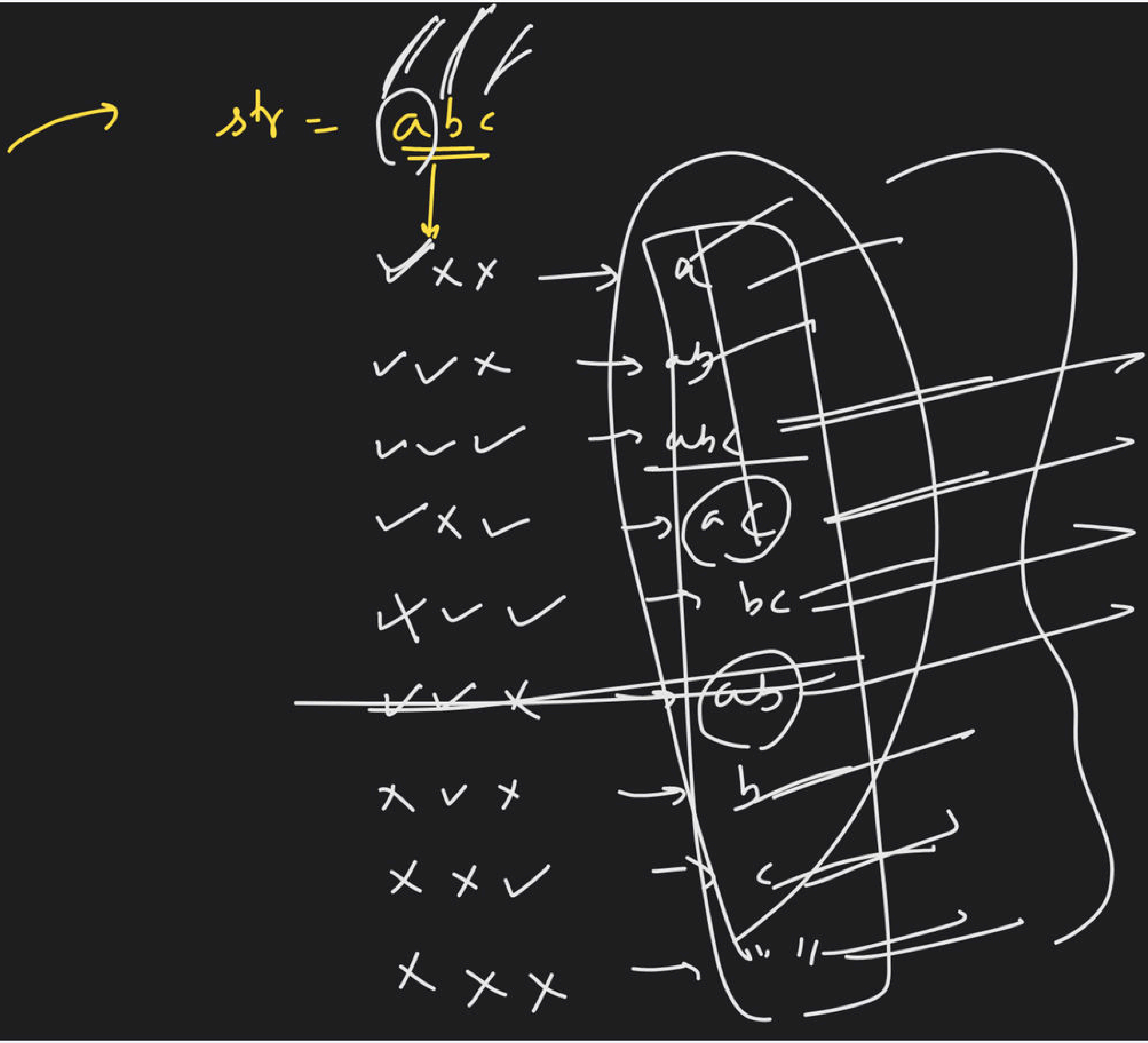
Dynamic Programming Class - 6

Special class

Love Babbar • June 14, 2023

→ DP on Strings

① LCS → Longest Common Subsequence



str = abc

✓ ✗ ✗ → a

✓ ✓ ✗ → ab

✓ ✓ ✓ → abc

✓ ✗ ✓ → ac

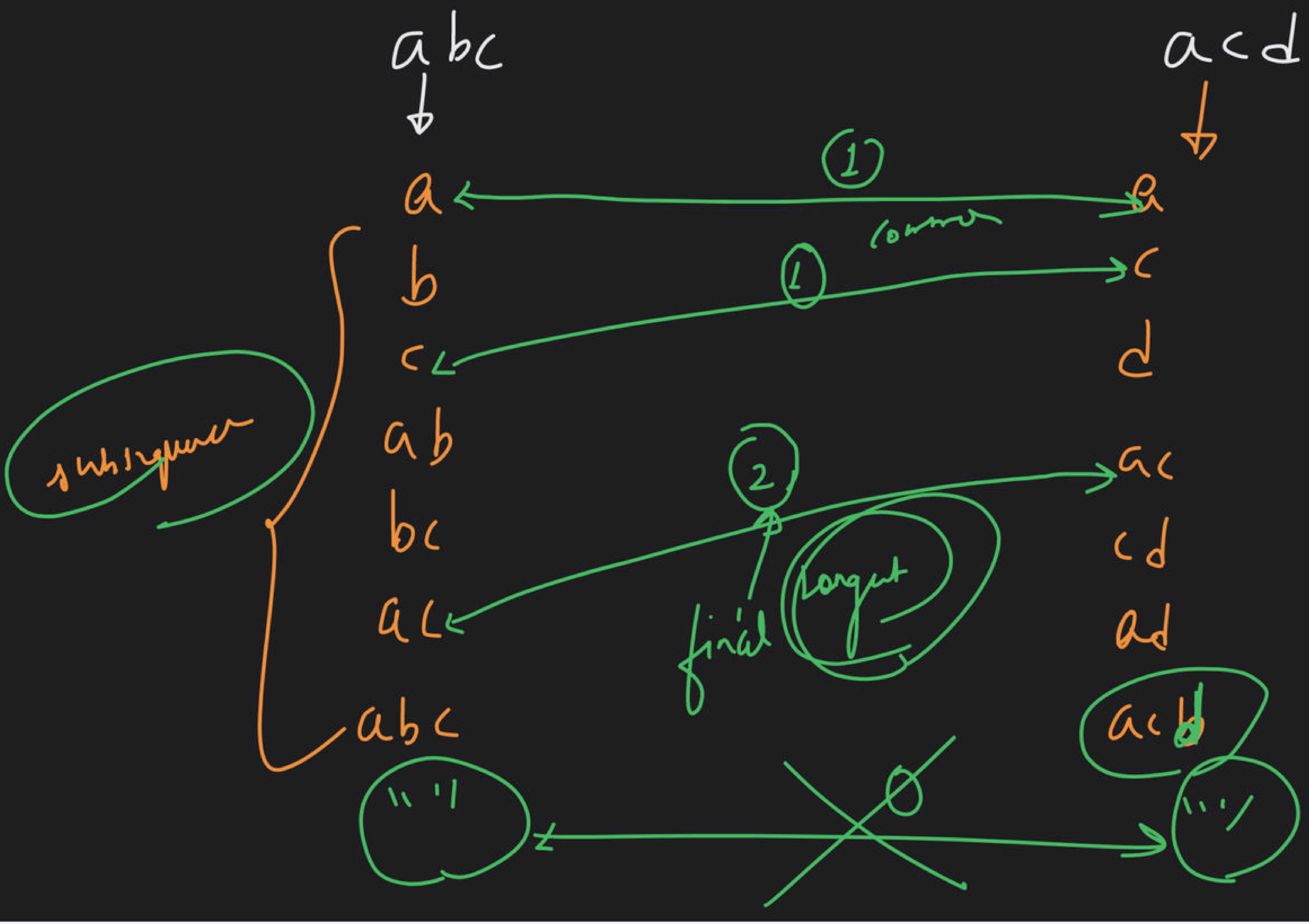
✗ ✓ ✓ → bc

~~✓ ✓ ✗~~ → ab

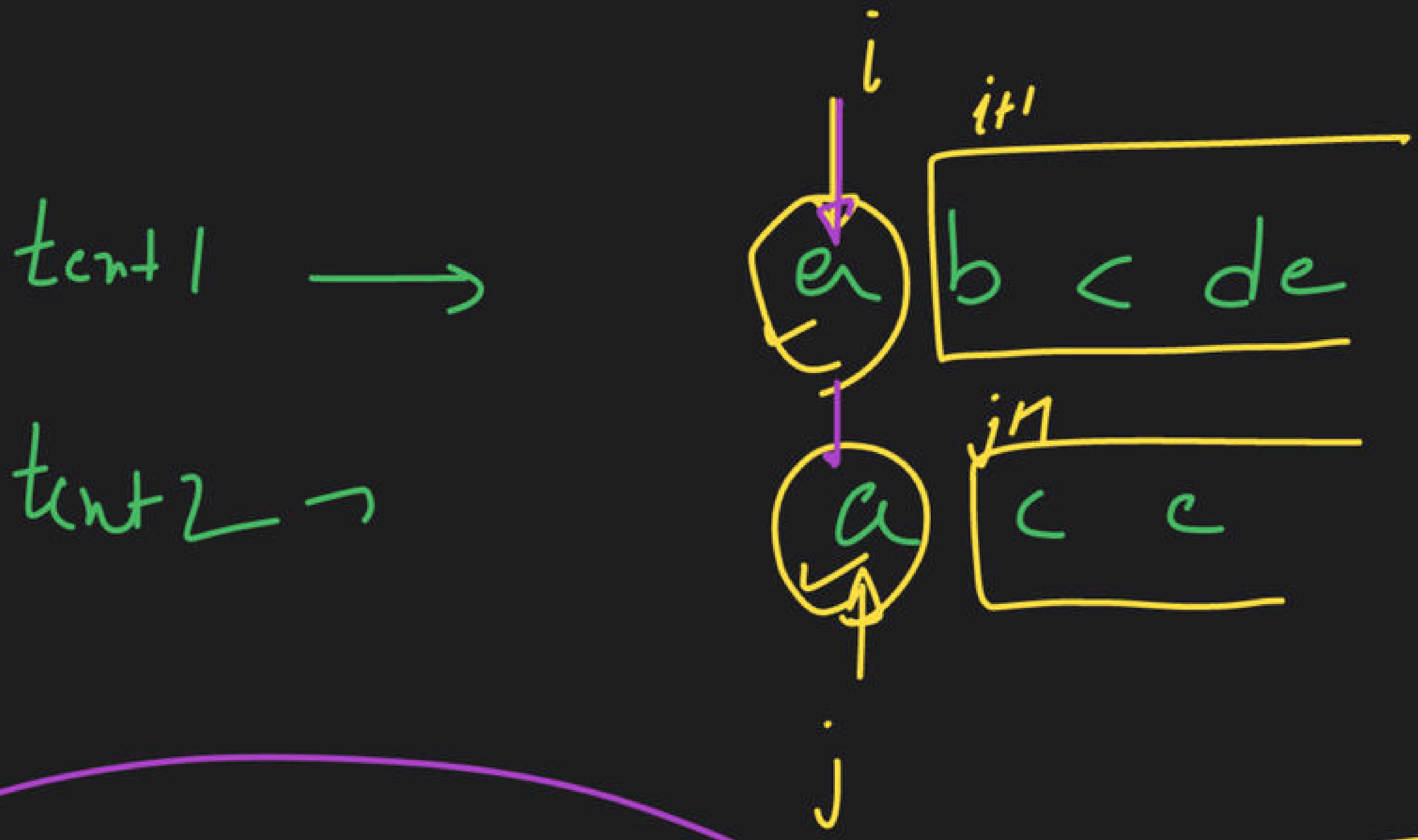
✗ ✓ ✗ → b

✗ ✗ ✓ → c

✗ ✗ ✗ → ""


abc
acd
a
b
c
ab
bc
ac
abc
''
subsequence
1
common
1
2
final
Longest
0
a
c
d
ac
cd
ad
acd
''

