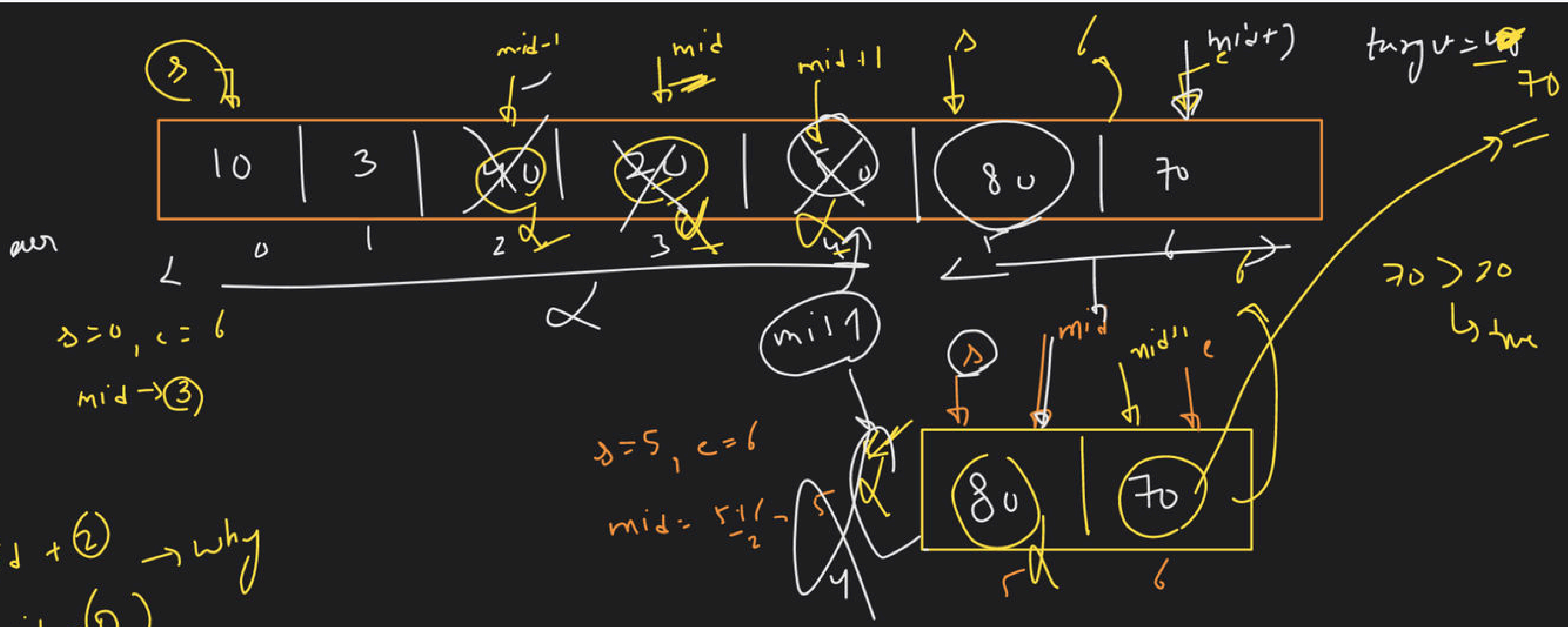
mid-1 >= 0

mid-1 >= s

```
if (arr[mid] == target)
    return mid;

if (arr[mid-1] == target)
    return mid-1;

if (arr[mid+1] == target)
    return mid+1;

if (target > arr[mid])
    └→ R.S → s = mid+2;

else
    └→ L.S → e = mid-2;
```

