[-1, 0, 1, 2, -1, -4]

sorting

-4, -1, -1, 0, 1, 2

0

0, -1, 1

0

numbers

2, -1, -1

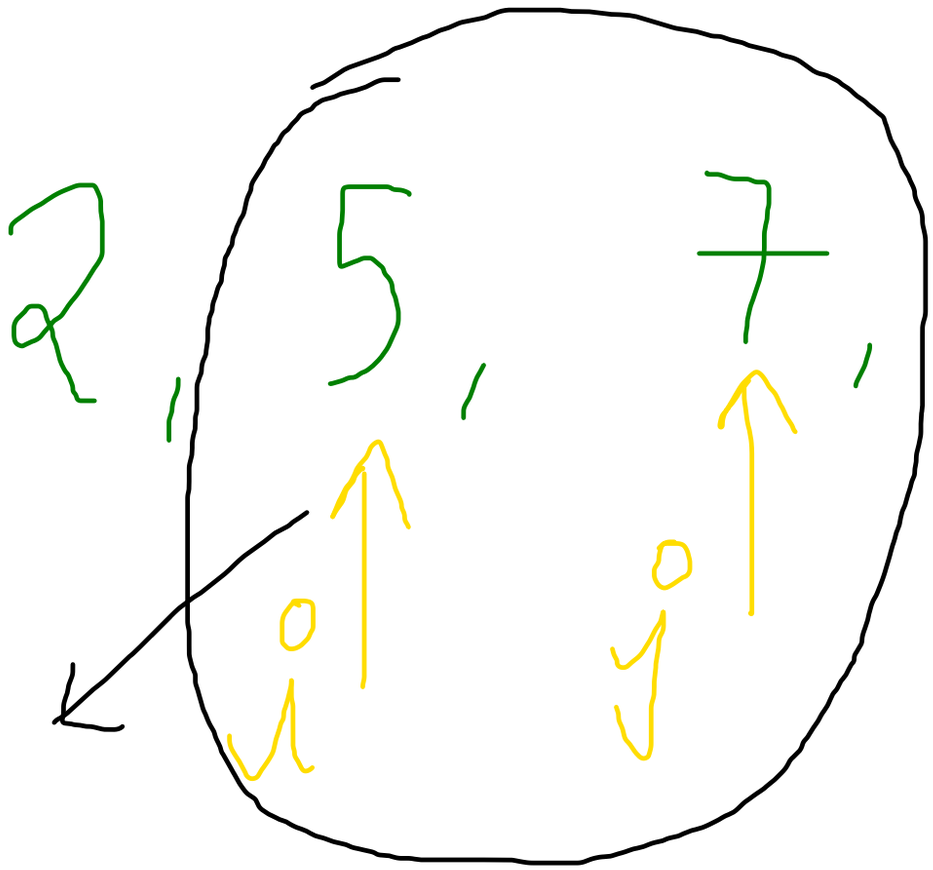
0

order change

-1, 0, 1

0

Sorted array



12

12

14

14 > 12

-4 | -1, -1, 0, 1, 2

i

j

a + b + c = 0

-4 + (a + b) → +4 → 0

-1 + (a + b) → +1

-4, -1, -1, -1, 0, 1, 2, 2

fixed

a

skipping

i

j

-4, 2, 2

-1 + a[i] + a[j]

↳ 1

-1, -1, 2

-1, -1, 2

[-4, -1, -1, -1, 0, 1, 2, 2] → a

i

skipping

int prev = -1;