text1
subsequence
text 2
Subs
length → ?
max length
TLC



char match

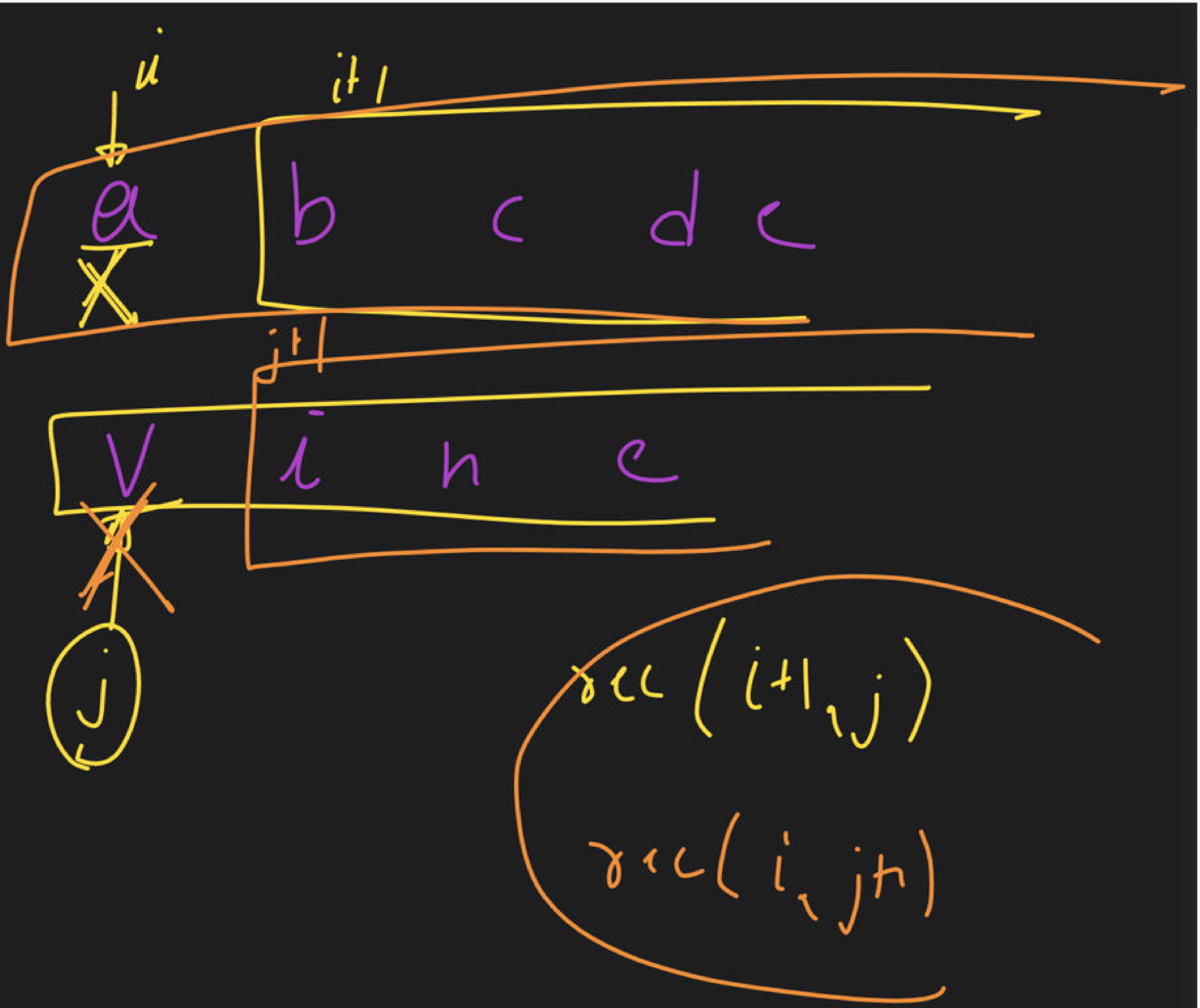
↓

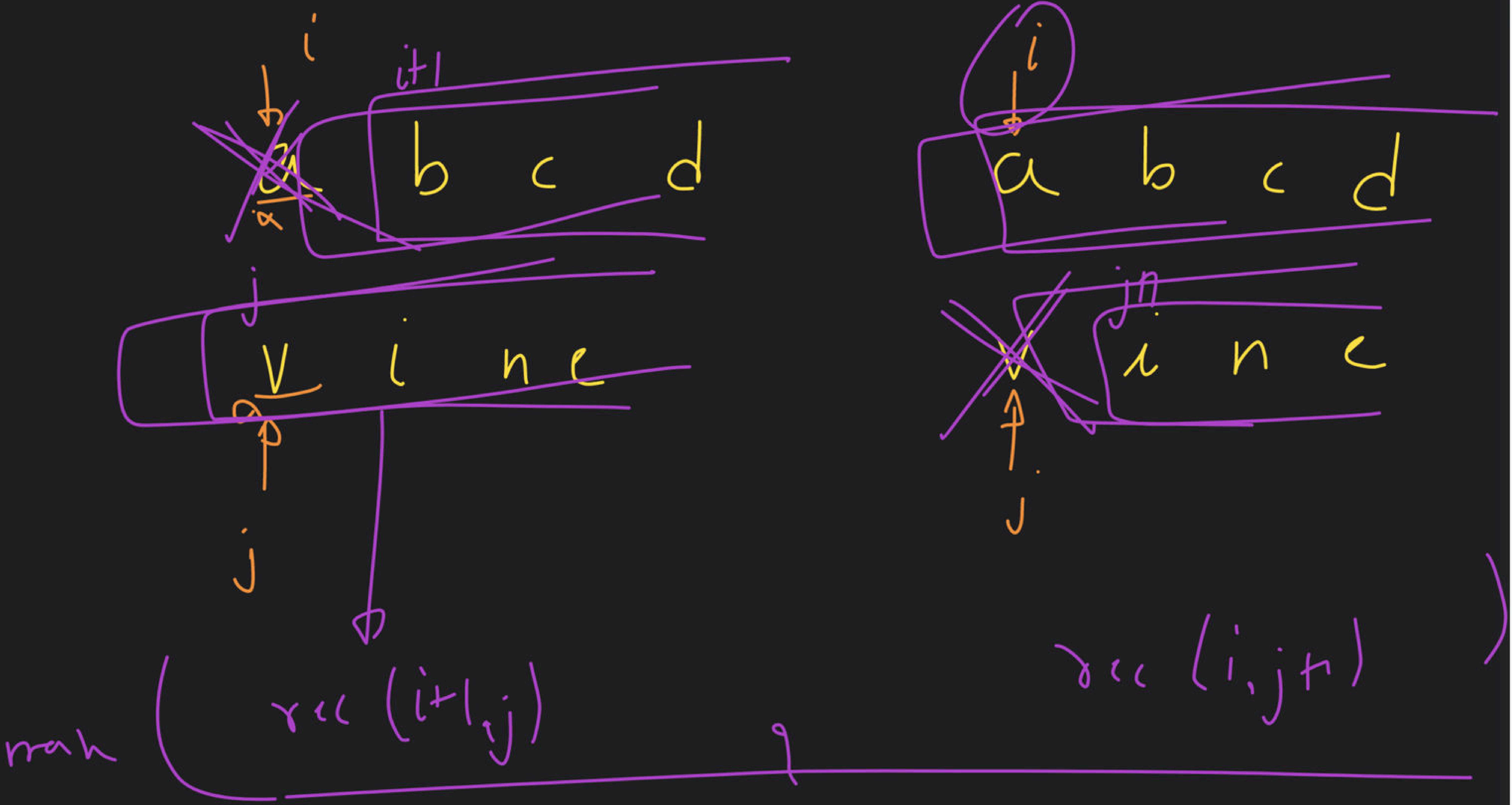
include

char match → aage ka recursion sambhalega

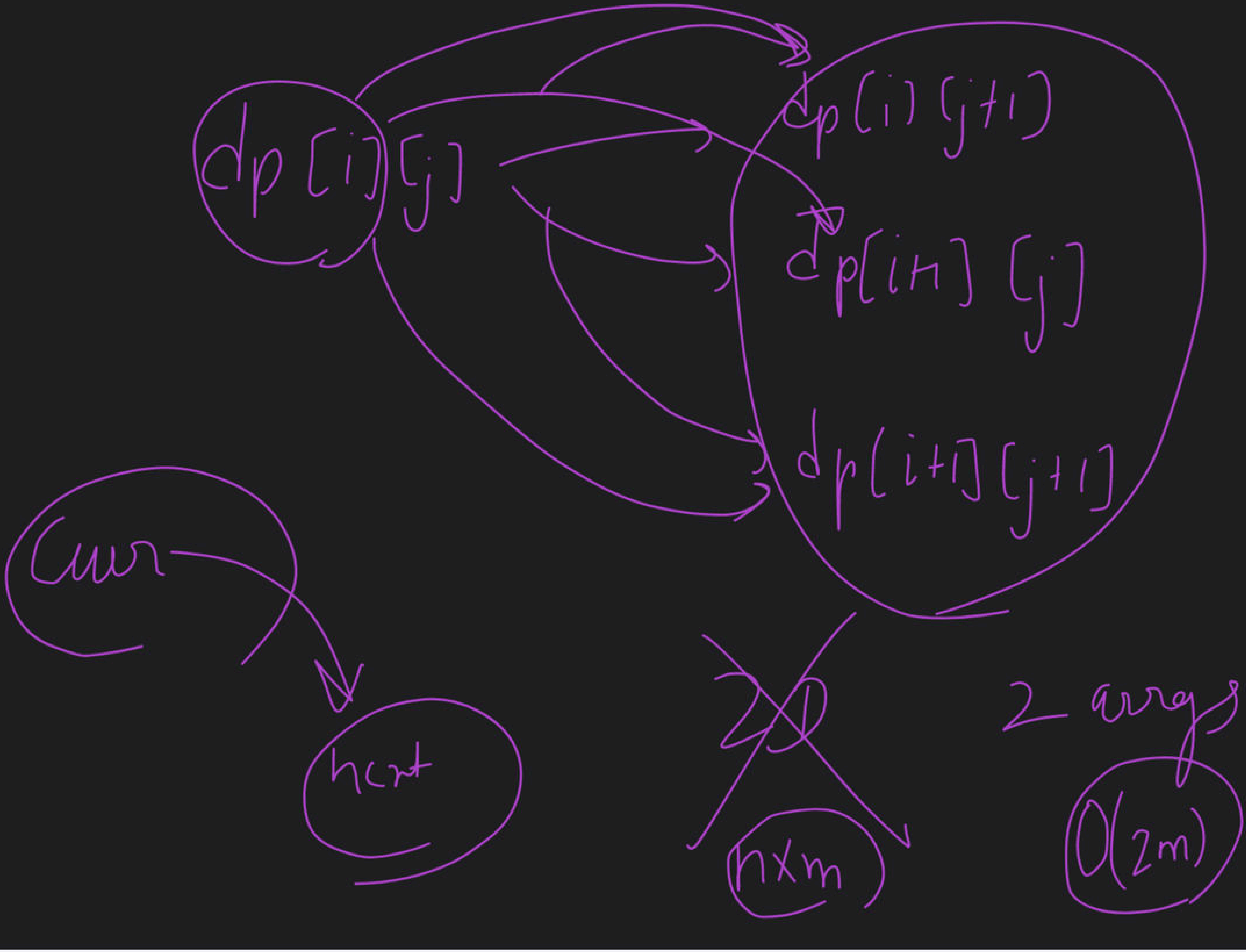
text1[i] == text2[j] → rec (i+1, j+1)

char not matching


i
i+1
a
b c d e
j+1
v
i n e
j
rec(i+1, j)
rec(i, j+1)

