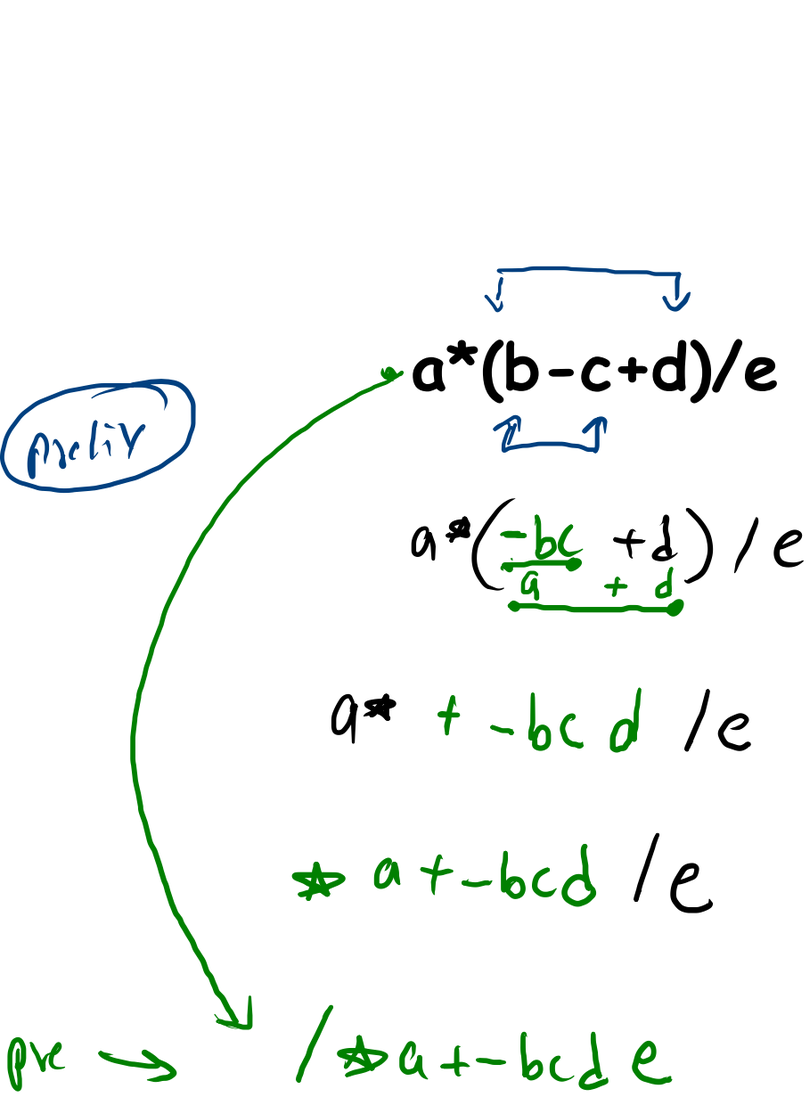
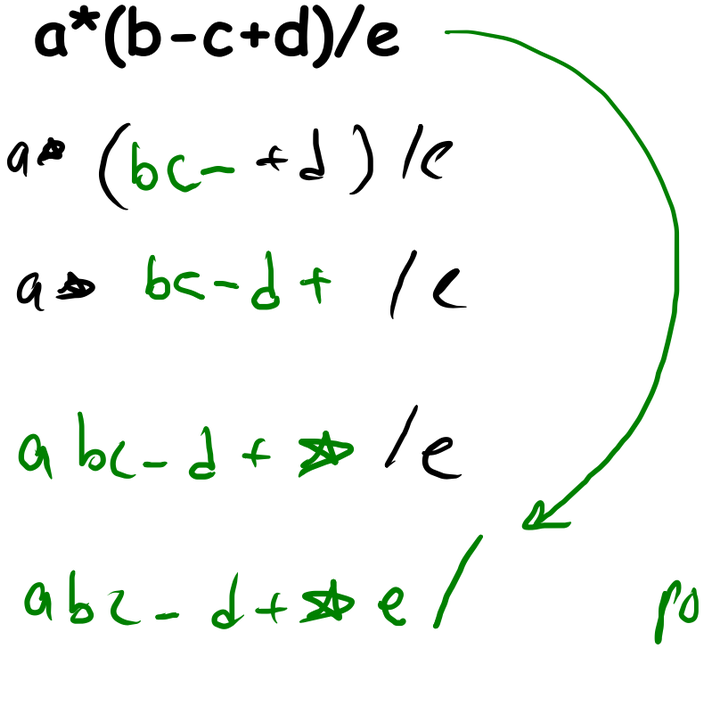
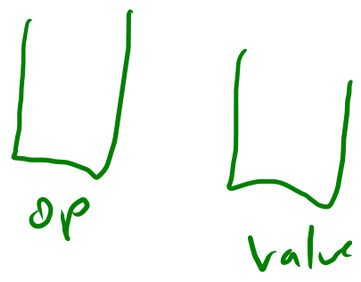
infix → a+b