why

why



s=0, e=6

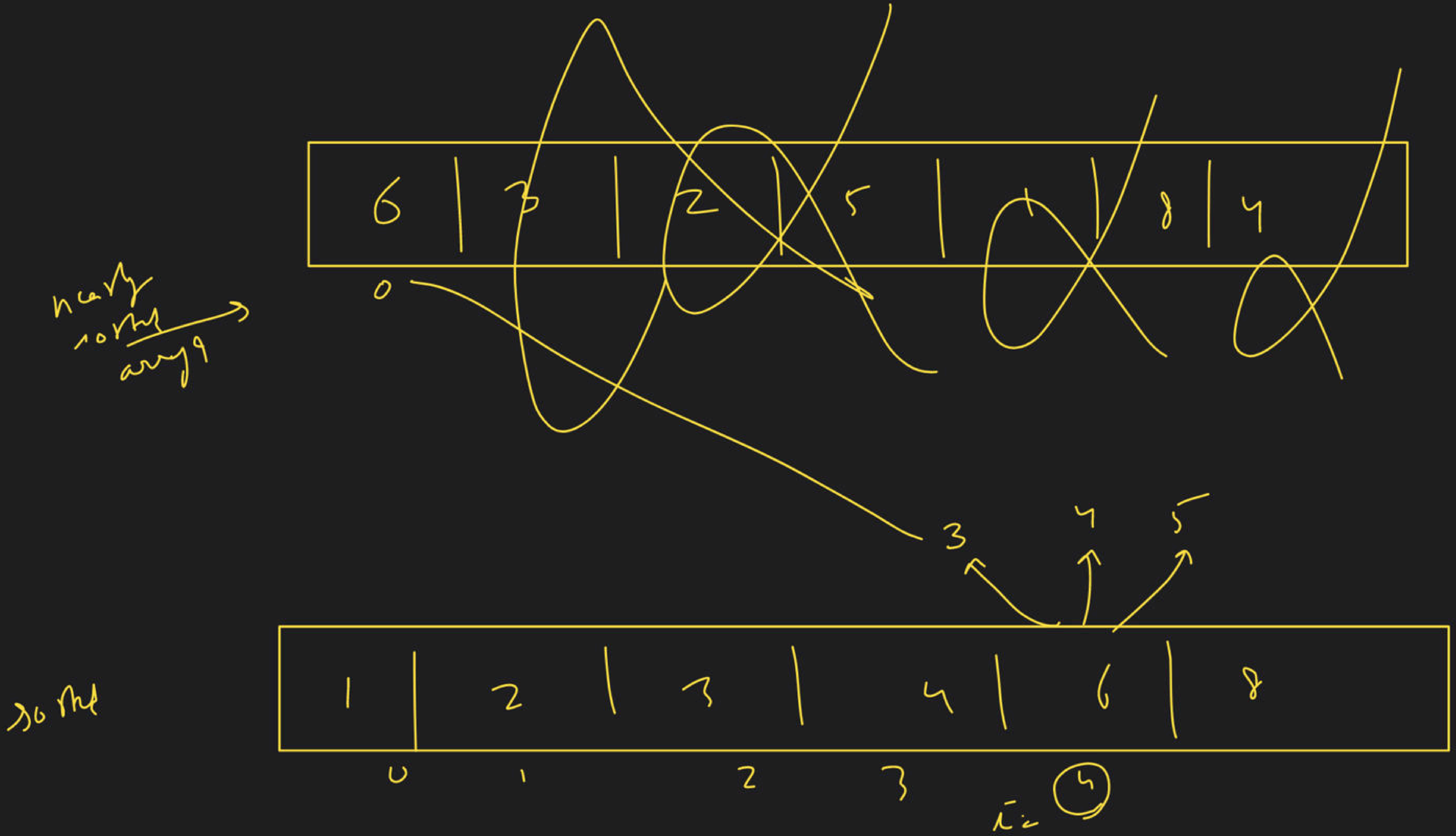
mid → 3

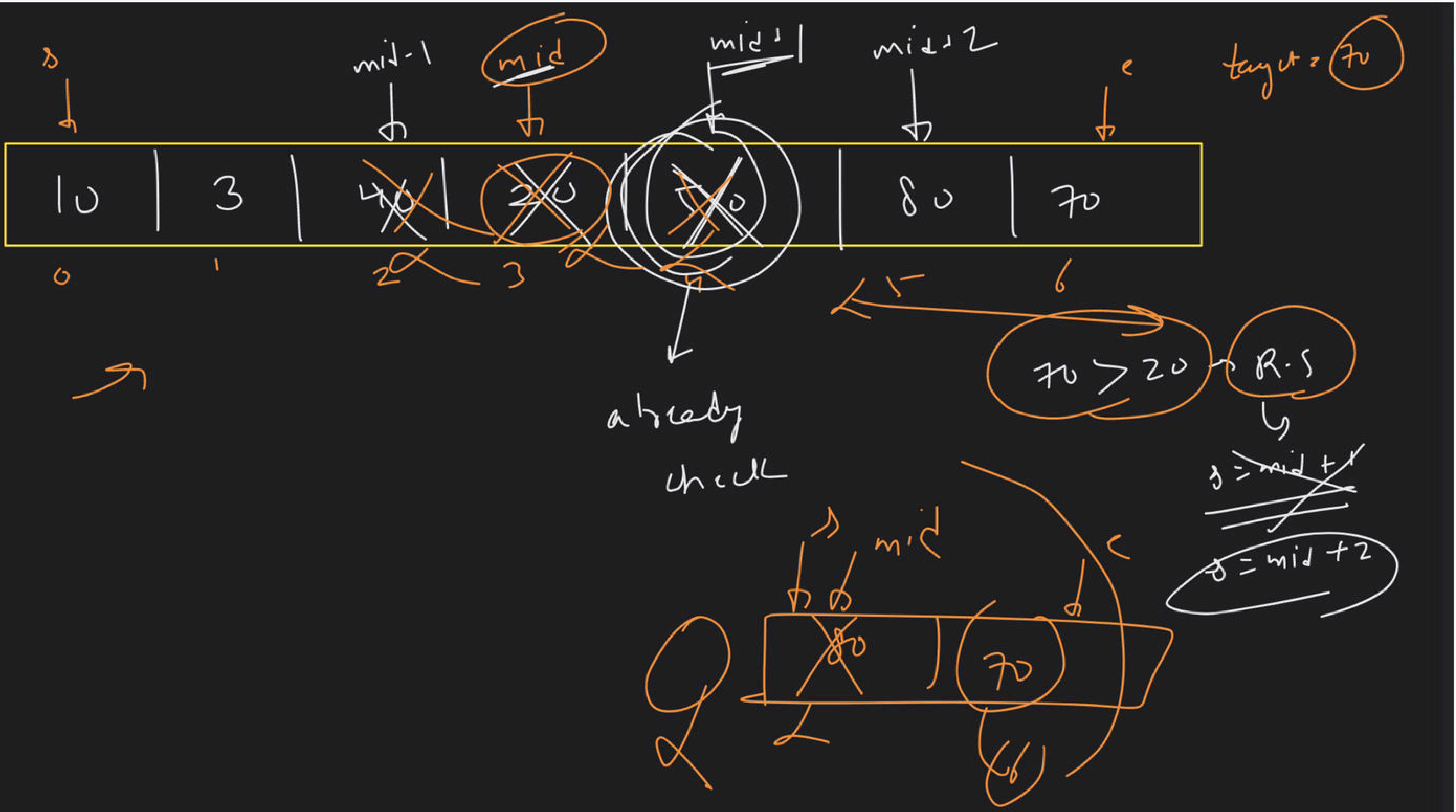
s=5, e=6

mid = $\frac{5+6}{2}$ → 5

s = mid + 1 → why

e = mid - 1


nearly sorted array
6
3
2
5
1
8
4
0
3
4
5
sorted
1
2
3
4
6
8
0
1
2
3
i = 4


s
mid-1
mid
mid+1
mid+2
e
target = 70
10
3
40
20
50
80
70
0
1
2
3
5
6
already check
70 > 20
R.S
~~s = mid+1~~
s = mid+2
s
mid
e
80
70
5
6