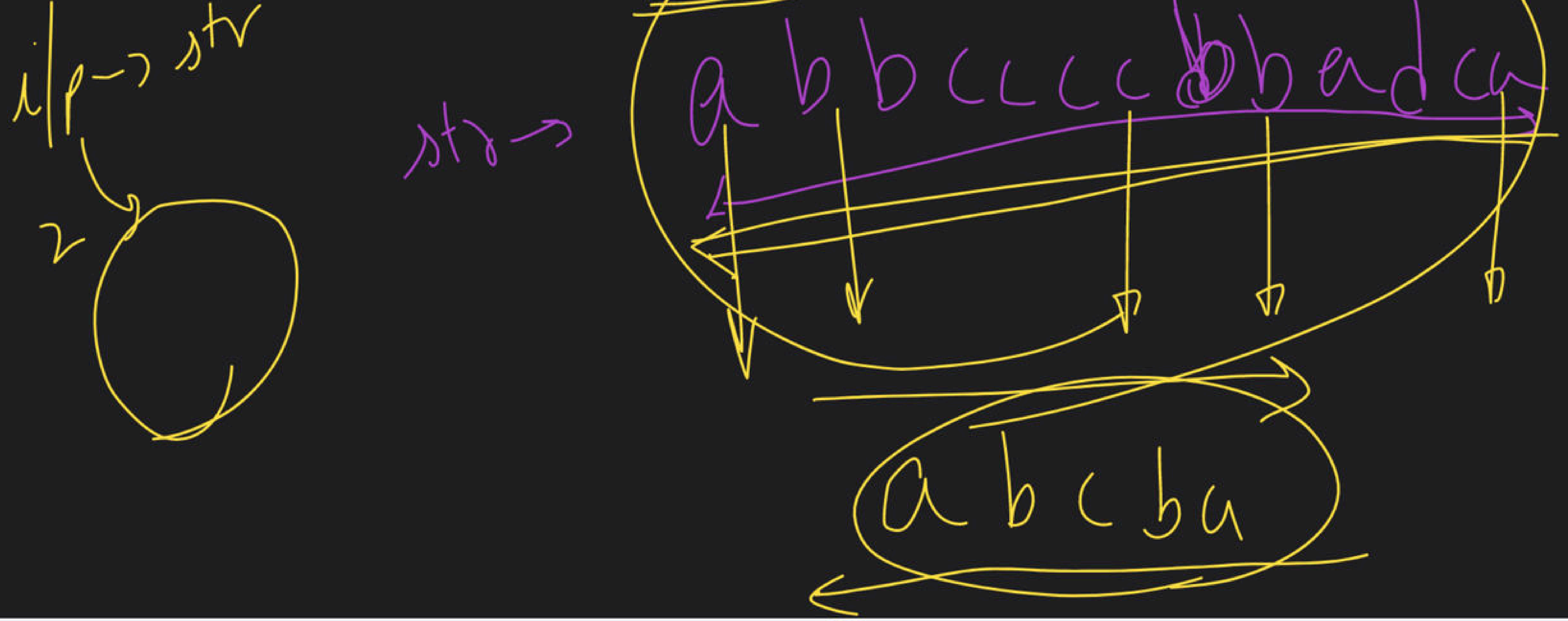

i
i+1
a b c d
j
v i n e
j
max (
rec (i+1, j)
,
i
a b c d
j+1
v i n e
j
rec (i, j+1)
)

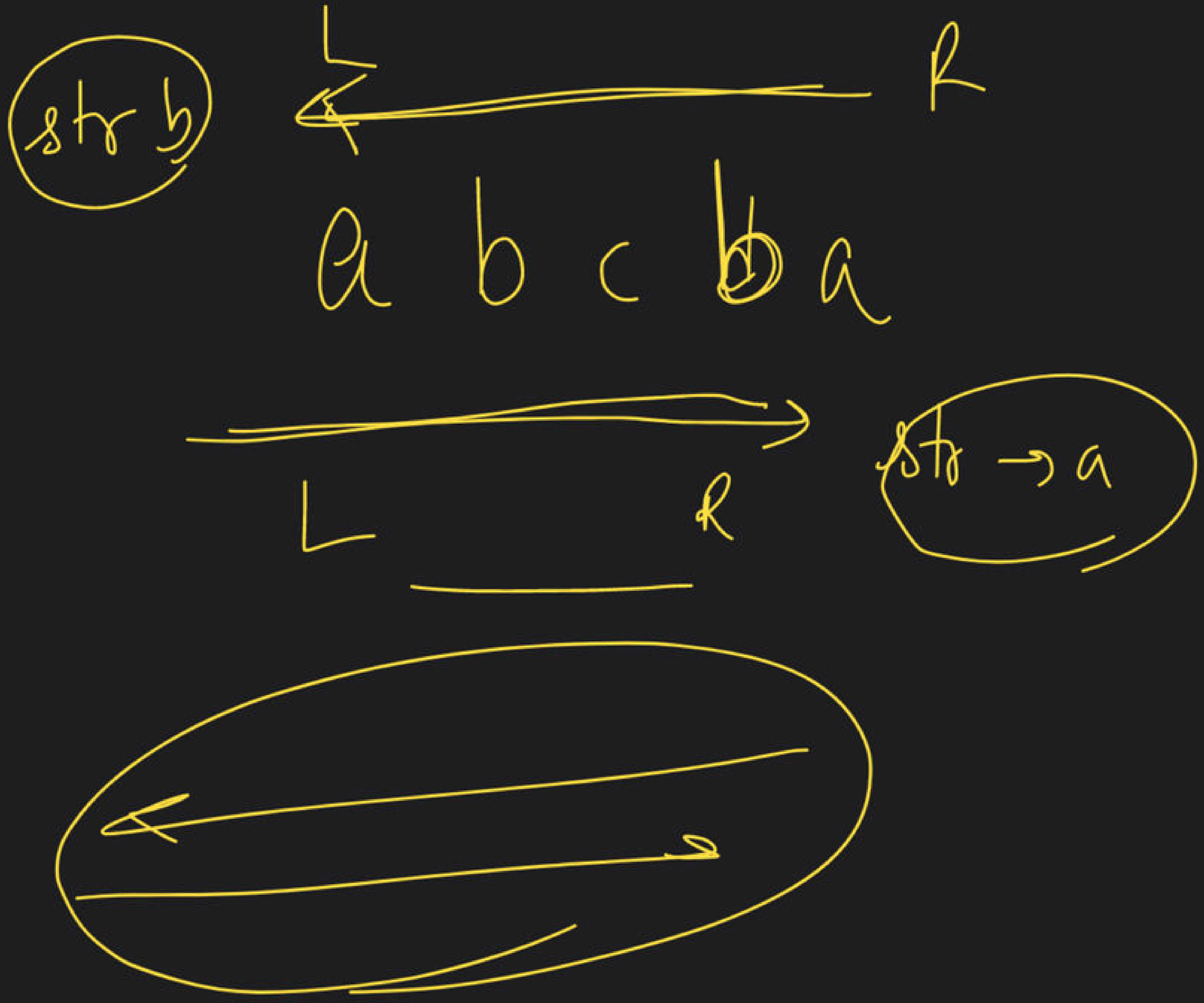
```
if (char match)
    rec(i+1, j+1)
else
    min( rec(i, j+1), rec(i+1, j) )
```

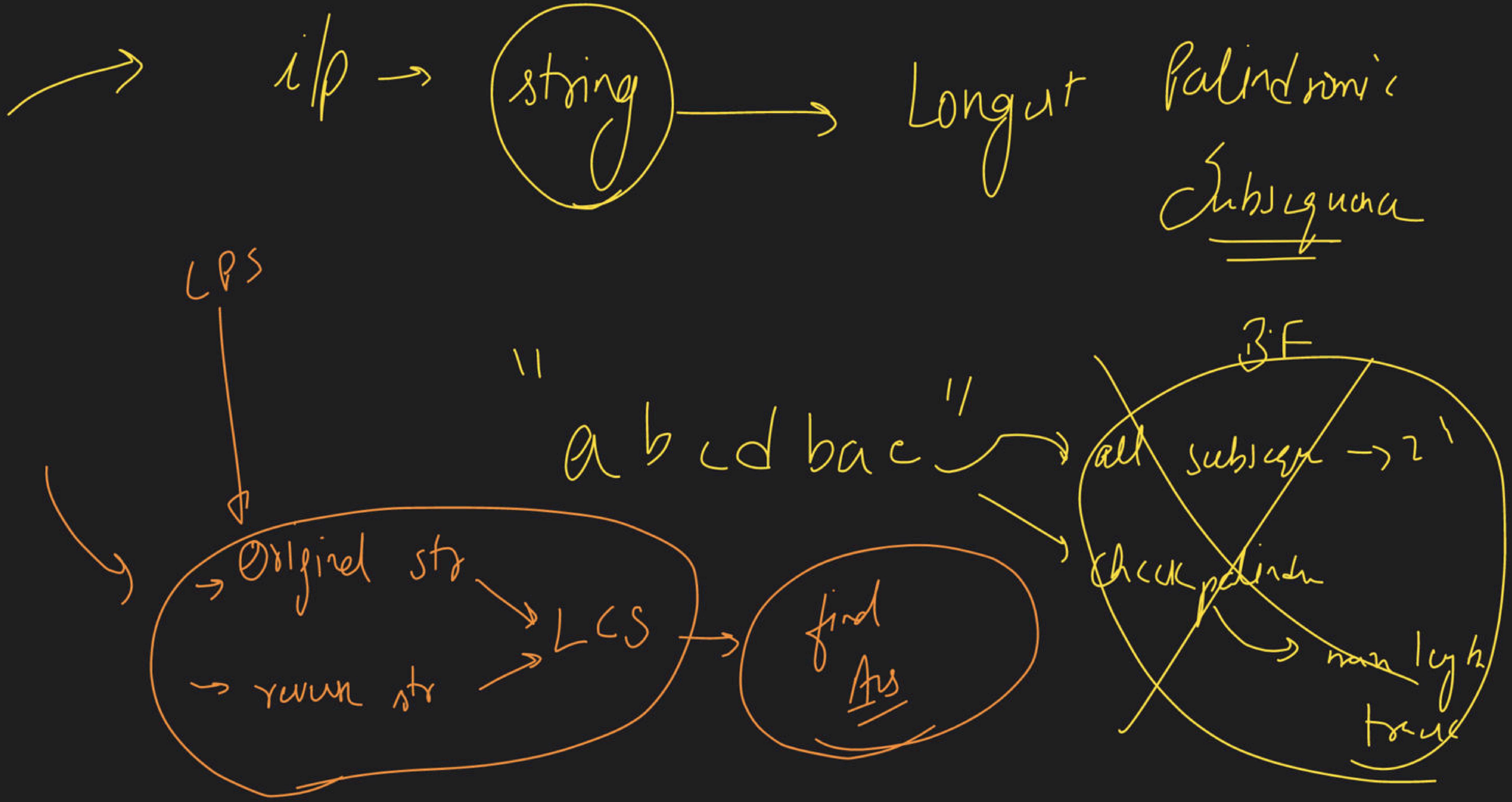

dp[i][j]
dp[i][j+1]
dp[i+1][j]
dp[i+1][j+1]
curr
next
~~2D~~
n×m
2 arrays
O(2m)