while (a[i] == prev)
{
i++;
}

pI

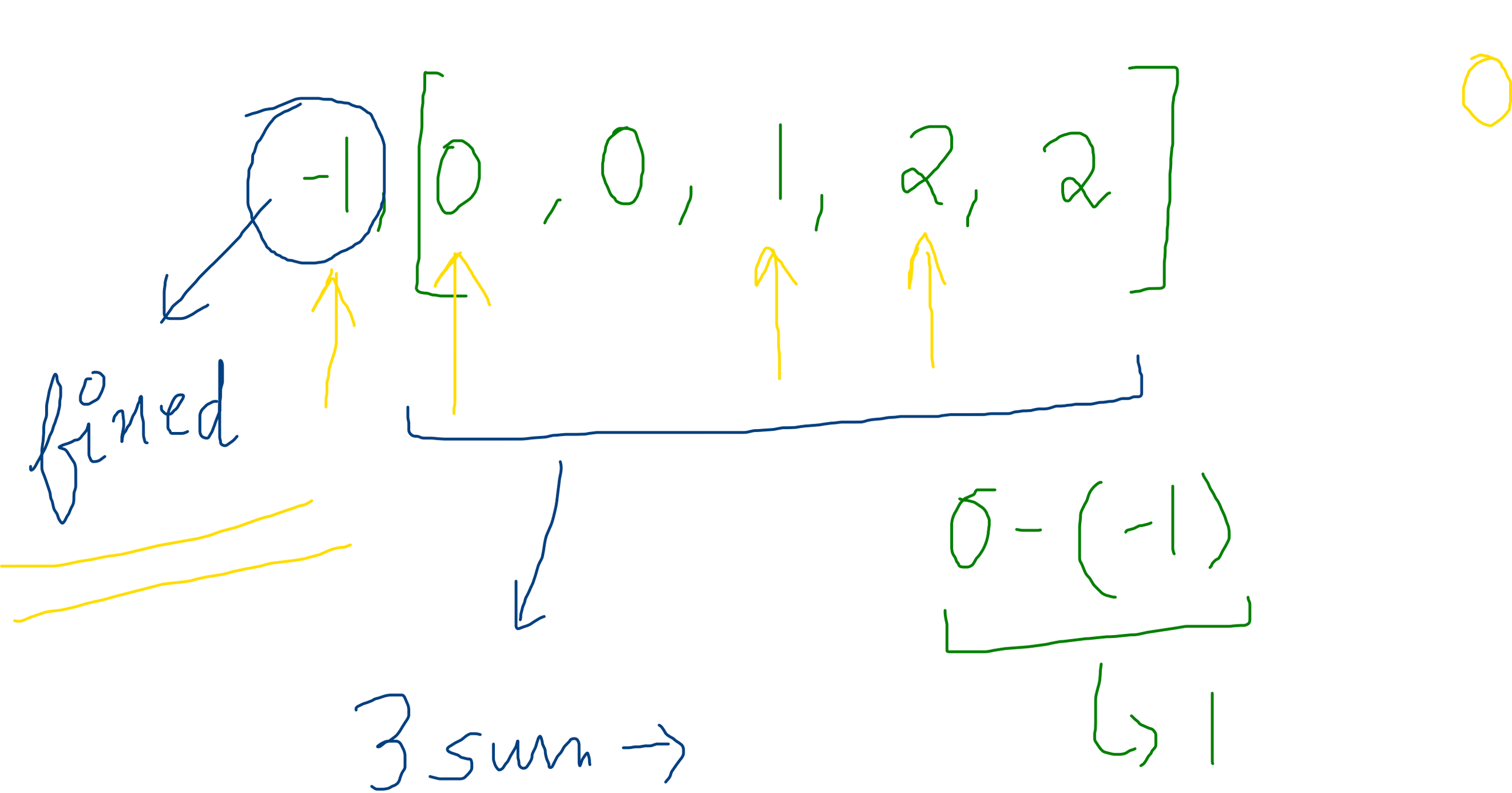
pJ

-4, -1, -1, -1, 0, 1, 2, 2, 2

i

j

k


-1
[0, 0, 1, 2, 2]
fixed
3sum →
0 - (-1)
↳ 1