→ Divide 2 numbers using Binary Search

dividend = 10

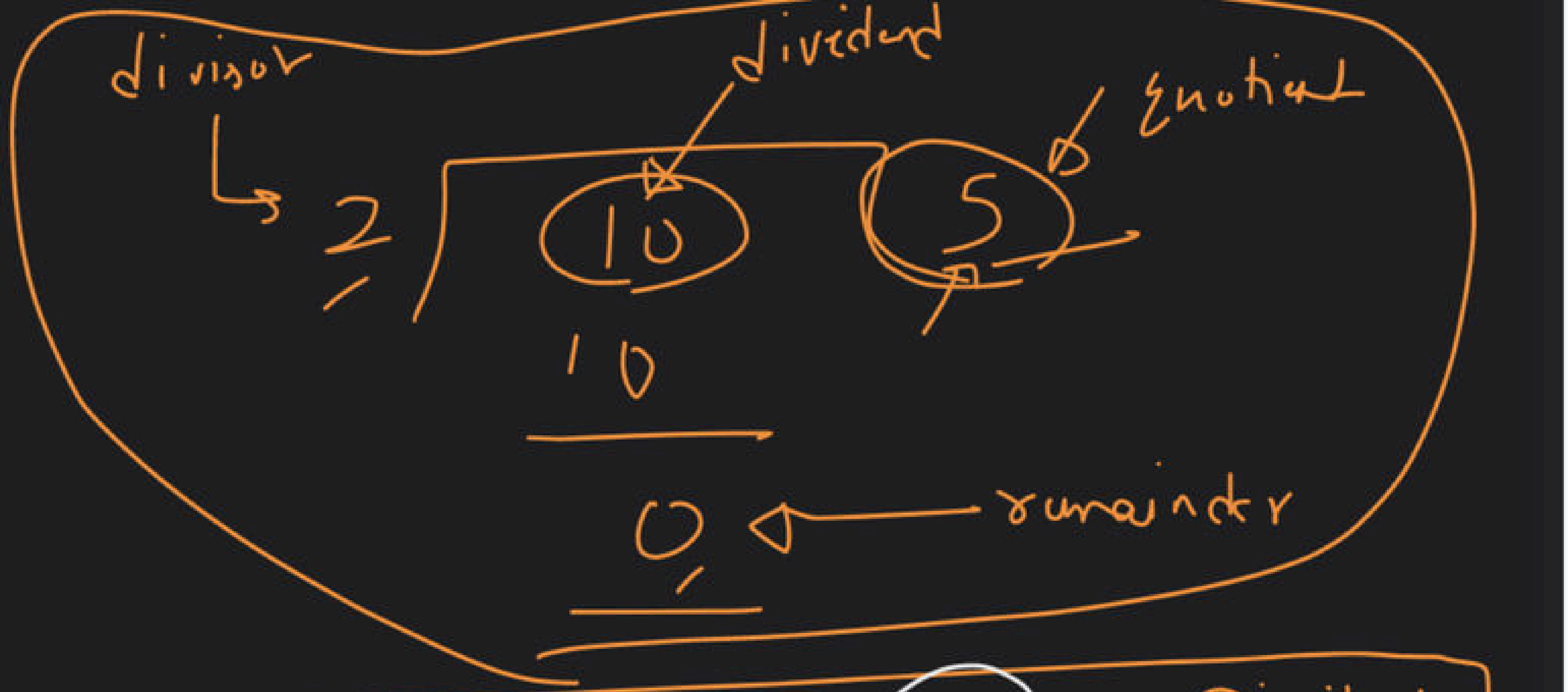
divisor = 2

quotient = ? → 5

0 → divident

s e

search space



quotient * divisor + rem = Divident

B.S

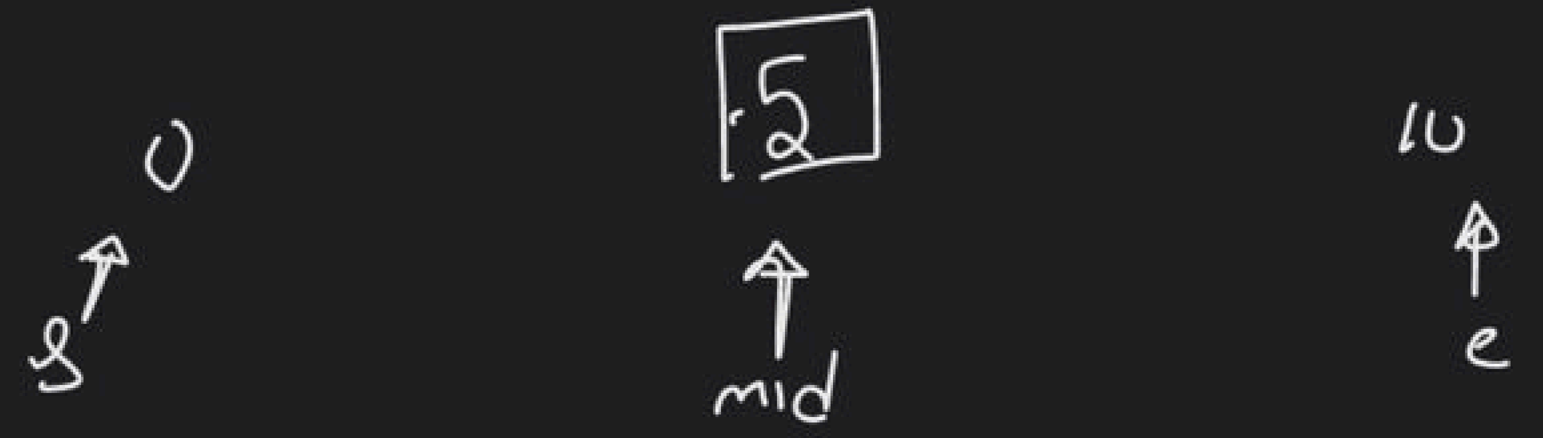
√25 → 0 → 25

mid * mid == target

quotient * divisor + ~~remainder~~ = dividend

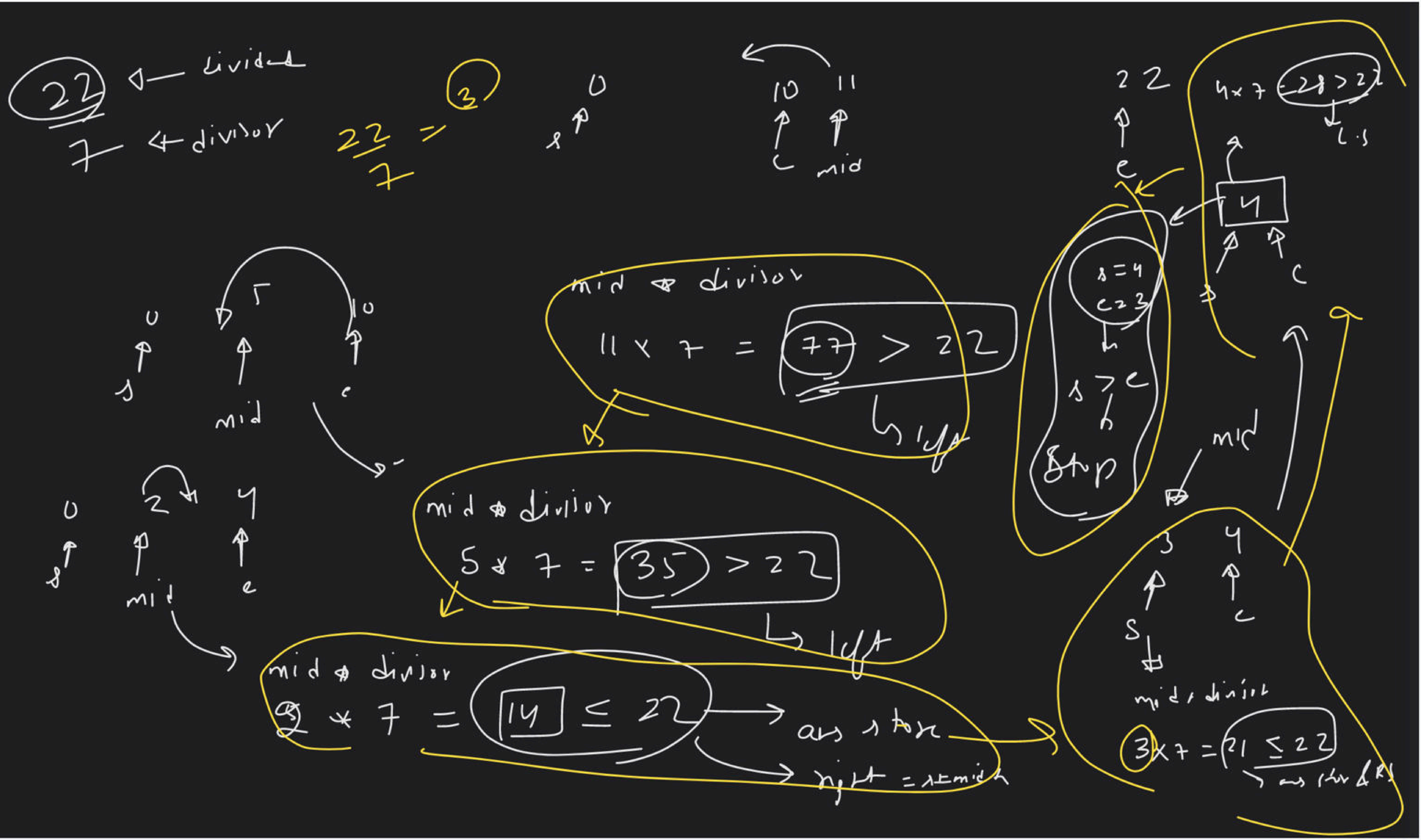
$\boxed{\text{quotient} * \text{divisor} <= \text{divendend}}$

$\frac{10}{\boxed{2}}$

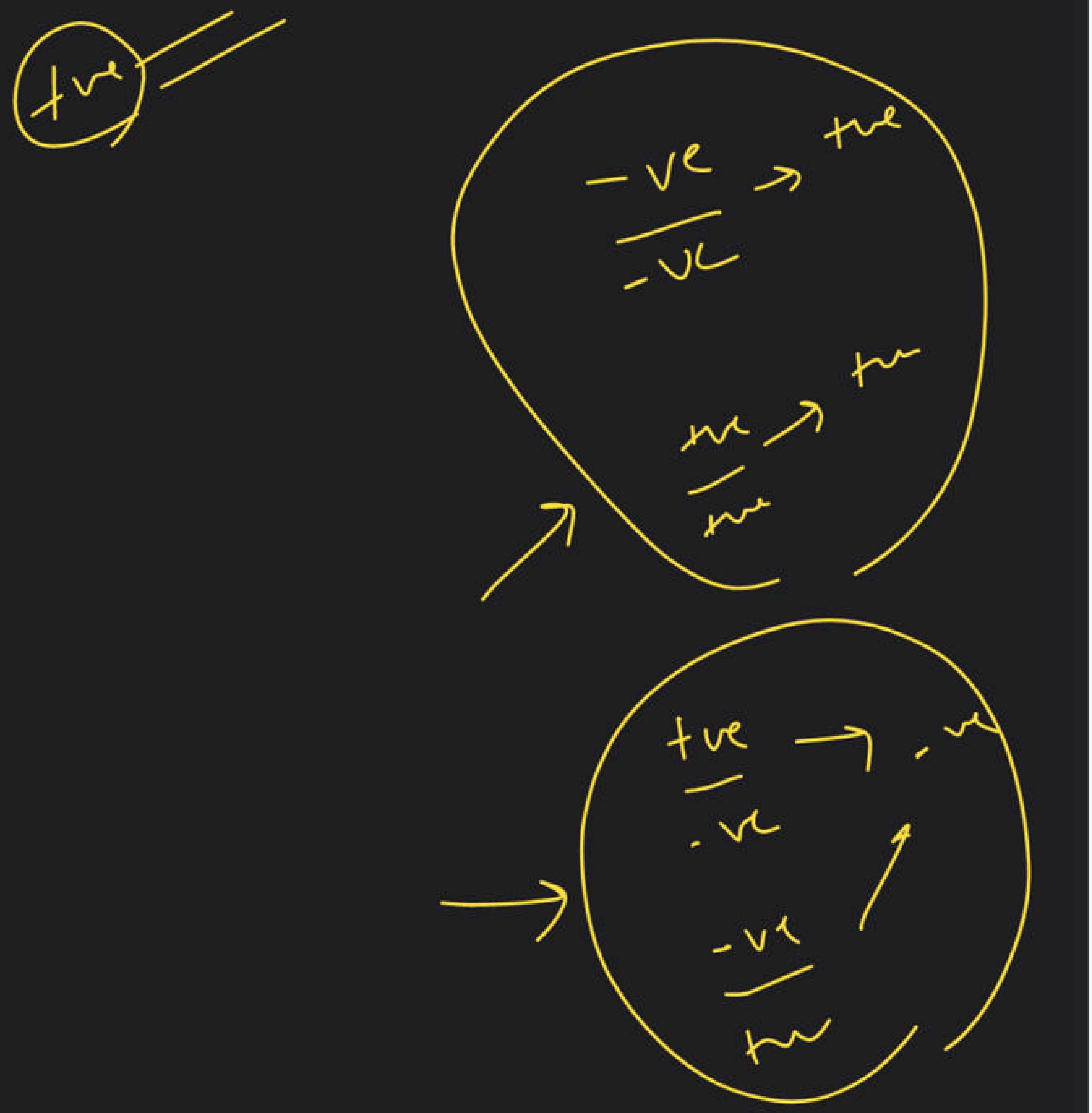


$5 * 2 <= 10$ → true

$5 * 2 == 10$ → return 5



22 ← dividend
7 ← divisor
22/7 = 3
0
s
10 11
e mid
22
e
4×7 = 28 > 22
l.s
4
s e
0 5 10
s mid e
mid * divisor
11 × 7 = 77 > 22
left
s=4
e=3
s > e
stop
mid * divisor
5 * 7 = 35 > 22
left
0 2 4
s mid e
mid
mid * divisor
2 * 7 = 14 ≤ 22
ans store
right = s=mid+1
3 4
s e
mid * divisor
3×7 = 21 ≤ 22
ans store & RS