Palindromic Subsequnce

i/p -> str




str b
L
R
a b c b a
str → a
L
R

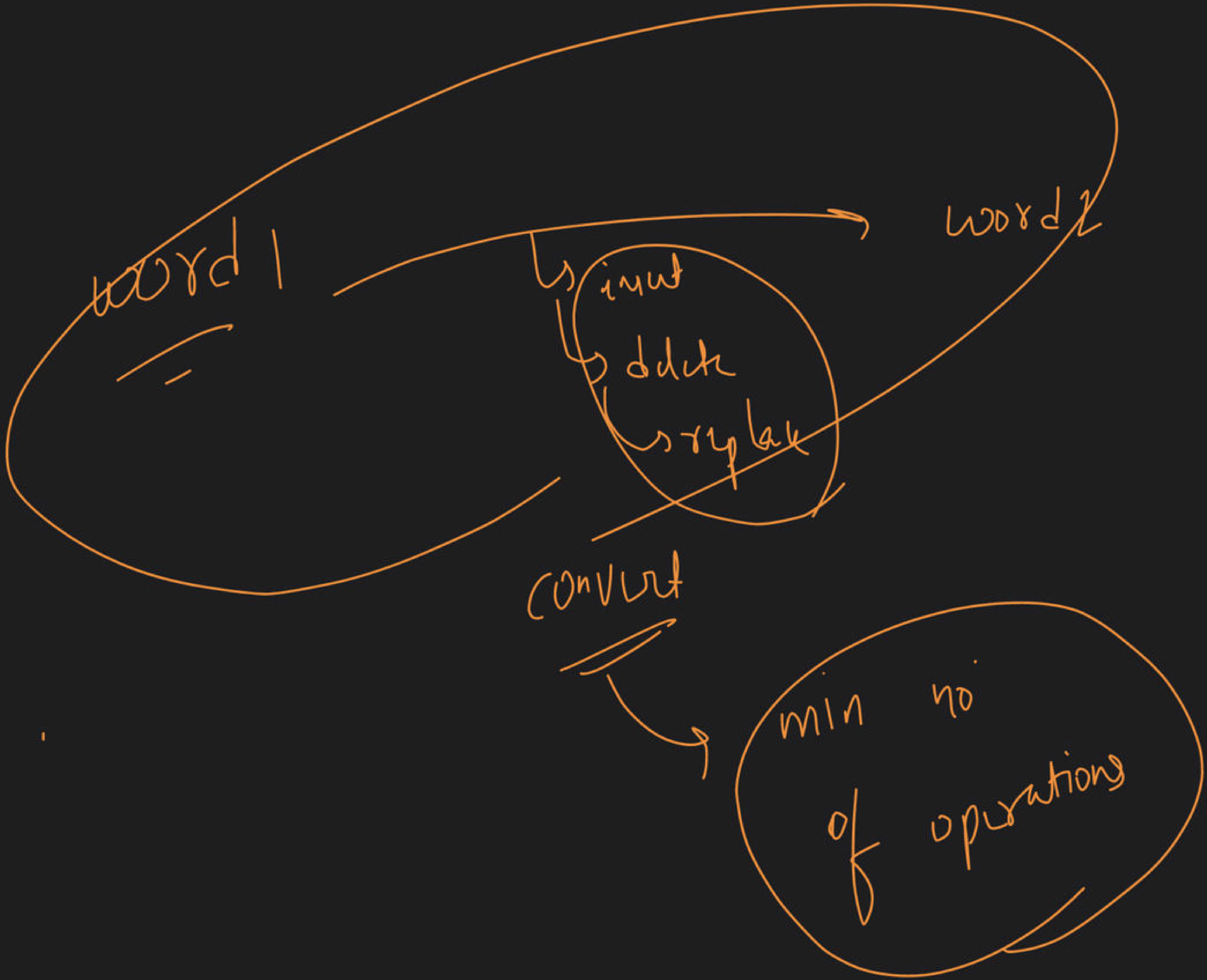
i/p → string → Longest Palindromic Subsequence
LPS
"abcdbac"
B.F
all subseq → 2^n
check palindrome → max length track
Original str
reverse str
LCS
find Ans

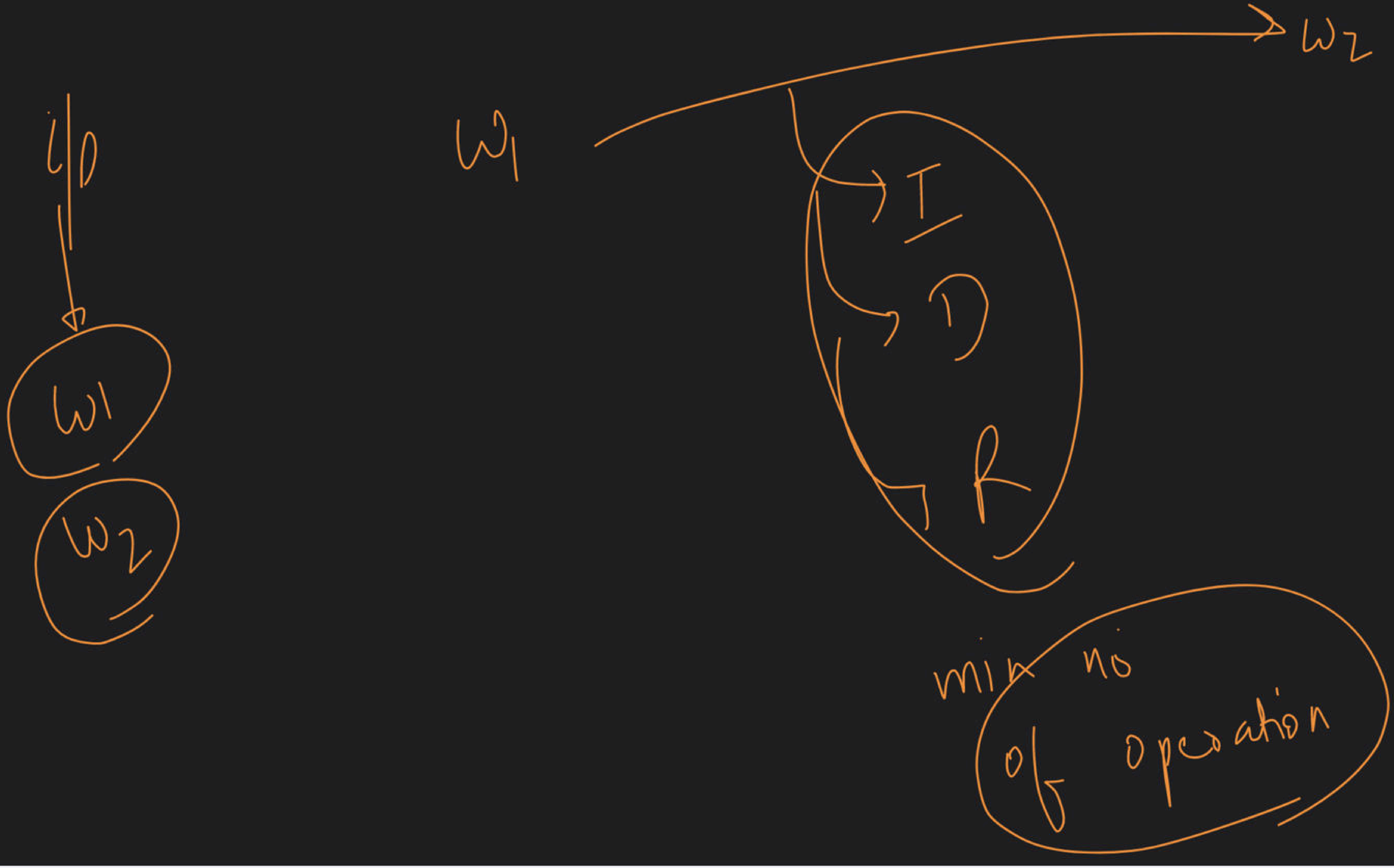
Dry Run

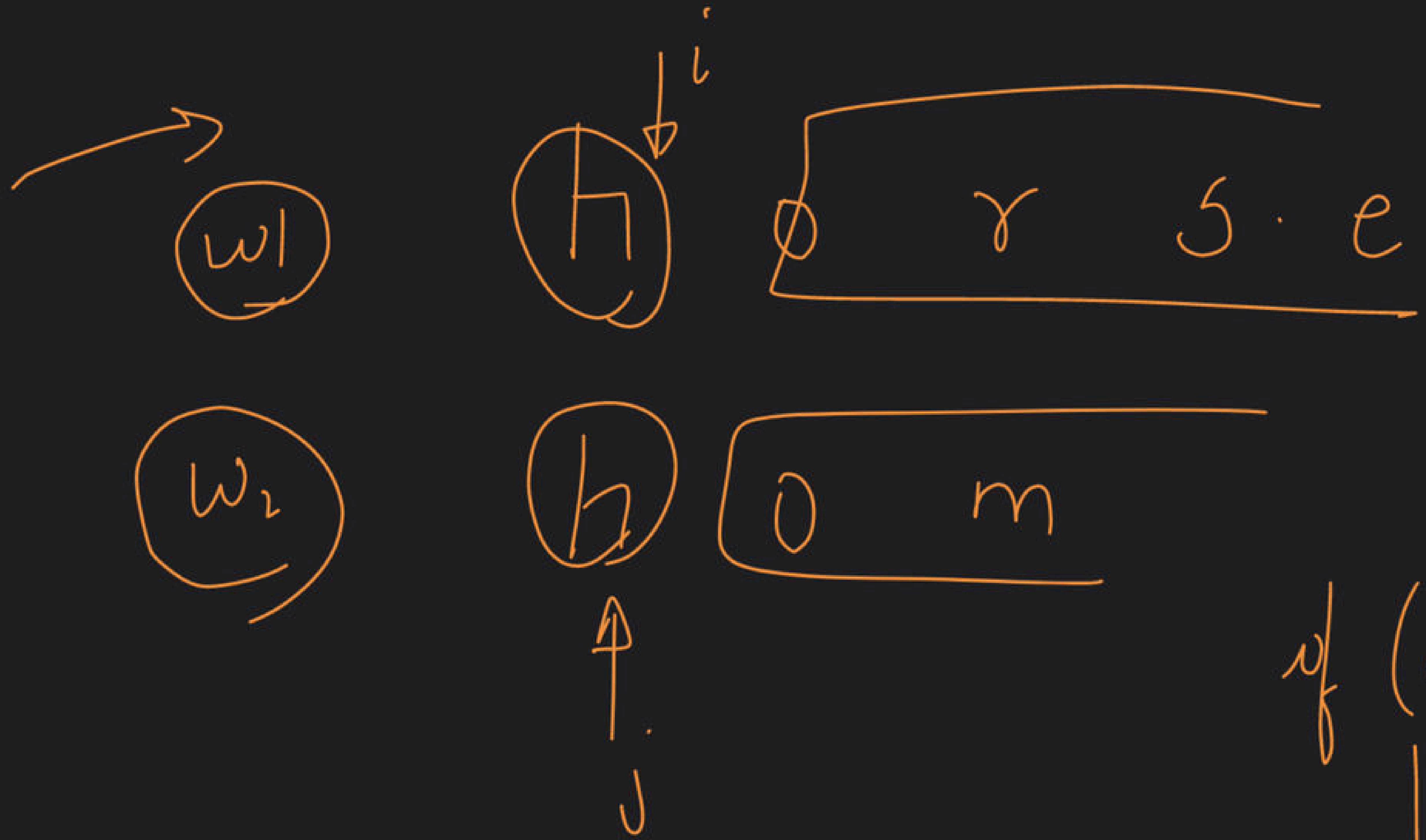
Loss?