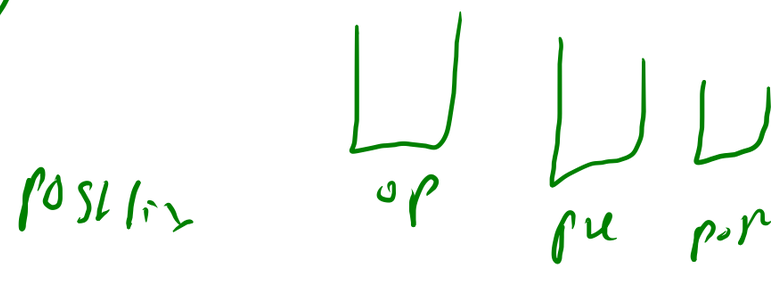
prefix → +ab

postfix → ab+









v1 op v2

a + b

+	a	b	op v1 v2
a	b	+	v1 v2 op

a*(b-c+d)/e

a* (-bc +d) /e

+ -bc d

peek = '(' && peek >= +

- >= +

* >= /

v2 e

v1 *a+-bcd

/*a+-bcde

Strings

Prefix

op /

Operator Character

v2 = e

v1 = abc-d+*

abc-d+*e /

postfix String

→ solve

post → 264*8/+3-

2 24 8/ +3 -

2 3 + 3 -

5 3 -

2

inf. 2+4

post 2 4 $\boxed{+}$

a b

post

prc

→ infix

6*4

24/8 → 3

x = a+b

$\boxed{(\ )}$ $\boxed{+-}$

α α

a+b * c

a+ bc*

abc* +

(a+b)* c

ab+ * c

ab+ c *

(a+b) * (c-d)

ab+ * cd-

ab+ cd- *

a+b * c-d

→ Solve

Postf → 264*8/+3-

2 24 8/ +3-

2 3 +3-

5 3 -

2

prefix ab+ → +ab

264*8/+3-

2 *64 8/+3-

2 /*648 +3-

+ 2 /*648 3-

- 2 /*6483

Infix a+b → ab+

264*8/+3-

2 6*4 8/+3-

2 6*4/8 +3-

2 + 6*4/8 3-

2 + 6*4/8 - 3

Evaluation

$$((p \to r) \Leftrightarrow (q \to p))$$

$$\to (p \to r)\ (q \to p)$$

$$\Leftrightarrow p \to r)\ (q \to p)$$

$$\Leftrightarrow \neg p \vee r)\ (q \to p) \quad \Leftarrow \quad (p \to r) \Leftrightarrow (q \to p)$$