+ve
-ve / -ve → +ve
+ve / +ve → +ve
+ve / -ve → -ve
-ve / +ve

65 ← dividend

4 ← divisor

16

$\cancel{65}$

$\cancel{4}$

~~15~~ 16

ans

0 ↑ s   32 ↑ mid   65 ↑ e

mid * divisor = 32×4 = 128 > 65

↳ L.S → e = mid - 1

e = 32 - 1 = 31

0 ↑ s   15 ↑ mid   31 ↑ e

mid * divisor = 15×4 = 60 < 65

↳ ans store

↳ R.S → s = mid + 1 = 15+1 = 16

16 ↑ s   23 ↑ mid   31 ↑ e → mid * divisor = 23×4

= 92 > 65 → L.S → e = mid - 1 = 23 - 1 = 22

16 $\quad$ $\boxed{19}$ $\quad$ 22

s $\quad$ mid $\quad$ e

$mid * divisor = 19 \times 4 = 76 > 65$

↳ Left $= e = mid - 1$

$= e = 19 - 1$

$= 18$

16 $\quad$ 17 $\quad$ 18

s $\quad$ mid $\quad$ e

$mid * divisor = 17 \times 4 = 68 > 65$

↳ left → $e = mid - 1 = 16$

mid

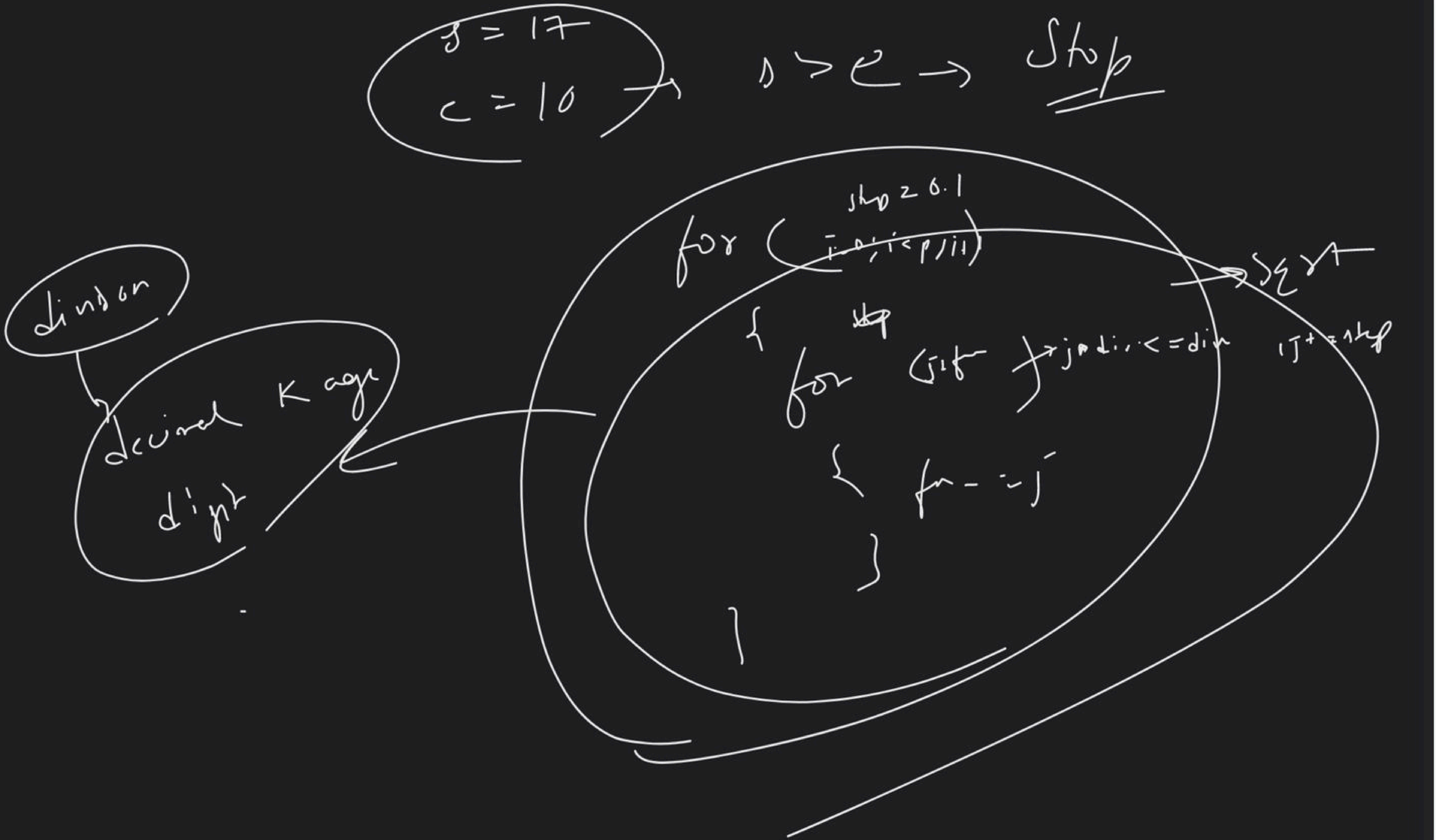
$\boxed{16}$

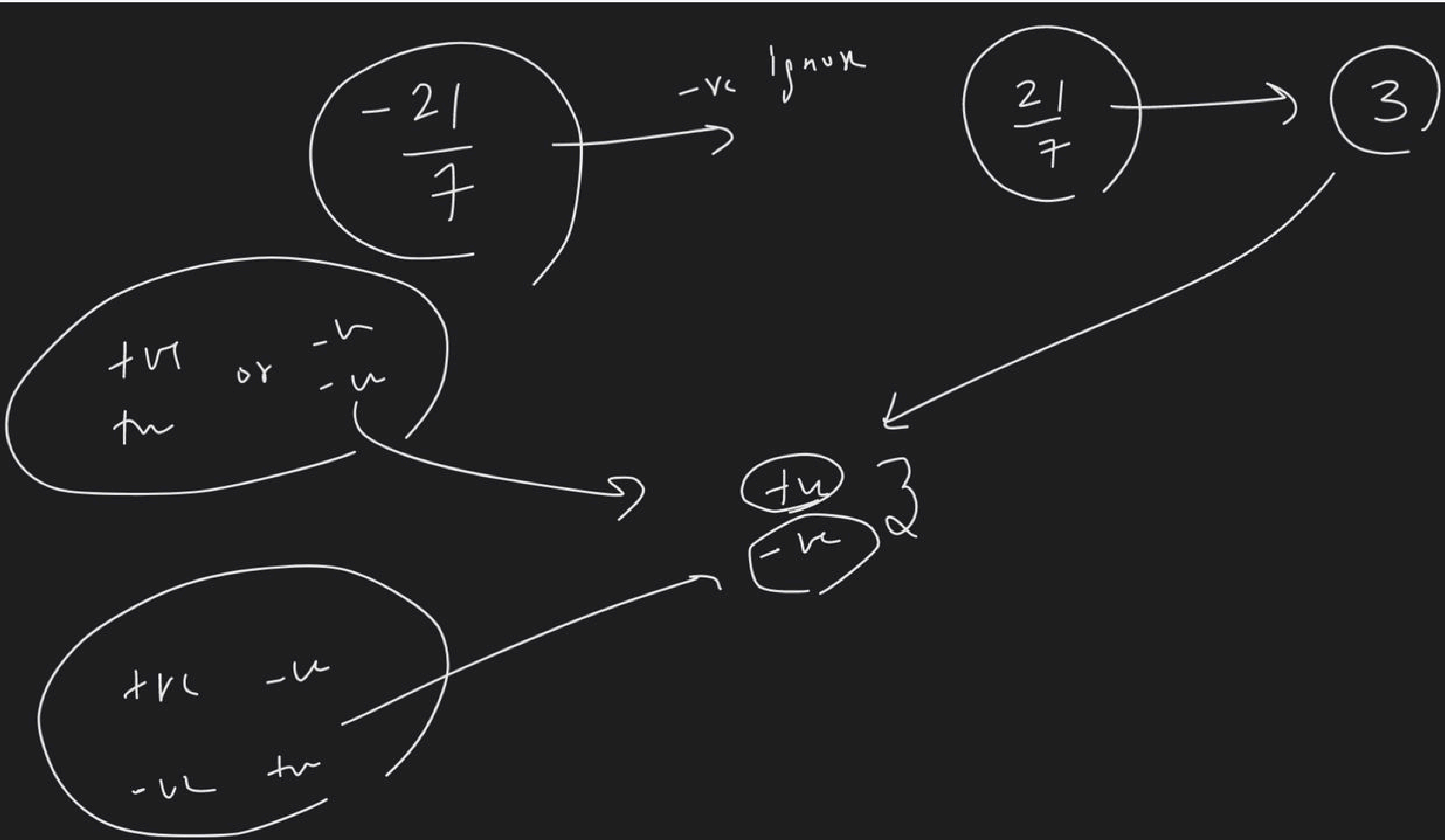
s $\quad$ e

→ $mid * divisor = 16 \times 4 = 64 < 65$ → answer

↳ R-1 → $s = mid + 1$

$= 17$

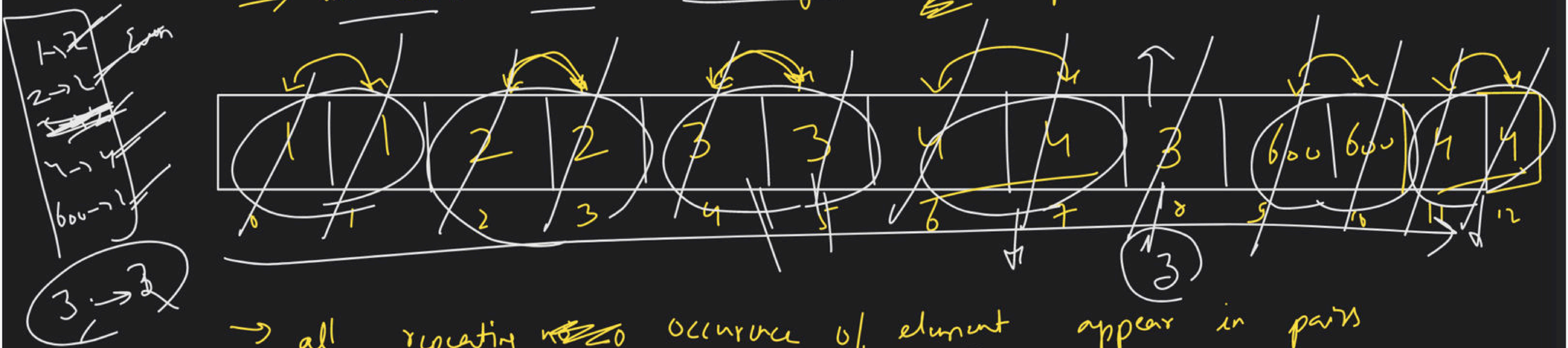
s = 17
e = 10
s > e → Stop
step = 0.1
for (i=0; i<p; i++)
{
step
for (j=0 → j=di ; j<=dim j+=step
{ for – i j
}
}
SET
division
decimal K age
digit



$-\frac{21}{7}$
-ve Ignore
$\frac{21}{7}$
3
+ve or -ve
+ve -ve
+ve
-ve
3
+ve -ve
-ve +ve

→ find the Odd occuring element in an array.