abc

bac
cab
cba
bca
acb

26

aabc

baac

a → 2
b → 1
c → 1

a → 2
b → 1
c → 1

0	1	2	3	4	5
2	1	1	0	0	0

a → 0
b → 1
c → 2
z → 25

abcabd

'a' - 97

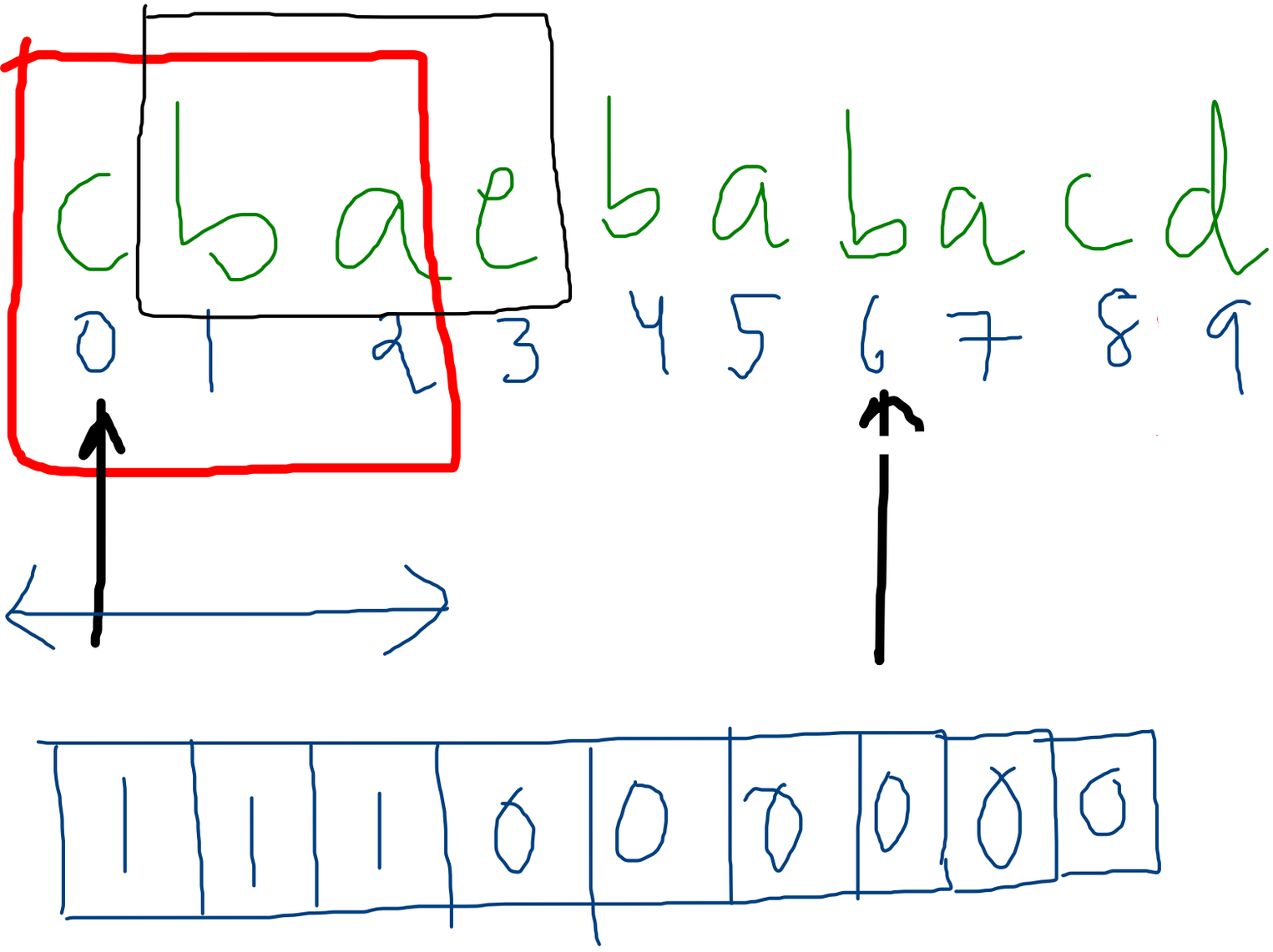
0

26

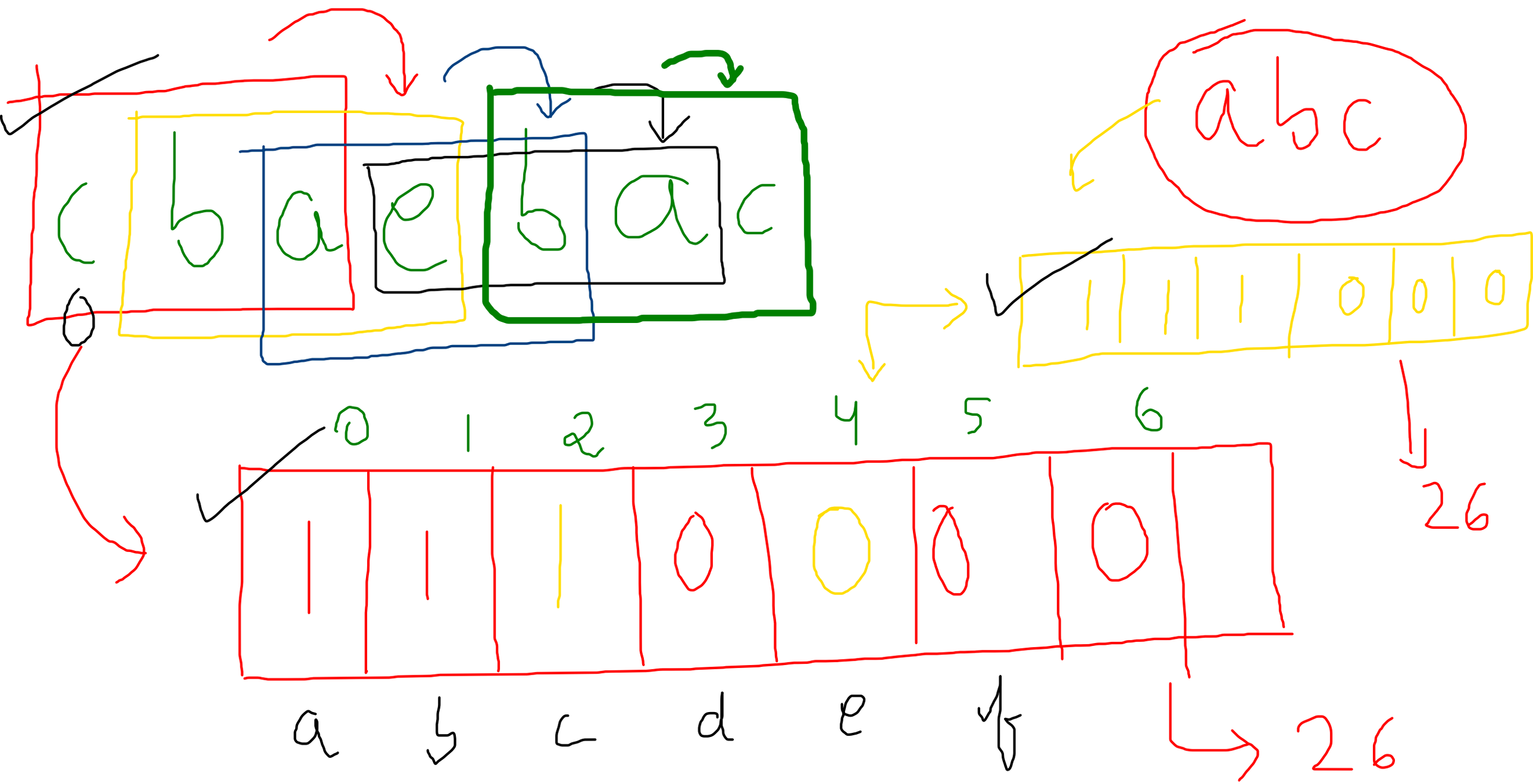
2	2	1	1	0	0	0	0

1

'b' - 97


cbaebabacd
0 1 2 3 4 5 6 7 8 9


abc
[0, 6]
↳ fe


cbaebac
0
abc