Why

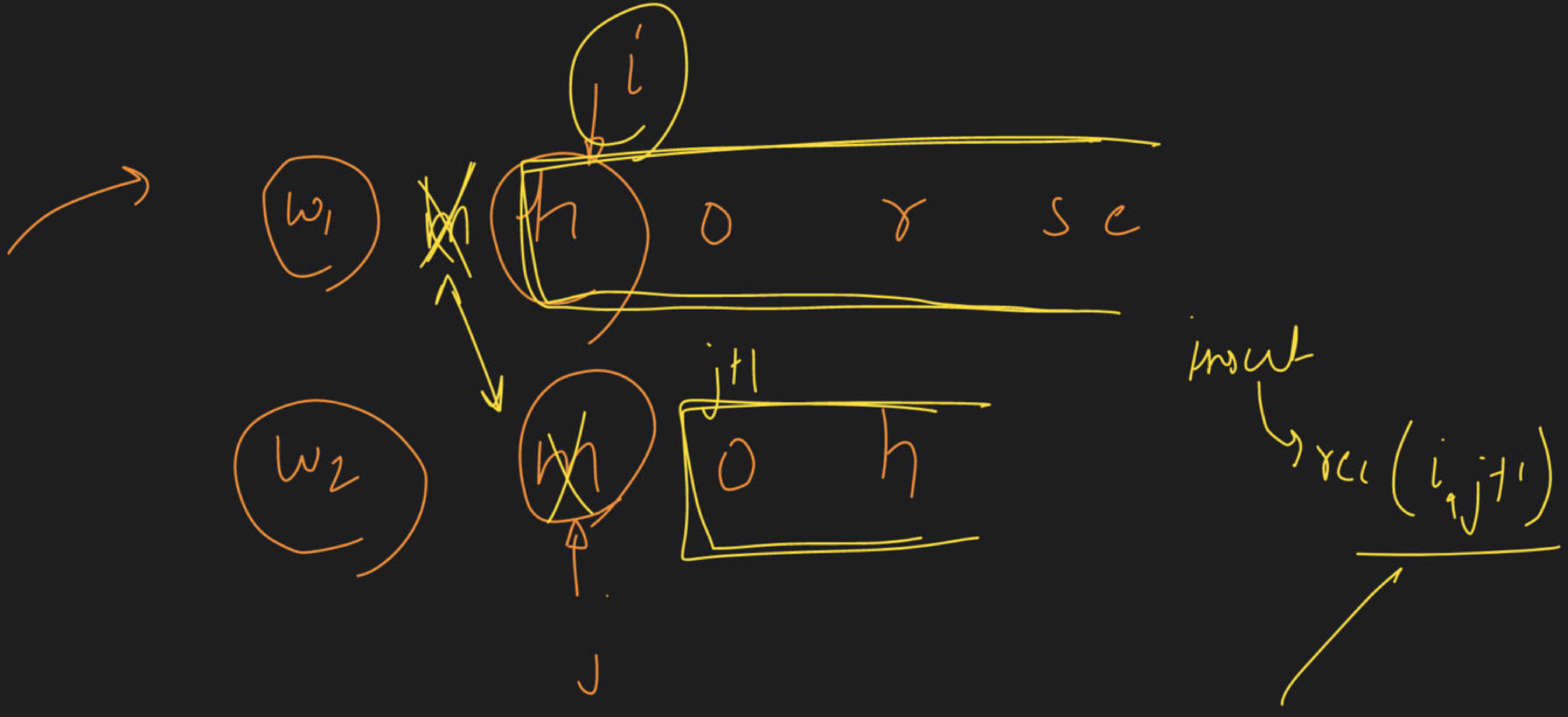
EDIT DISTANCE


word1
word2
insert
delete
replace
convert
min no of operations

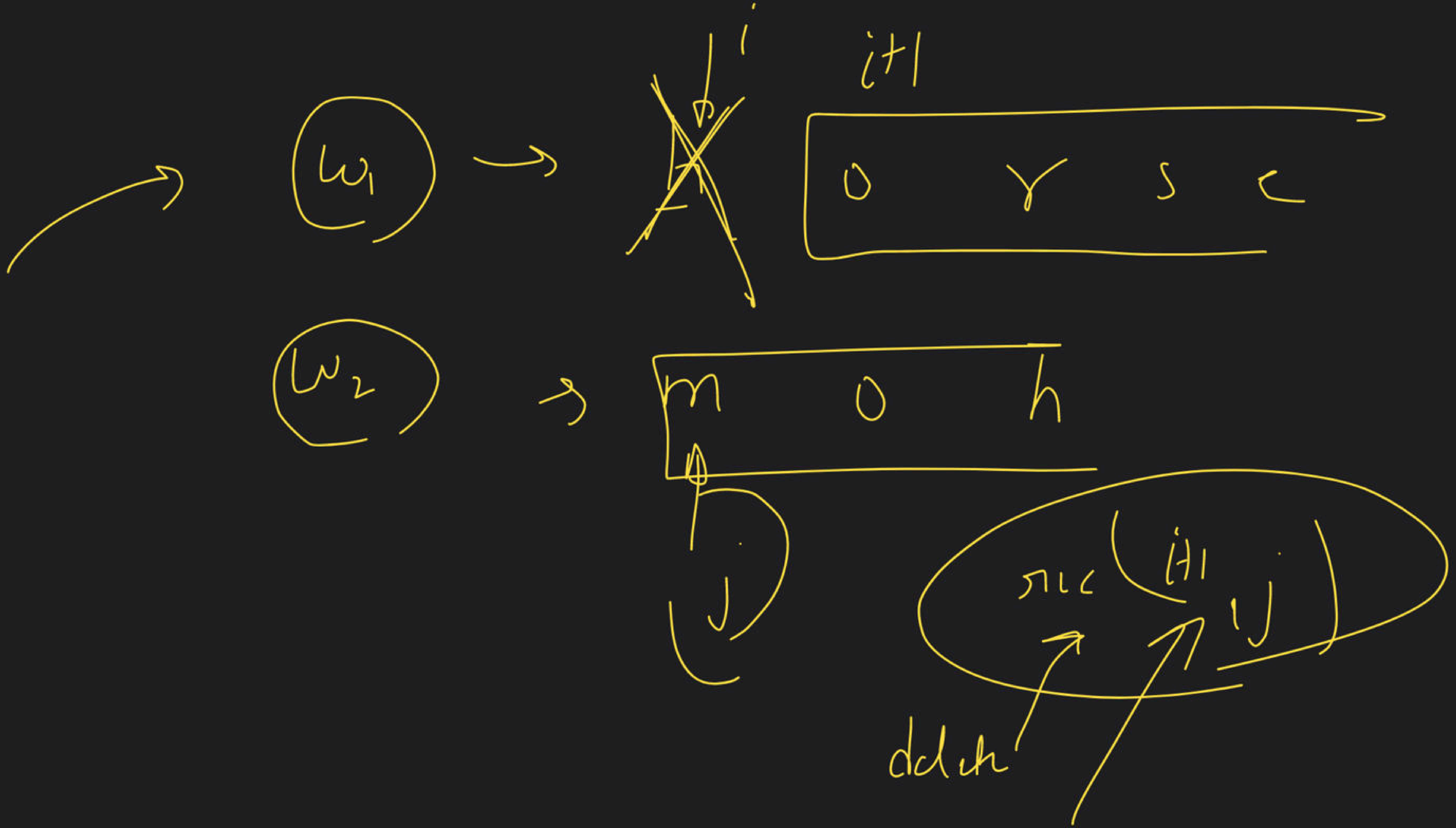

i/p
w1
w2
w1
w2
I
D
R
min no of operation

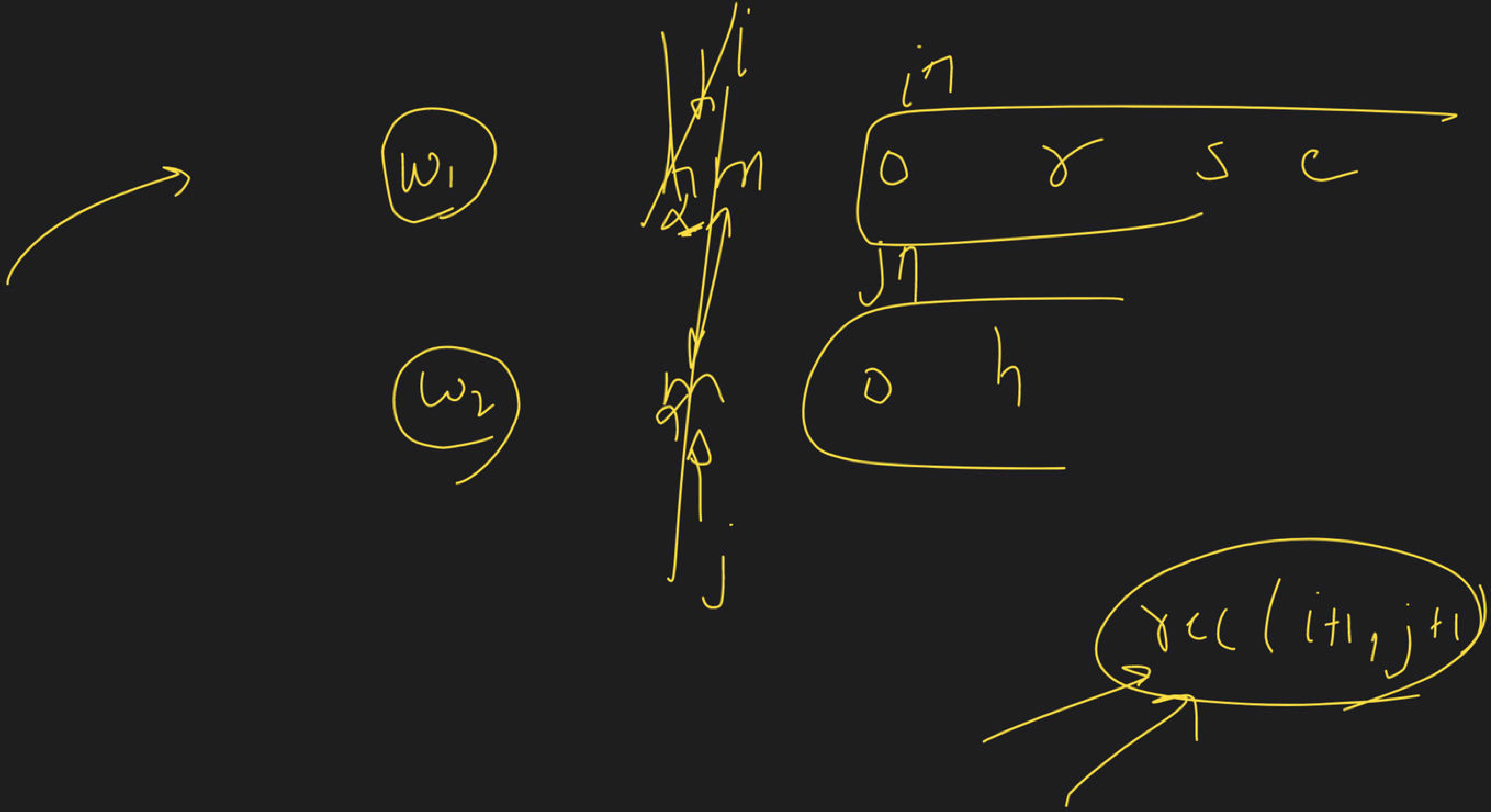


```
//char match
if (a[i] == b[j])
    ↳ rec(i+1, j+1)
else
```

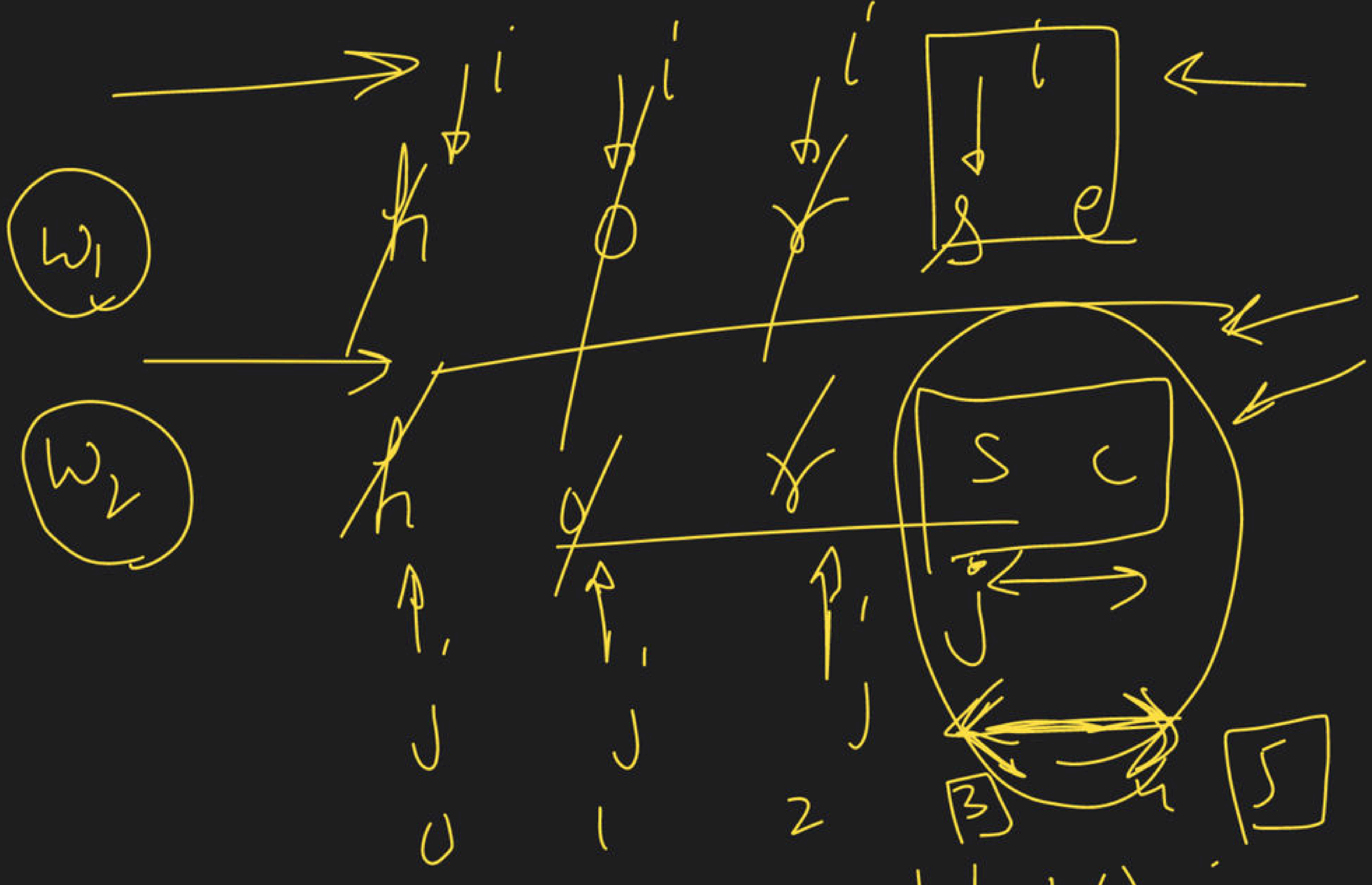


i
w1
~~h~~
h o r s e
j+1
w2
~~m~~
o h
j
insert
rec(i, j+1)


w_1
i
i+1
~~A~~
o r s e
w_2
m o h
j
rec (i+1, j)
delete


w1
w2
i
j
o r s c
o h
rec(i+1, j+1)