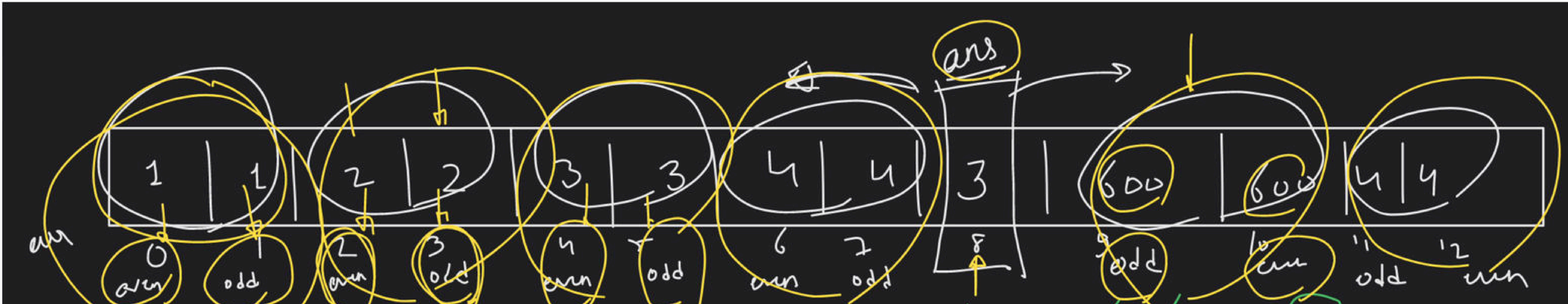
→ all element occur even no of times ~~&~~ except one



→ all repeating ~~no~~ occurrence of element appear in pairs

& pairs are not adjacent (there cannot be more than 2 (consecutive occurrence of any element)

→ find the element that appears odd no of times

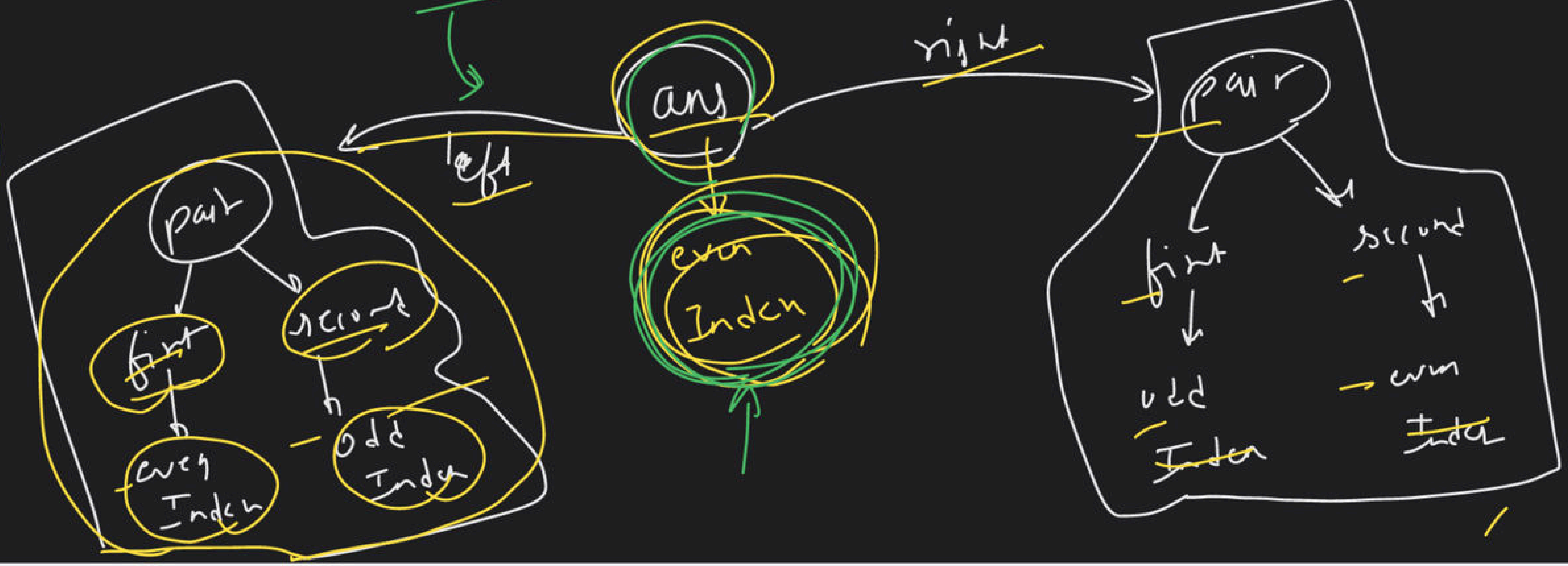


Xor approach → 1^1^2^2^3^3^4^4^3^600^600^4^4 = 3

T.C → O(n)