$$(q \to p)$$

$$\uparrow$$

$$+ q\ p$$

→ ab * *

6*4
24/8

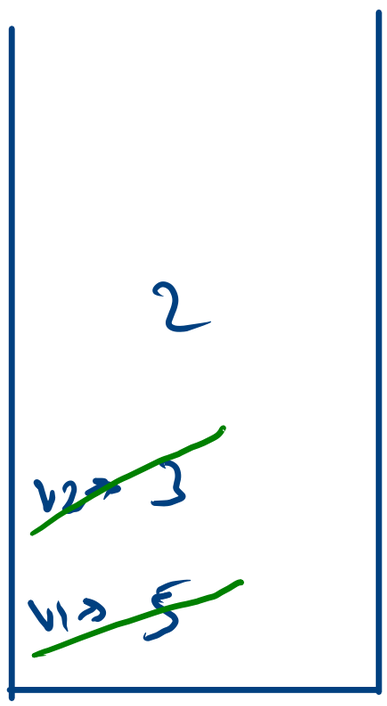
postfix → 264*8/+3-

ab+2

(a+b)

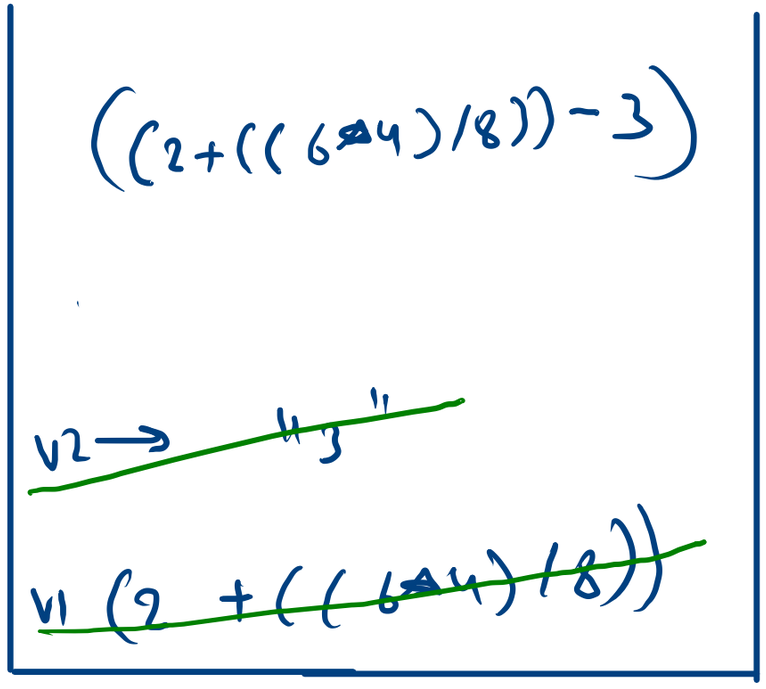
```
stack prefix
stack infix
stack value

iterate 0 to n-1
    ch a b c d
        +++

    + - / *
        solve

print
```

ab



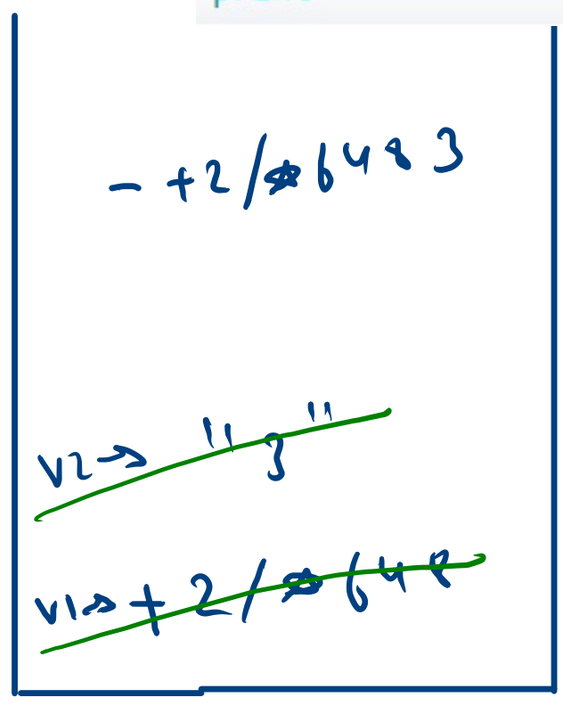
value
Integer

((2+((6*4)/8))-3)