$i == a.length()$

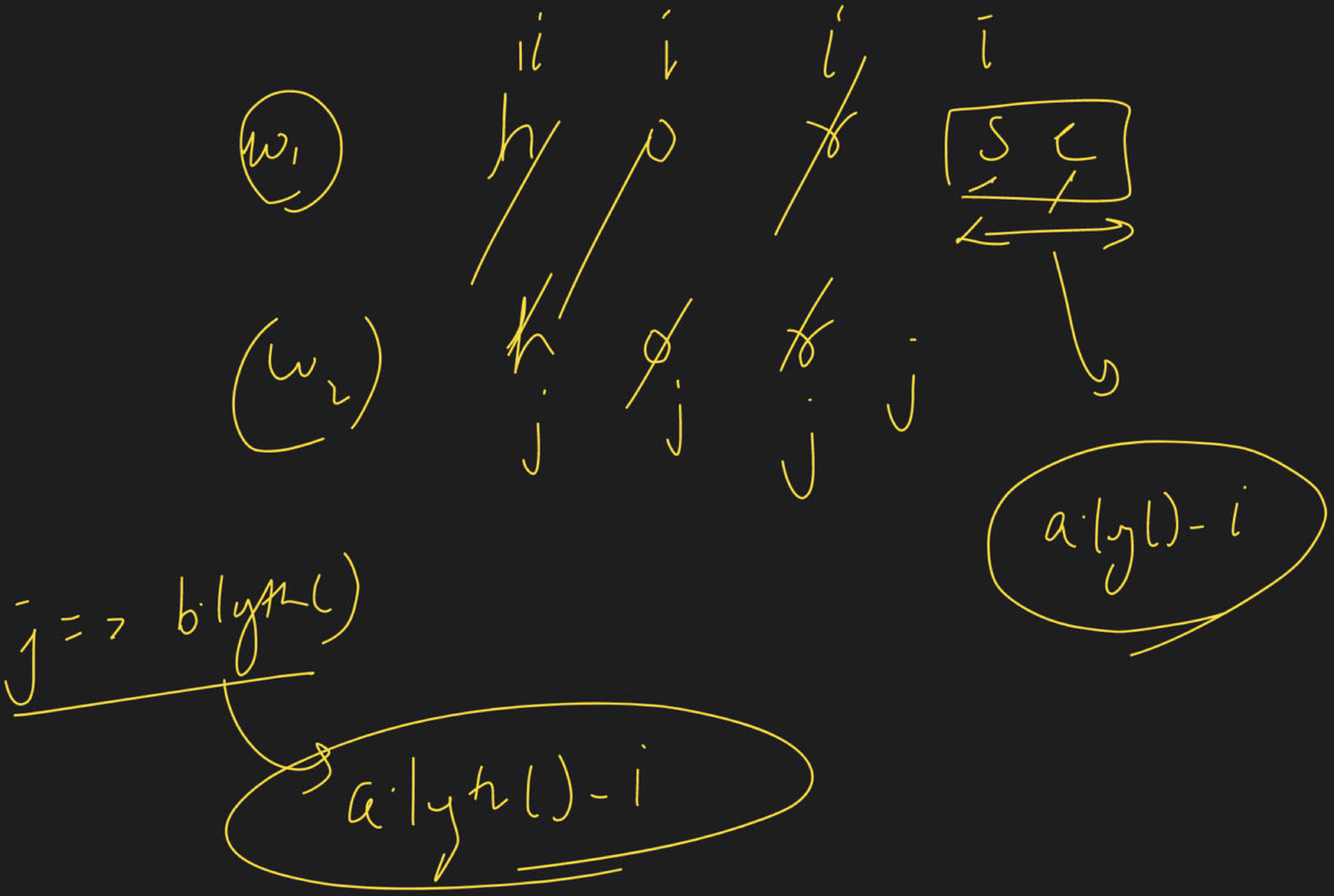


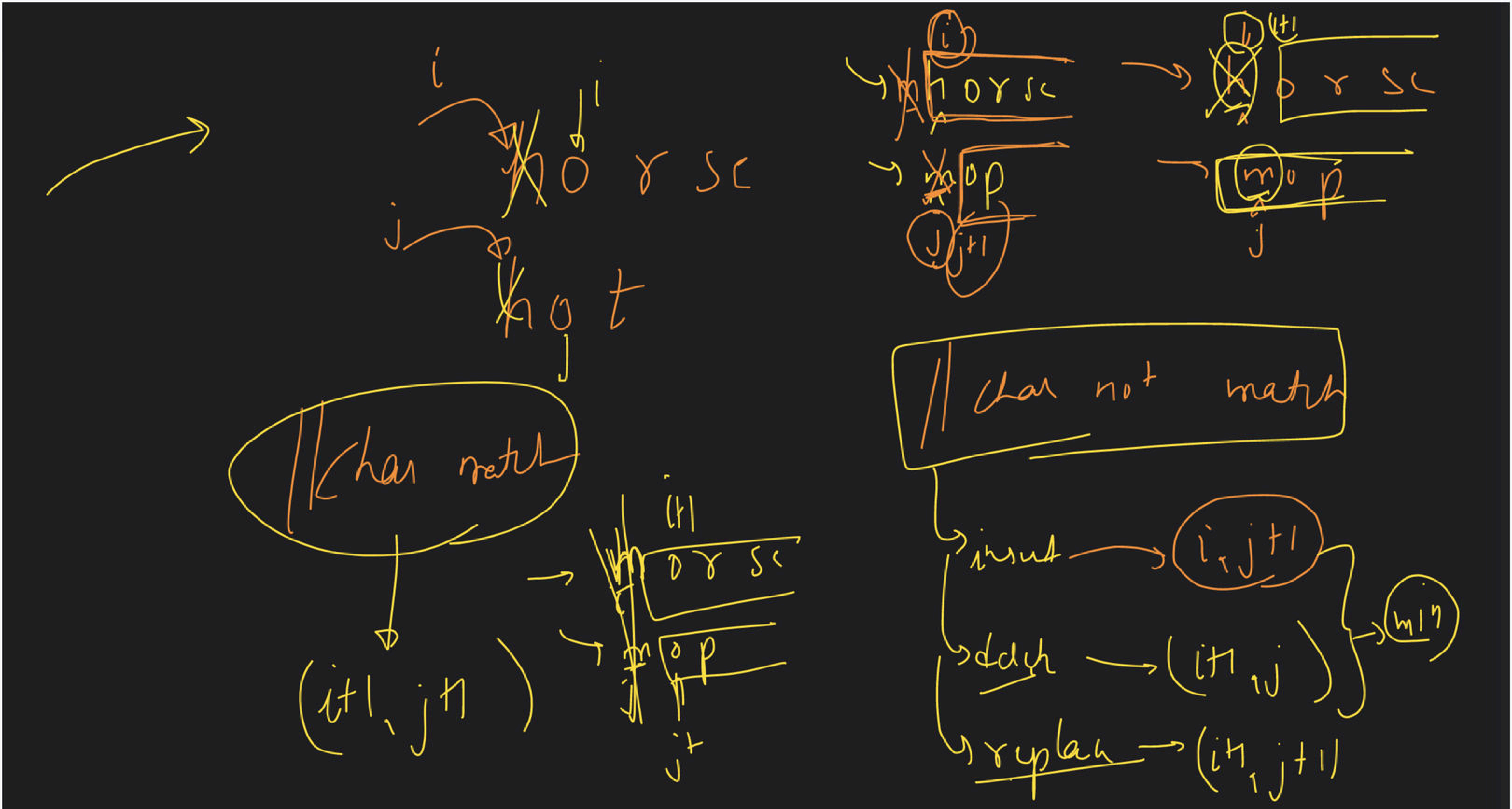
$i == a.length()$

$b.length() - j$

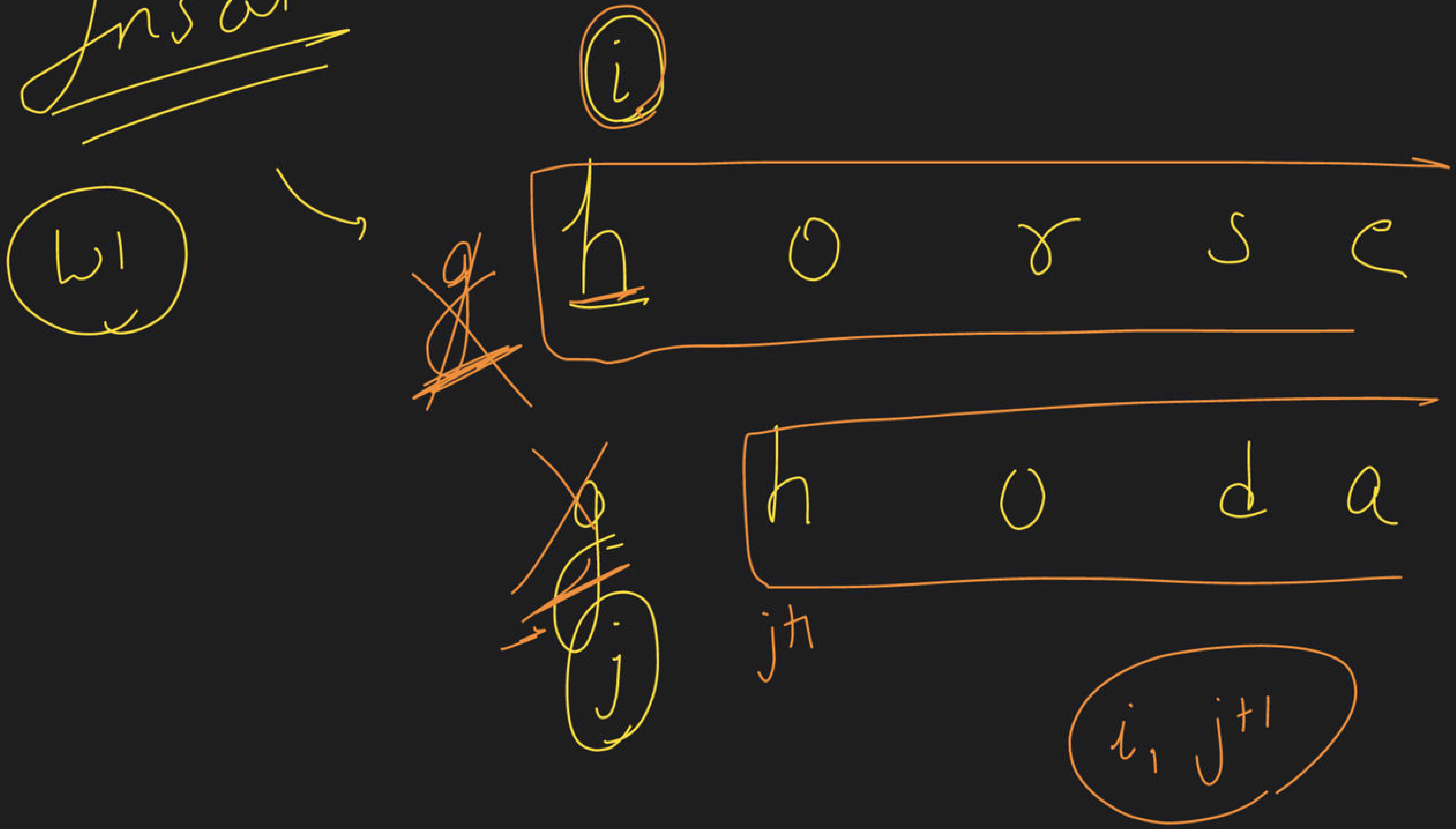
$b.length() - j$

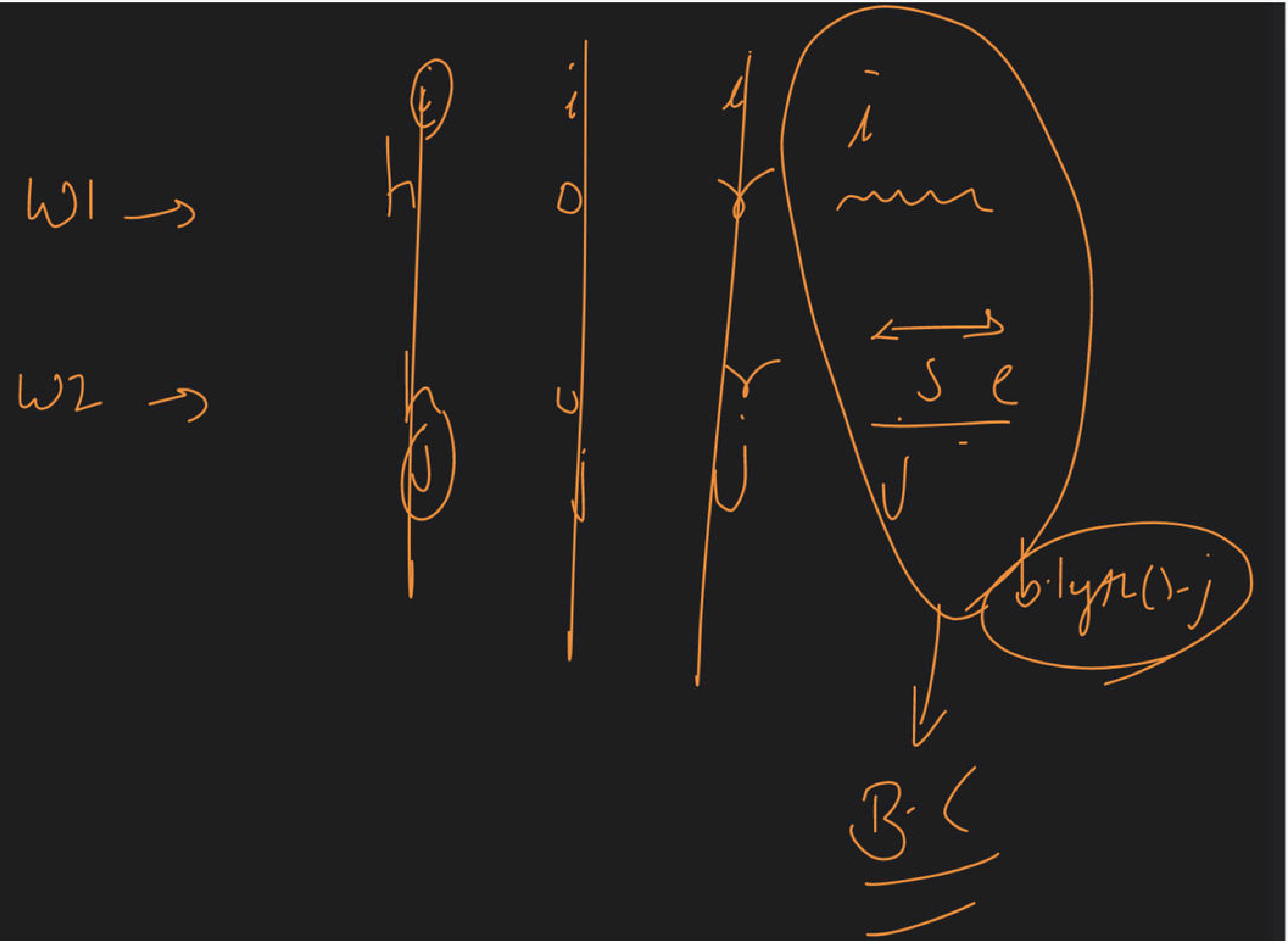
$5 - 3 = \boxed{2}$

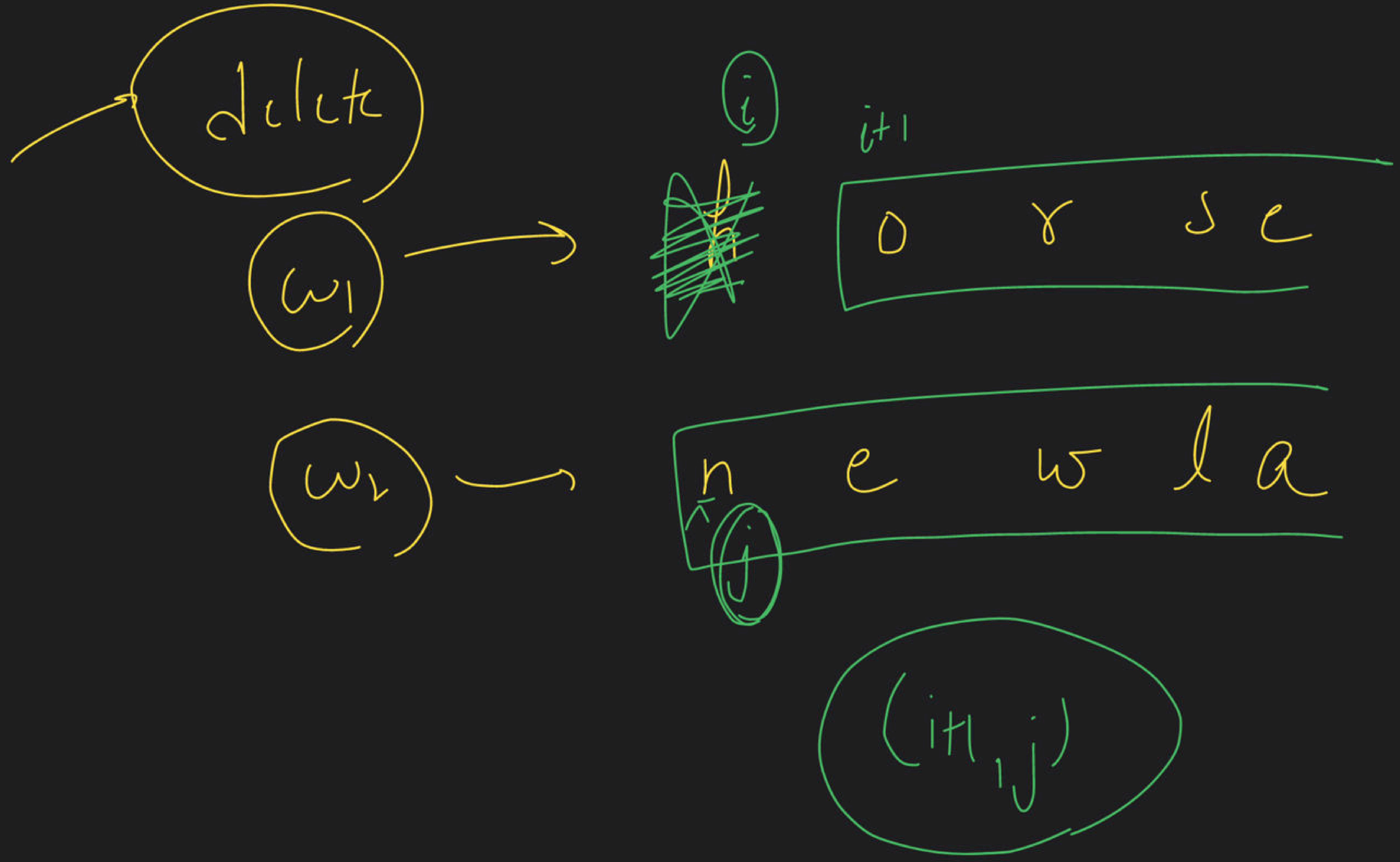

i i i i