B.S

Verify

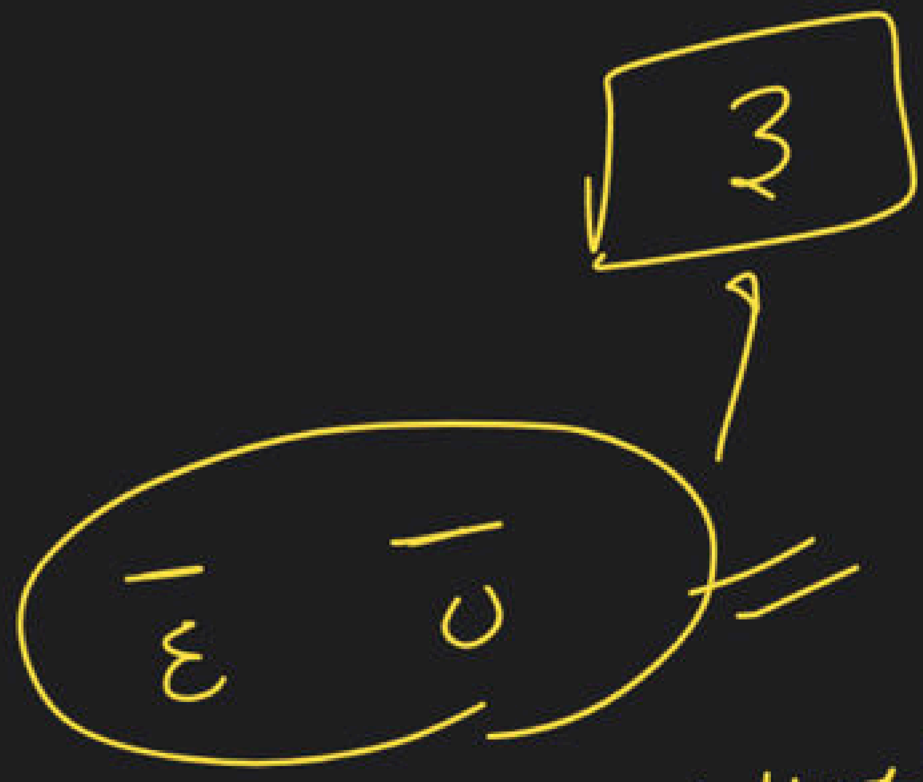


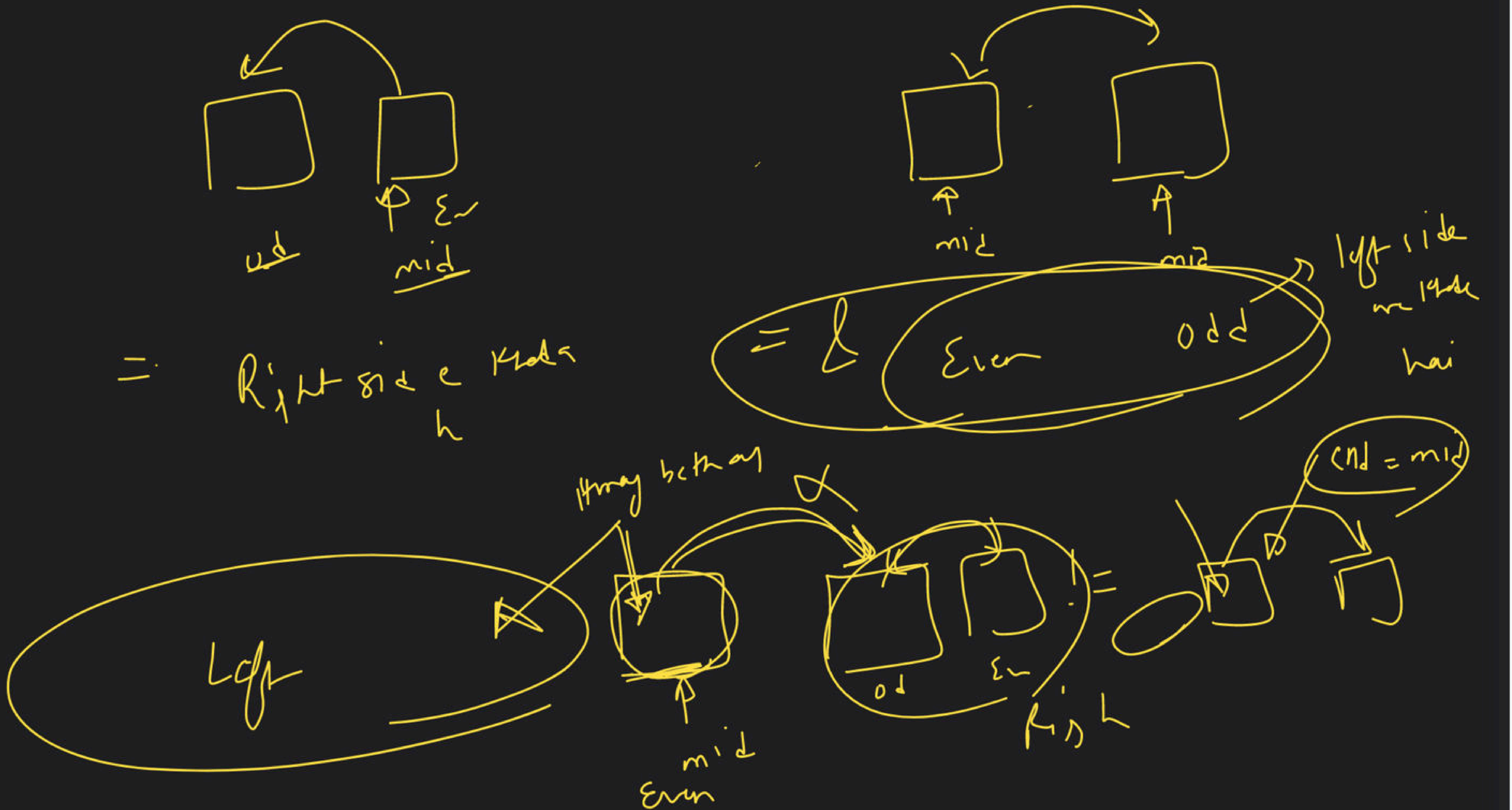
$x \to x$

even odd

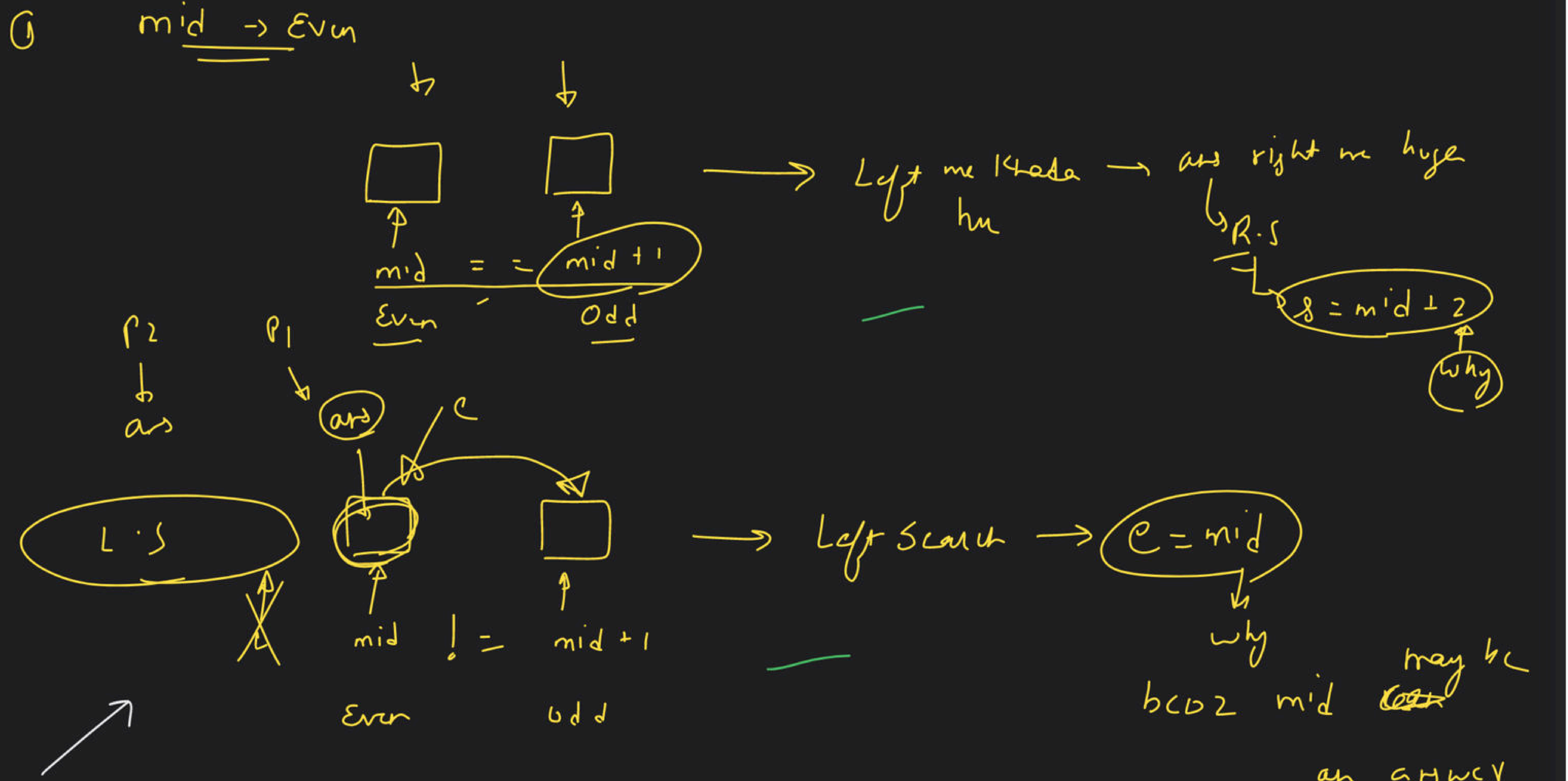
```
s = 0 , e = n-1

mid = (s+e)/2

while (s <= e) {

    if (s == e)
        return s;

    if (mid % 2 == 0)
    //Even
        if (arr(mid) == arr(mid+1))
            └→ R.S → s = mid+2
        else
            e = mid

    else //odd
    {

    }
}
```

left me khade hu




od
Ev
mid
= Right side e khda h
mid
mid
= l Even odd
left side me khade hai
Array bekhar
Left
mid
Even
od
Ev
Right
end = mid