1 1 1 0 0 0
26
0 1 2 3 4 5 6
1 1 1 0 0 0 0
a b c d e f
26

a b a b a

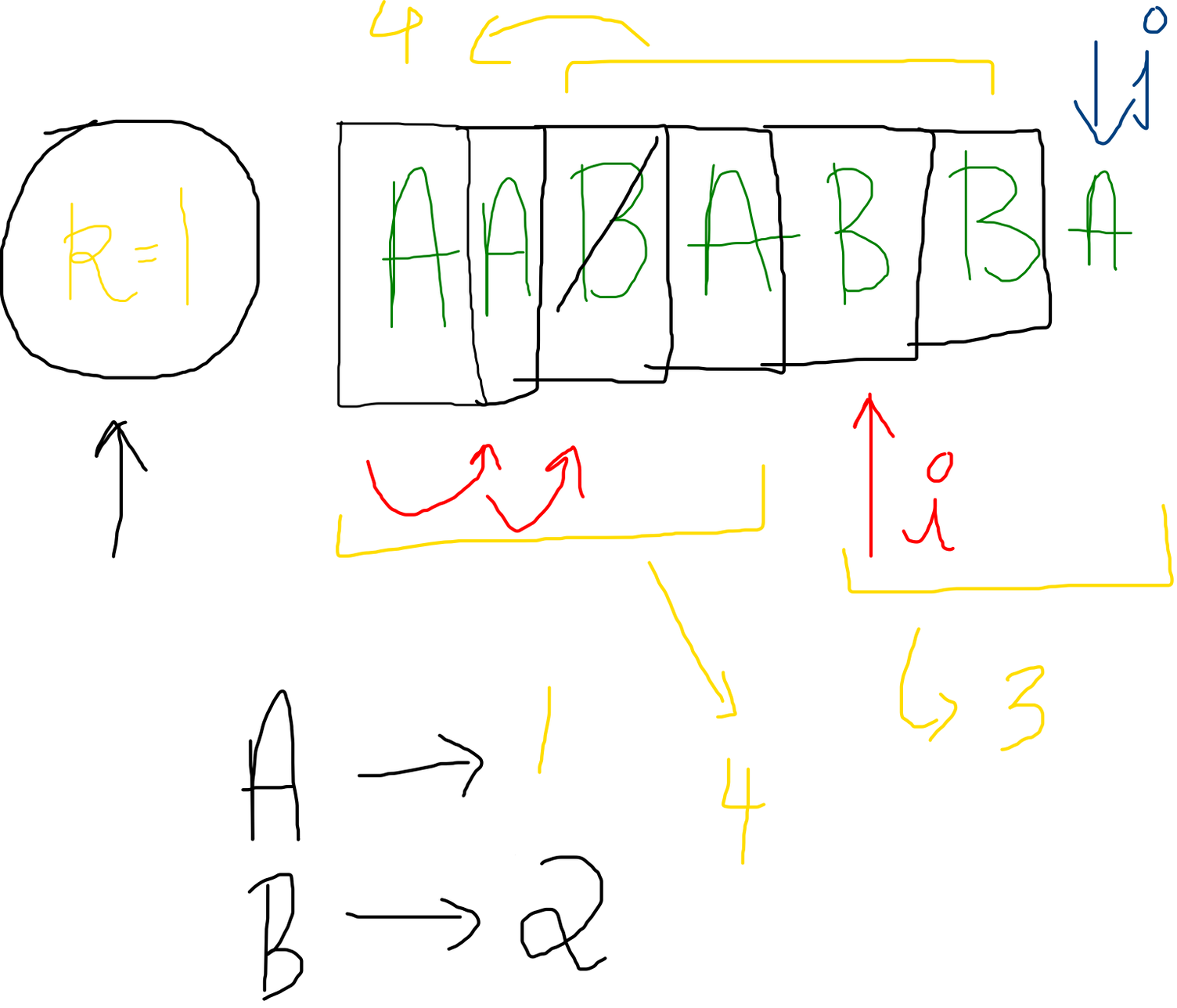
a - 3

b - 2

3	2	0	0	0	0
0	1	2	3		

↳ 26

a - 'a'