w1 h o r s c
w2 h o r
j j j j
a.lgl() - i
j ==> b.lgth()
a.lgth() - i

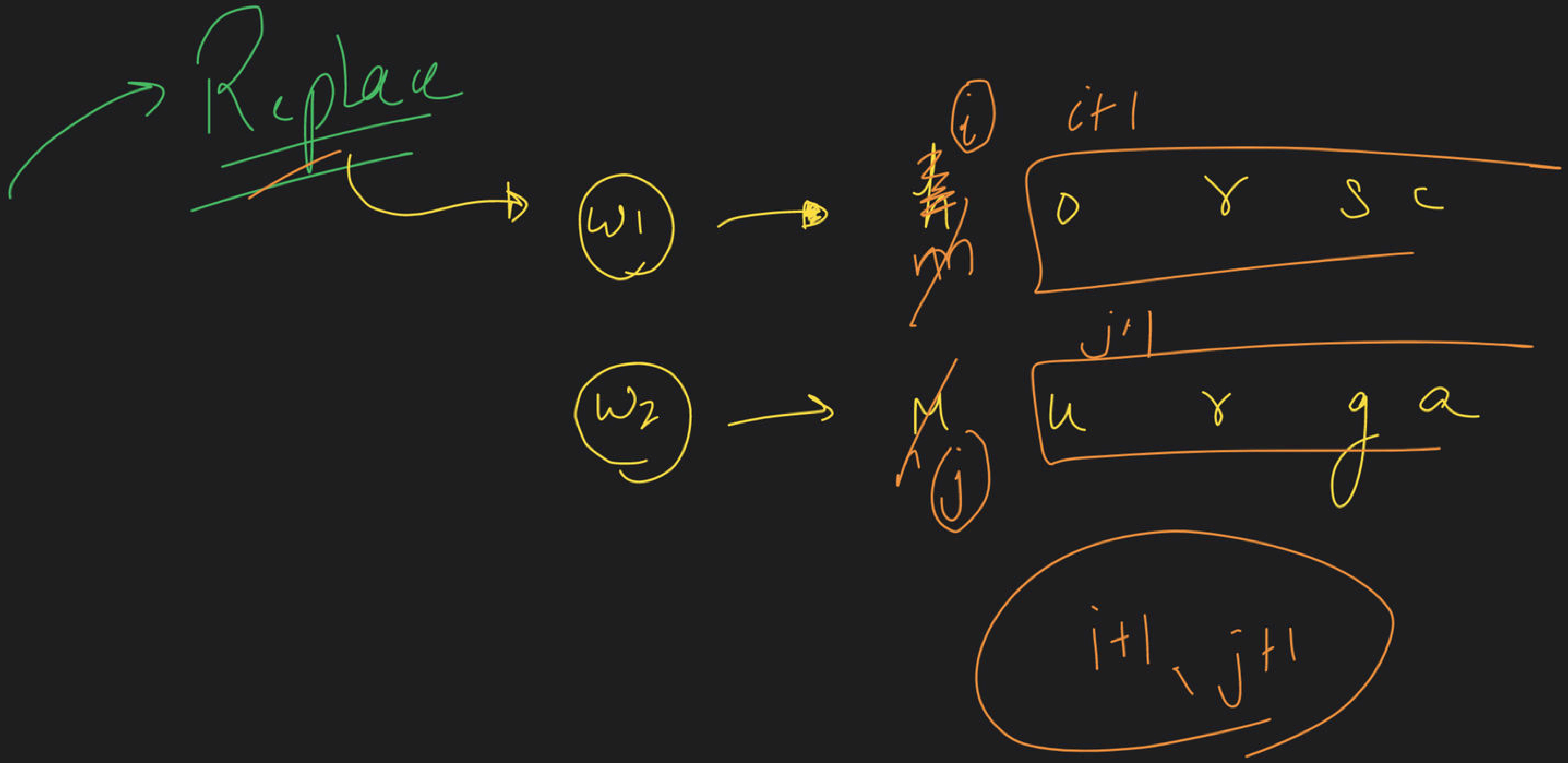

i
j
h o r s e
j
h o t
j
// char match
(i+1, j+1)
i+1
m o r s e
m o p
j+1
i
m o r s e
m o p
j
j+1
i
i+1
h o r s e
m o p
j
// char not match
insert → (i, j+1)
delete → (i+1, j)
replace → (i+1, j+1)
min