not equal



① mid -> Even

mid == mid + 1

Even Odd

→ Left me khada hu → ans right me hoga

↳ R.S

↳ s = mid + 2

why

P2 ↓ ans

P1 ↓ ans

e

L.S

mid != mid + 1

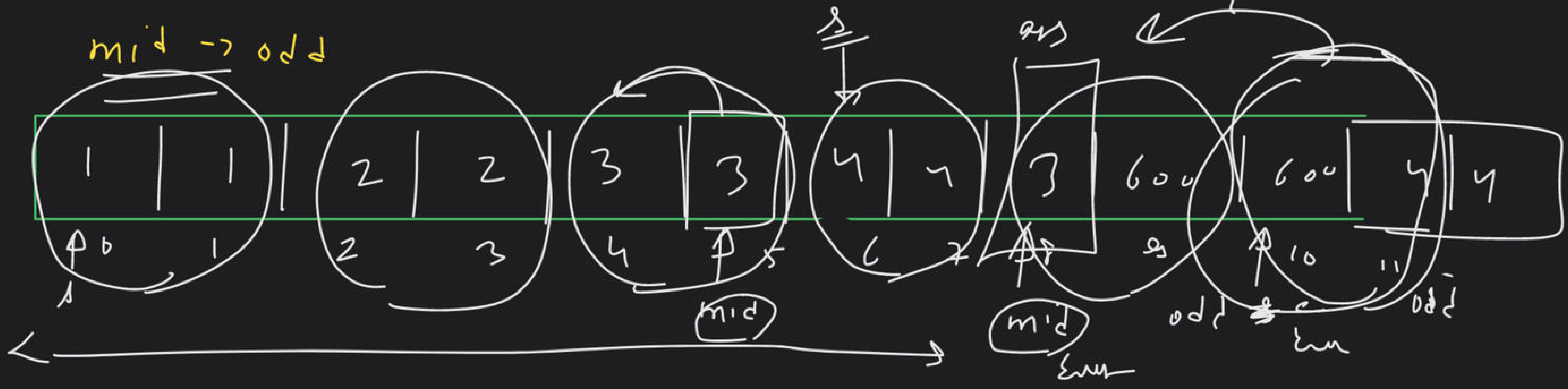
Even Odd

→ Left Search → e = mid

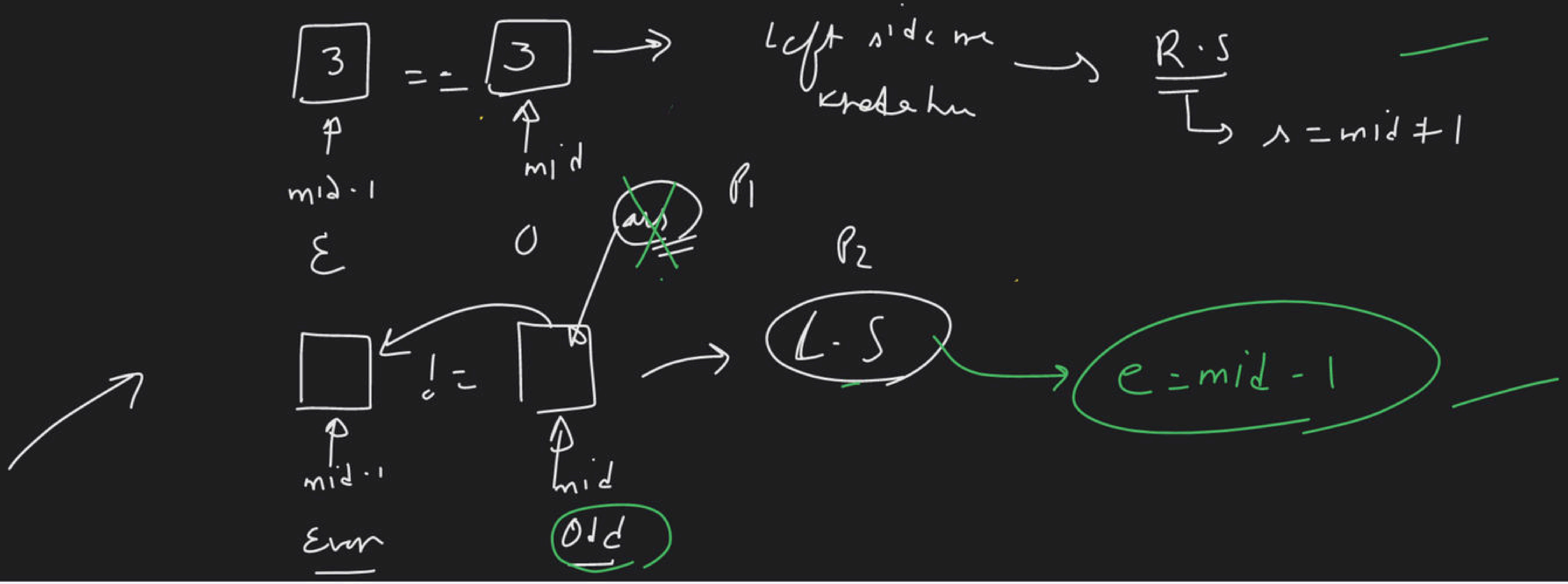
why

bcoz mid ~~[illegible]~~ may be an answer

(2) mid -> odd



1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 3 | 600 | 600 | 4 | 4
0 1 2 3 4 5 6 7 8 9 10 11
s
ans
mid
mid
Even
odd
Even
odd

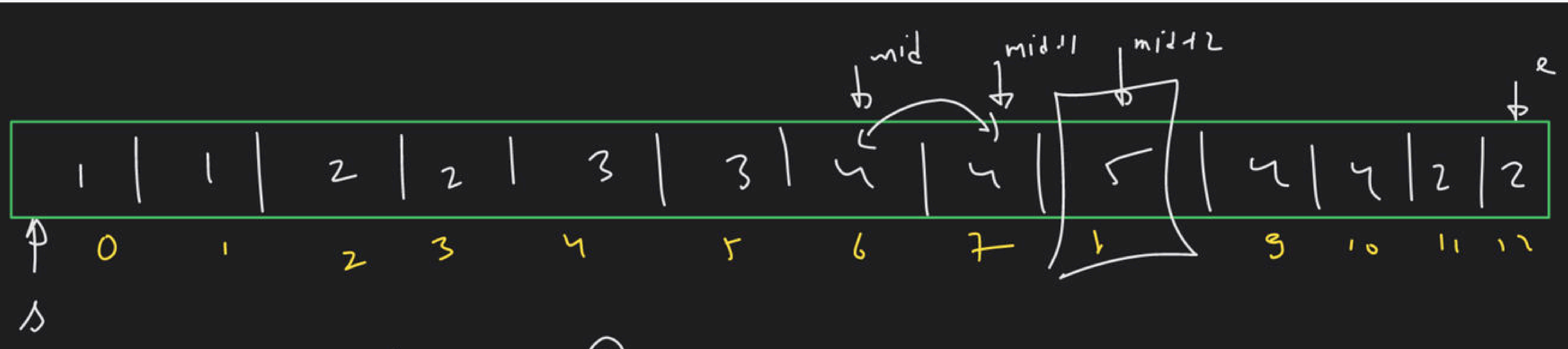


3 == 3 → Left side me khade hu → R.S
↳ s = mid + 1
mid-1
mid
E
O
~~ans~~
P1
P2
!= → L.S → e = mid - 1
mid-1
mid
Even
Odd

```
if (mid %2 ==0)
  {
        if (arr[mid] == arr[mid +1])
                s = mid + 2;
        else
              e = mid;

else → //mid -> odd
  {
      if (arr(mid) == arr[mid-1])
              s = mid +1;
      else
            e = mid-1;
  }
```

HomeWork

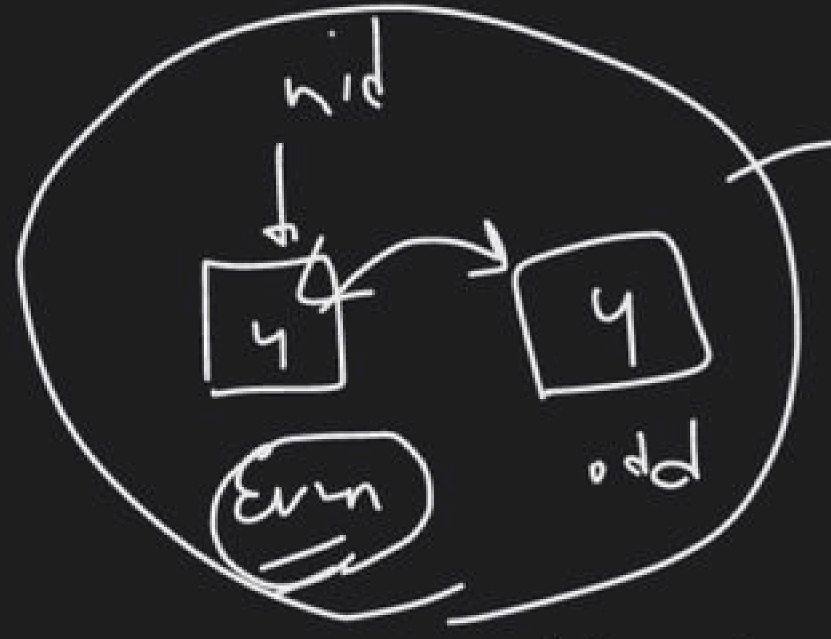
- find pairs with difference 'K' in an array
- find 'K' closest element to a given value in an array
- Exponential Search
- UnBounded Binary Search
- Adv B.S
  - Book Allocation
  - Painter Partition
  - Aggressive Cow
  - ROTI/PRATA SPOJ
  - EKO SPOJ