i → ending, inc

j → starting, dec

A → 1

B → 2

3

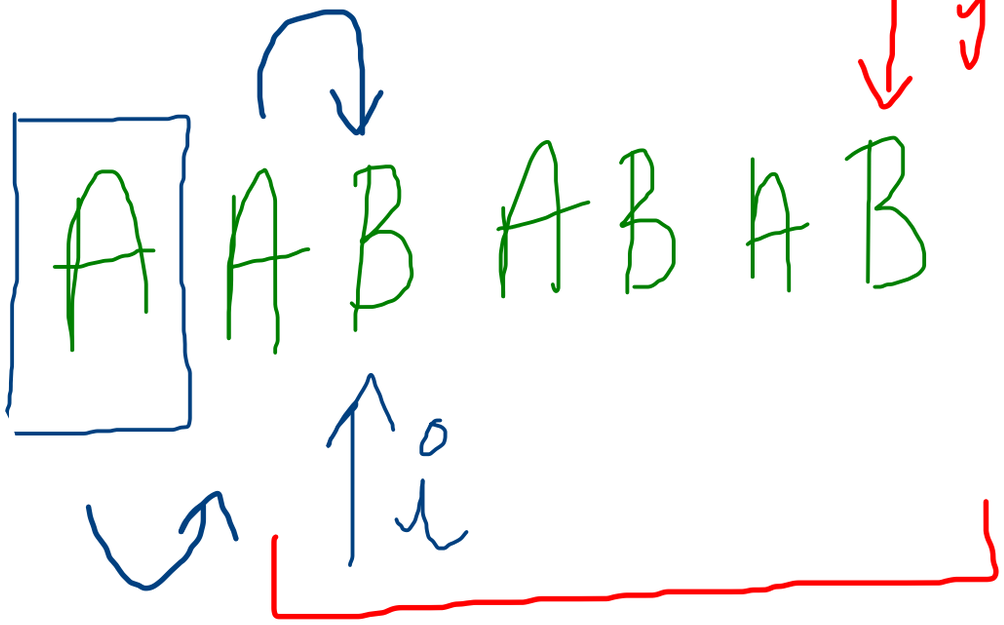
4

current → 3

longest → 1 / 4

A → 2

B → 3



k = 2

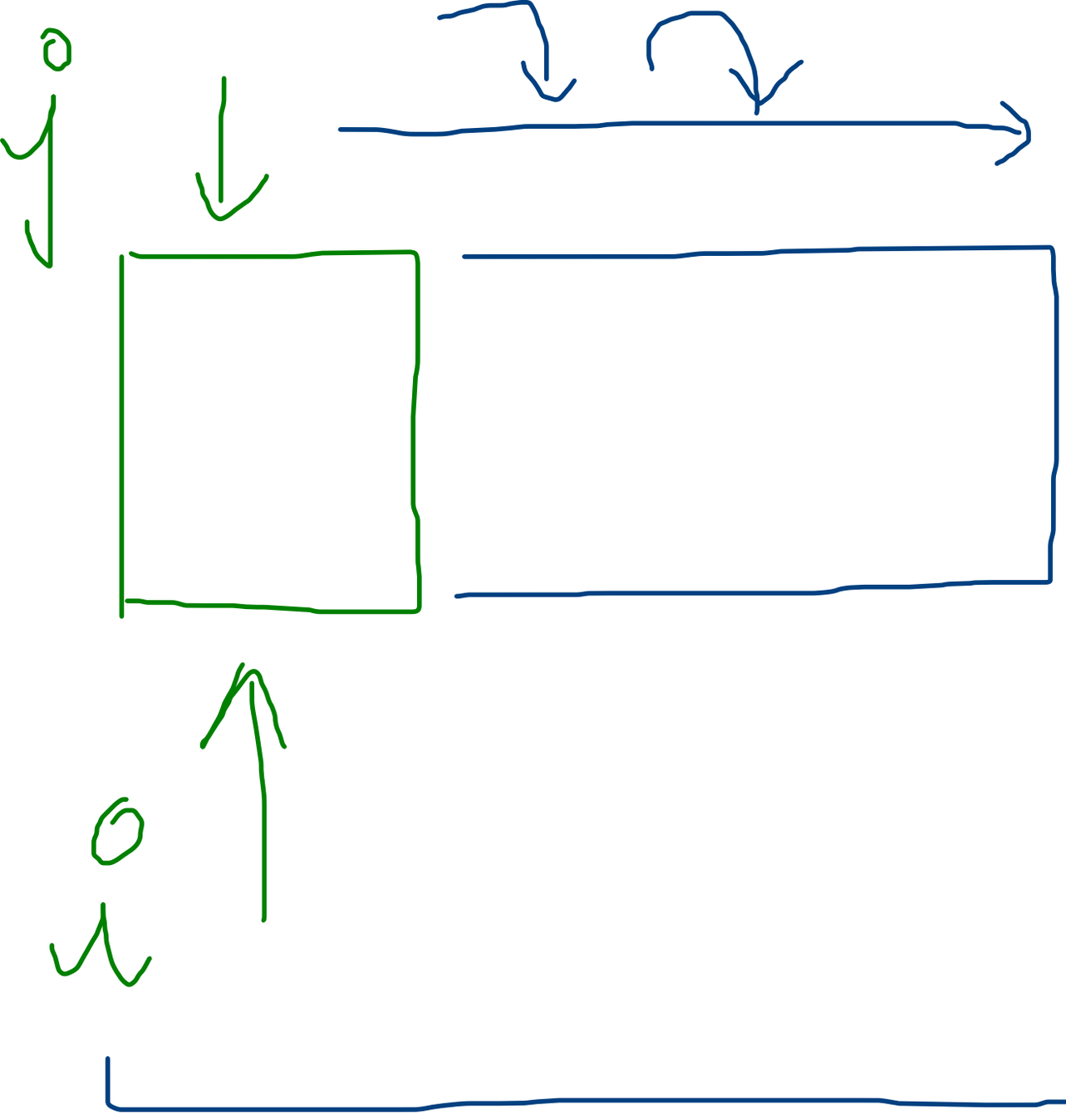
current → j - i + 1 → 6

i → starting

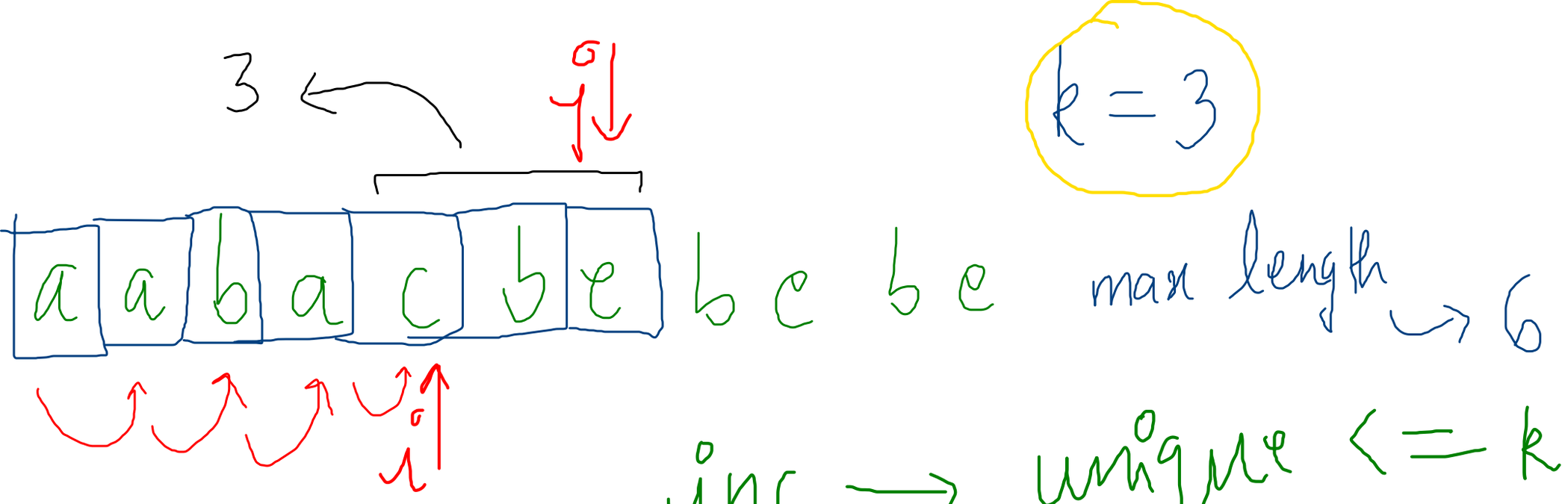
j → ending

↳ 5

max window → ~~1~~ 6



↳ correct mindset



i -> starting point of window, Used for decreasing the window size

j -> ending point of window, Used for increasing the window size

Length of current window - (j - i + 1)
Out of all length we find the maximum

5, 2, 6, 7, 4

42 − 8

34

a * b − c * d

7 * 6

2 * 4

1, 2, 3, -2, 1, 1, 1, 4, 2, -3

$k = 3$

1, 3, 6, 3, 4, 5, 6, 10, 12, 9

$(3-k)$

3

1 → 1

3 → 2

0 → 1

1, 2, 3, -3, 1, 1, 1, 4, 2

prefix[i]