Insert

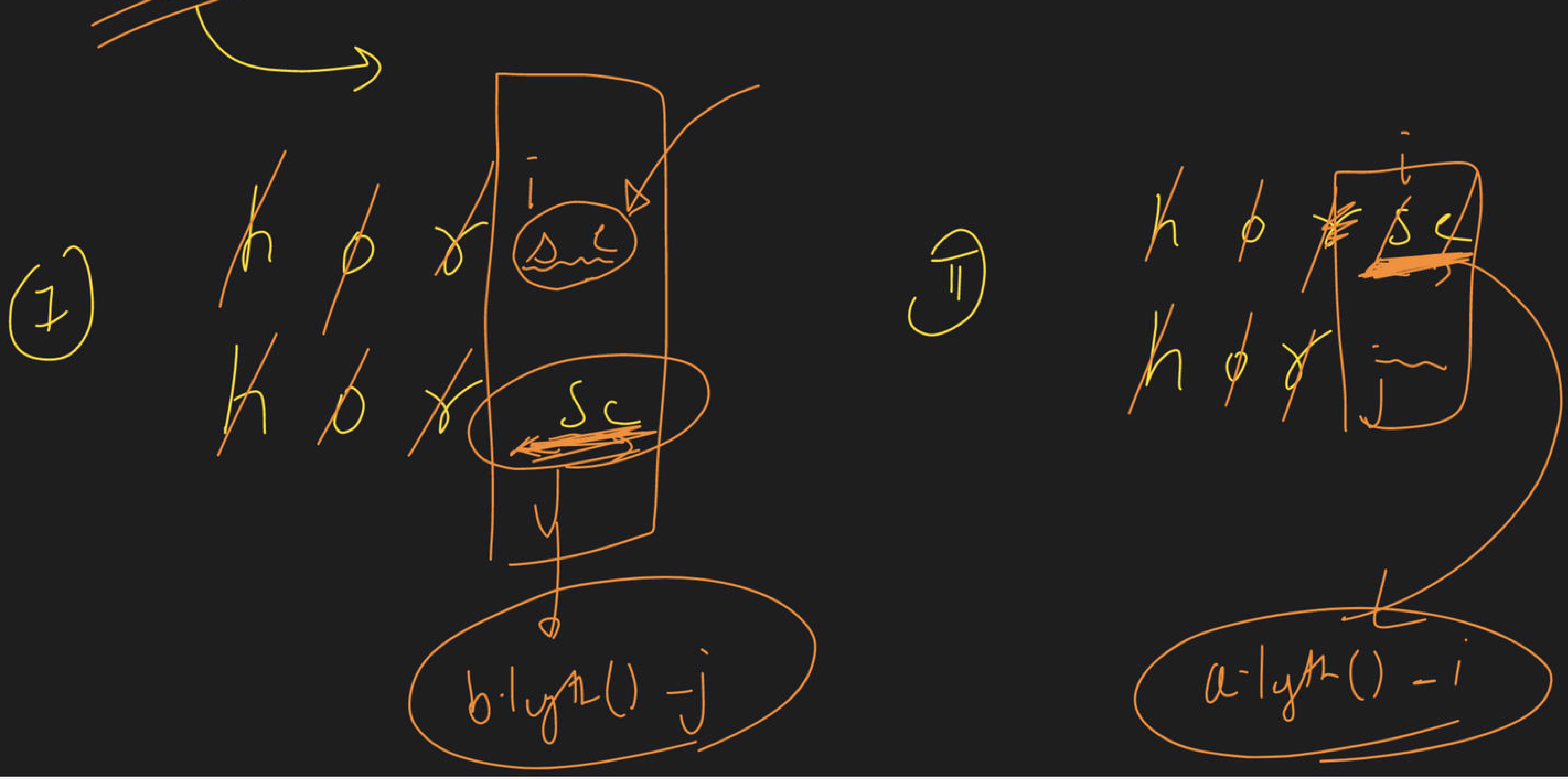


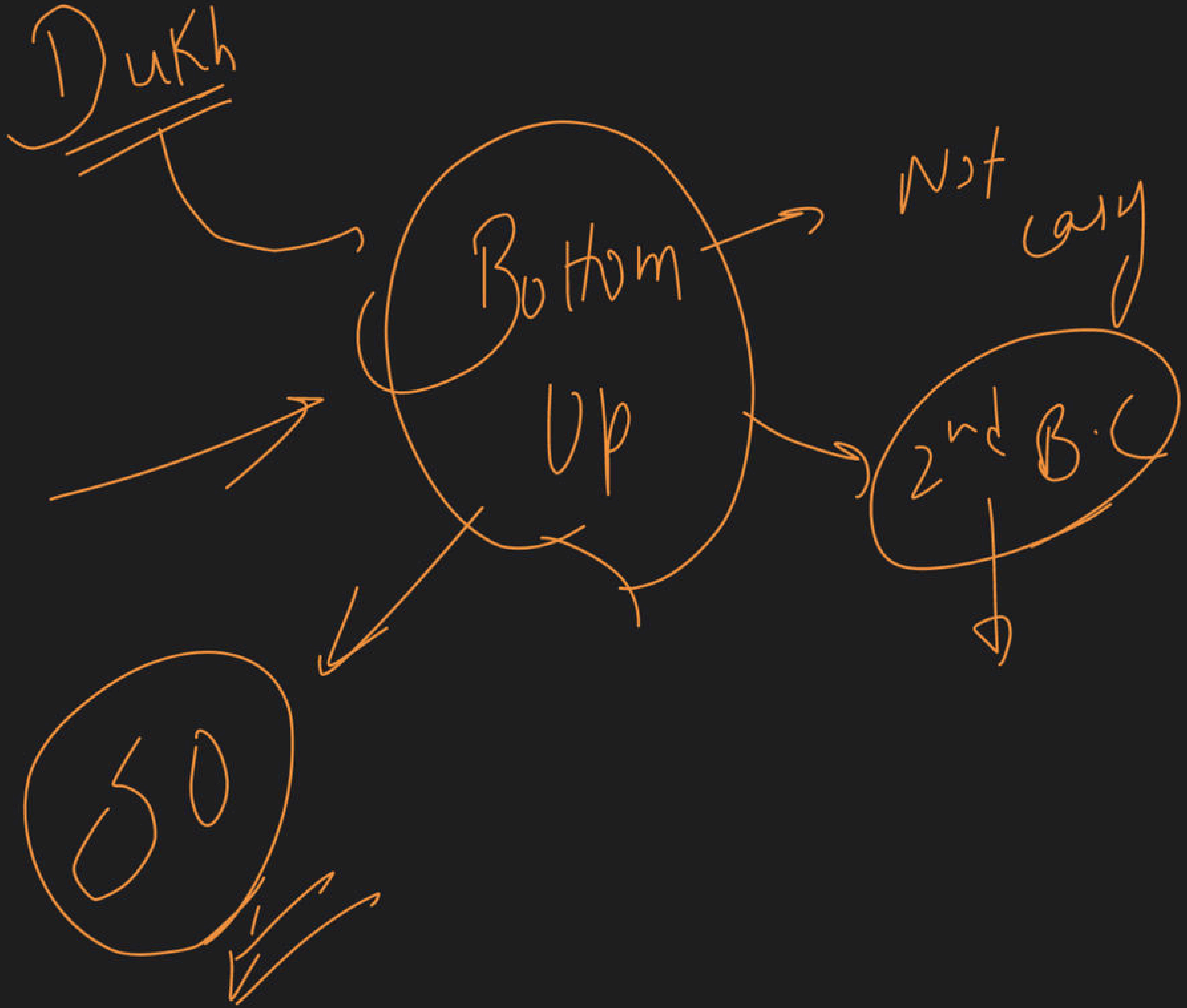

W1 →
W2 →
i
h
h
j
i
o
u
j
i
r
r
j
i
s e
j
b.lyn()-j
B.C

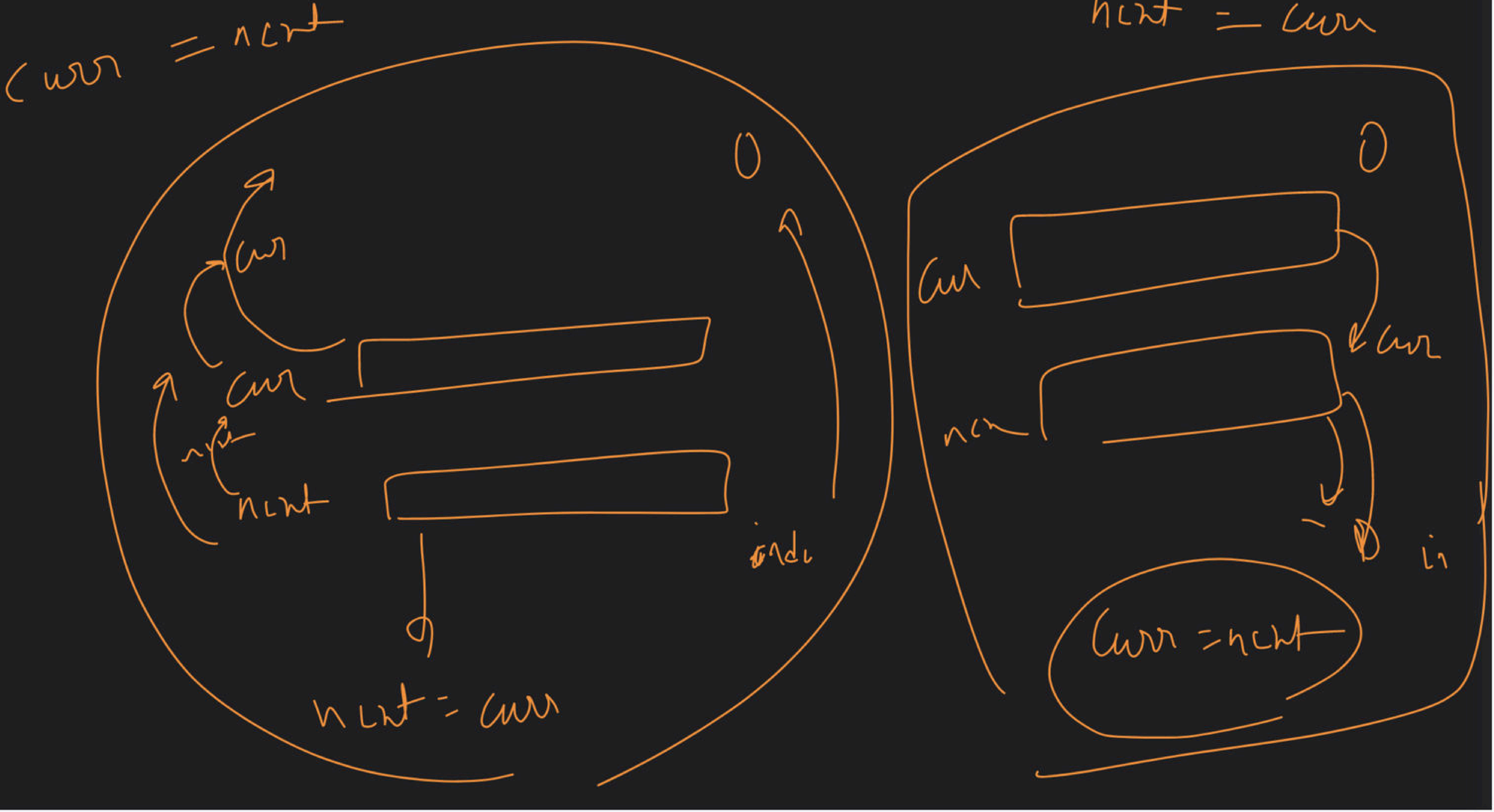

delete
w1
w2
i
i+1
h
o r s e
n e w l a
j
(i+1, j)


Replace
w1
w2
i
i+1
~~m~~
o r s e
j+1
~~M~~
u r g a
j
i+1, j+1

B.C




Dukh
Bottom Up
Not easy
2nd B.C
50



curr = next
next = curr
0
curr
curr
next
0
index
next = curr
0
curr
curr
next
0
li
curr = next