s = 0

e = 12

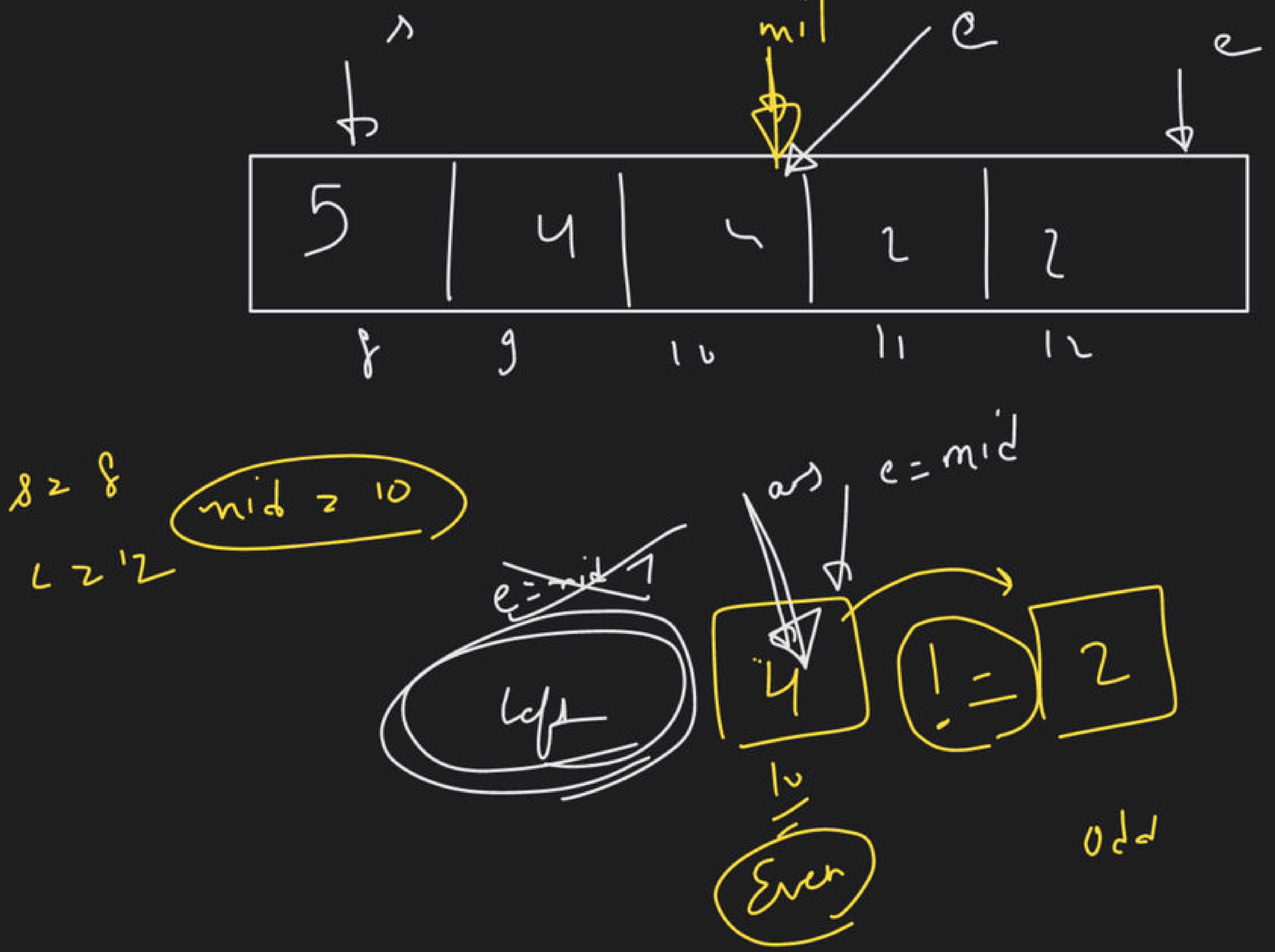
mid = $\frac{0+12}{2}$ = 6

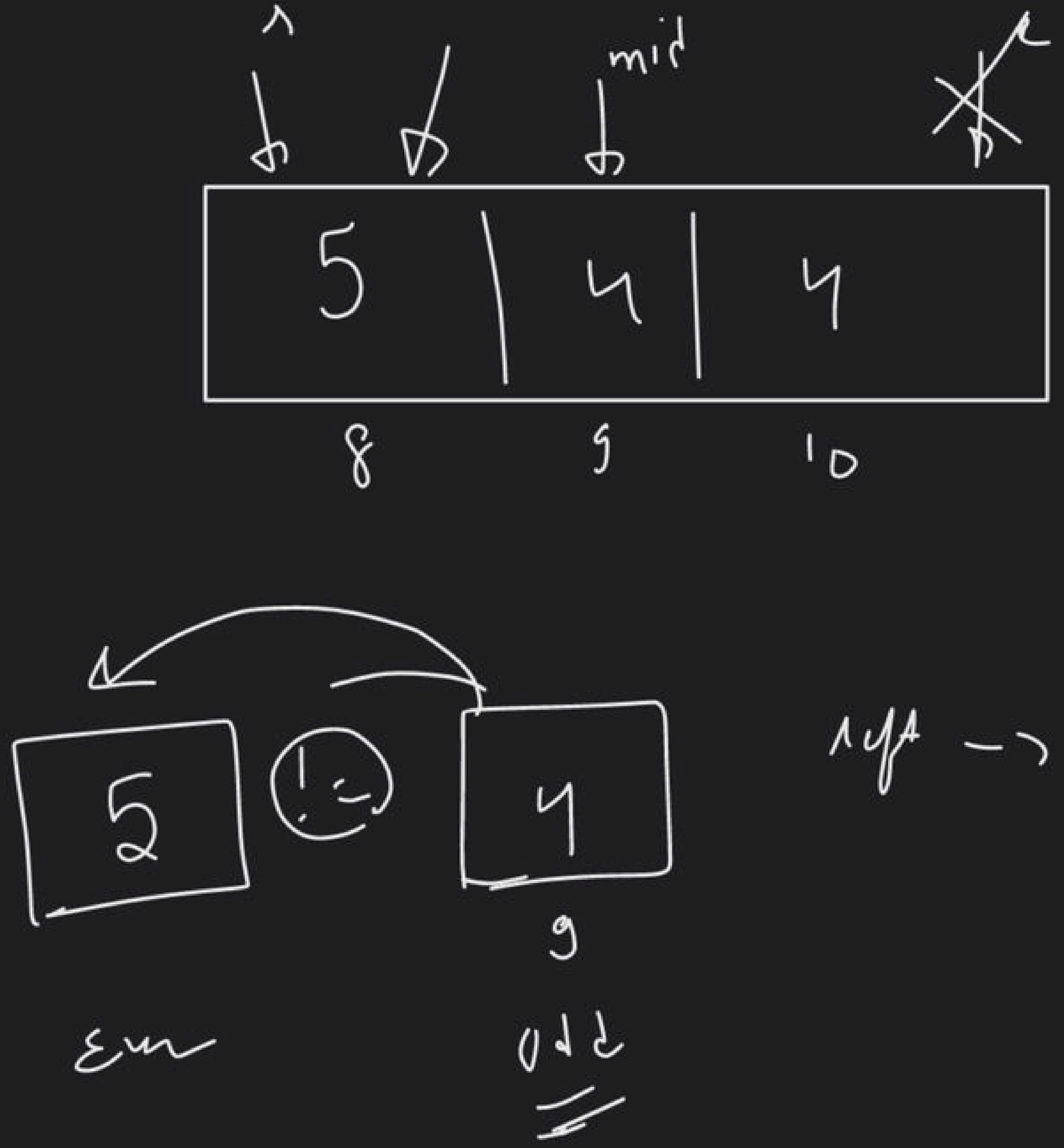


R.S

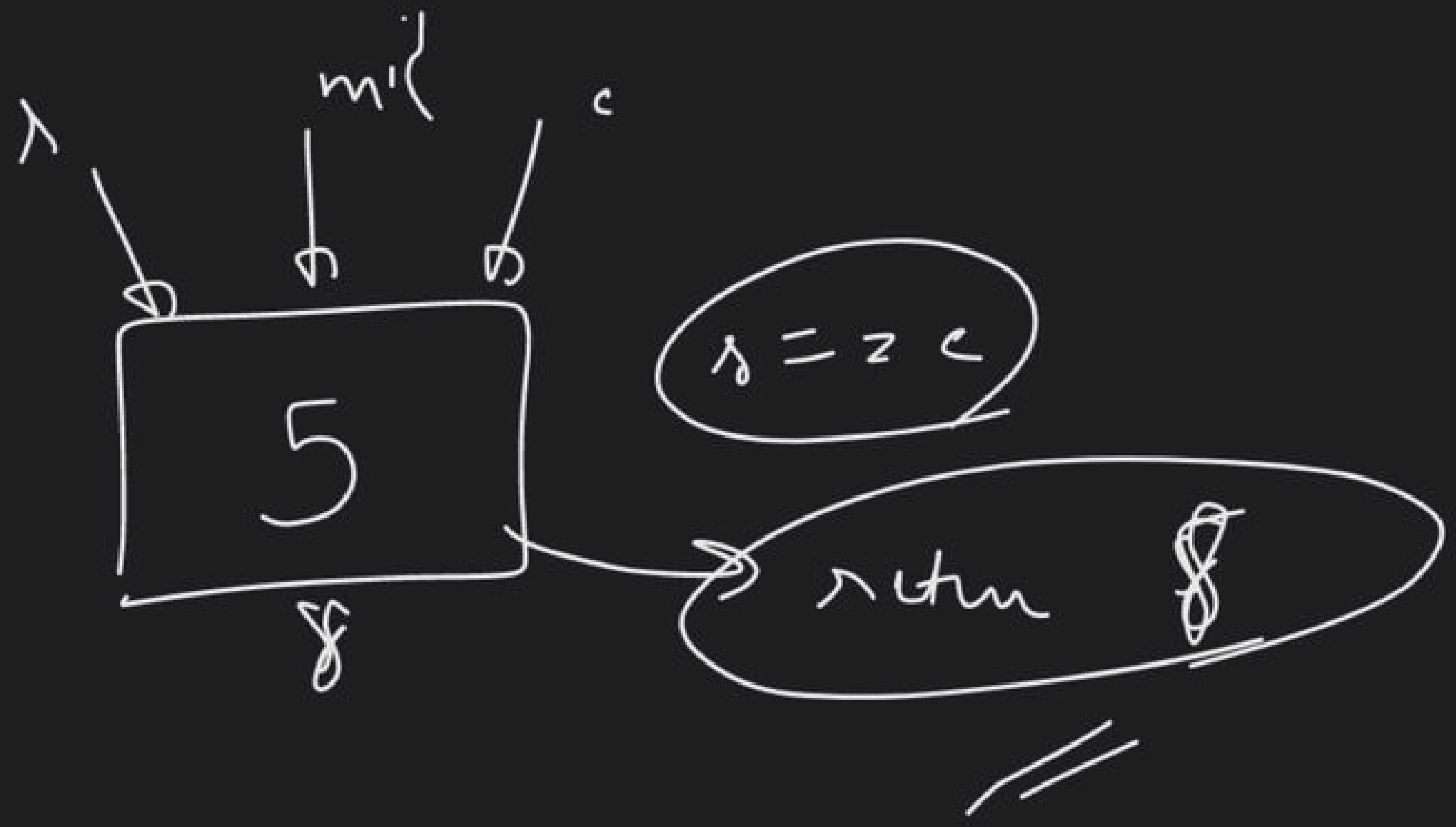
s = mid+2

Left side me
match ho


s
mil
e
e
5
4
4
2
2
8
9
10
11
12
s = 8
e = 12
mid = 10
ans
e = mid
~~e = mid - 1~~
left
4
10
Even
!=
2
Odd



left → r = mid - 1


s
mid
e
5
f
s == e
return f