i+1 ---- j

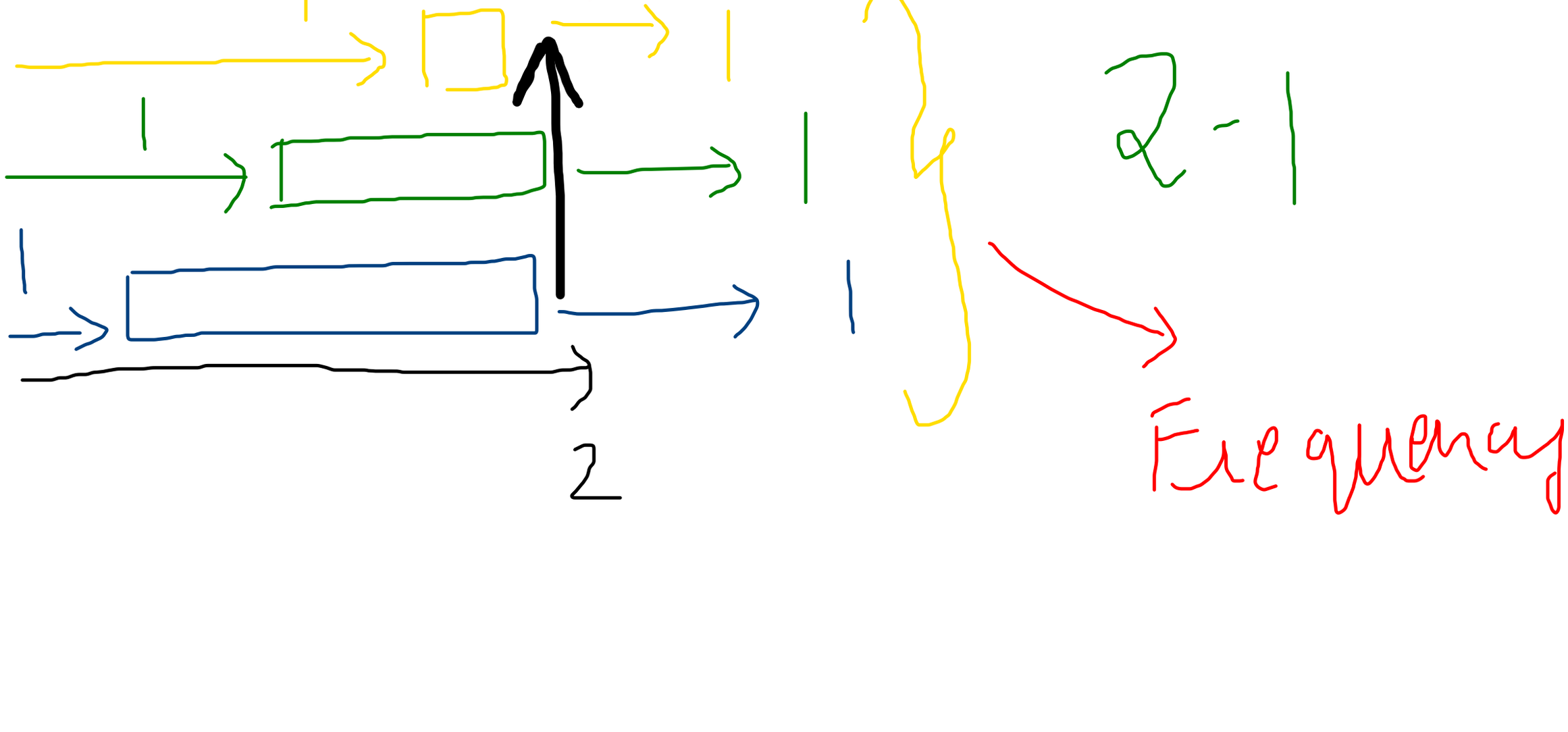
prefix[j]

p[i] - p[j]

1, 0, 0, 1, 2, -2, 1

k = 1

1, 1, 1, 1, 2, 4, 2, 3



8, 1, 2, 2, 3

frequency

0	1	2	1	0	0	0	0	1
0	1	2	3	4	5	6	7	8

1, 2, 2, 3, 8

0	1	1	3	4

0	1	3	4	4	4	4	4	5

prefix

4, 0, 1, 1, 3

1, 1, 1, 3, 4 → 0, 0, 0, 3, 4

O(N)

0	1	2	3	4
0	3	0	1	1

0	3	3	4	5

1 → p[0]

3 → p[2]

4 → p[3]

value → p[value − 1]