infix
string

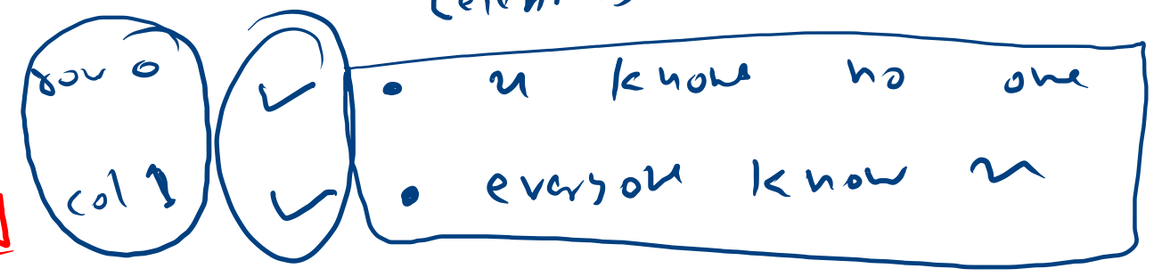
-+2/*6483



postfix
string

$O(n^2) \rightarrow O(n)$

celebrity → n



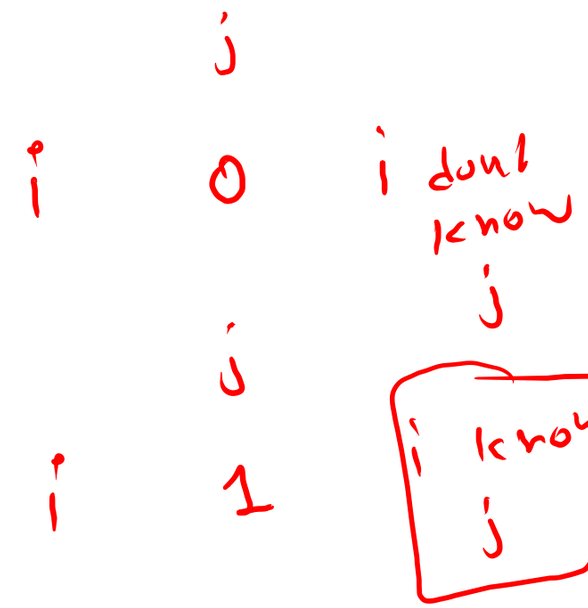
4

0000
1011
1101
1110

	a	b	c	d
a	0	0	0	0
b	1	0	1	1
c	1	1	0	1
d	1	1	1	0

a

xor 0



b → a ✓
b → c
b → d

a ✗ b a ✗ c a ✗ d

$n-1+2n$

$\boxed{O(n)}$

$n\ (2N)$

$O(n^2)$

$\downarrow$

$O(n)$

	a	b	c	d
a	0	0	0	0
b	10	0	1	1
c	1	1	0	1
d	1	1	1	0

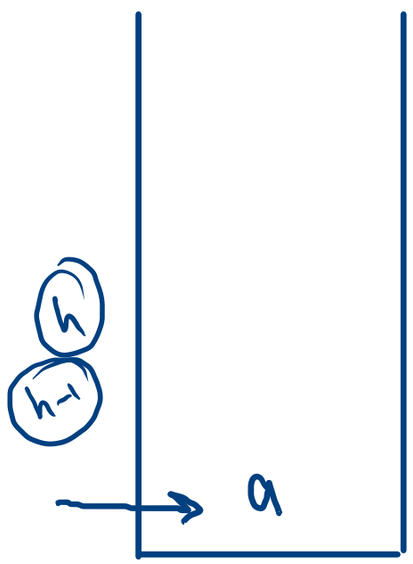
PC = a

$\overset{a}{[x]}\overset{d}{[y]} = 0 \rightarrow$ ✓ x ✗ y

$\underset{b}{[x]}\underset{d}{[y]} = 1 \rightarrow y$ x

✓ a

✗ none



✗ ✓

$[c]\ [d] = 1$

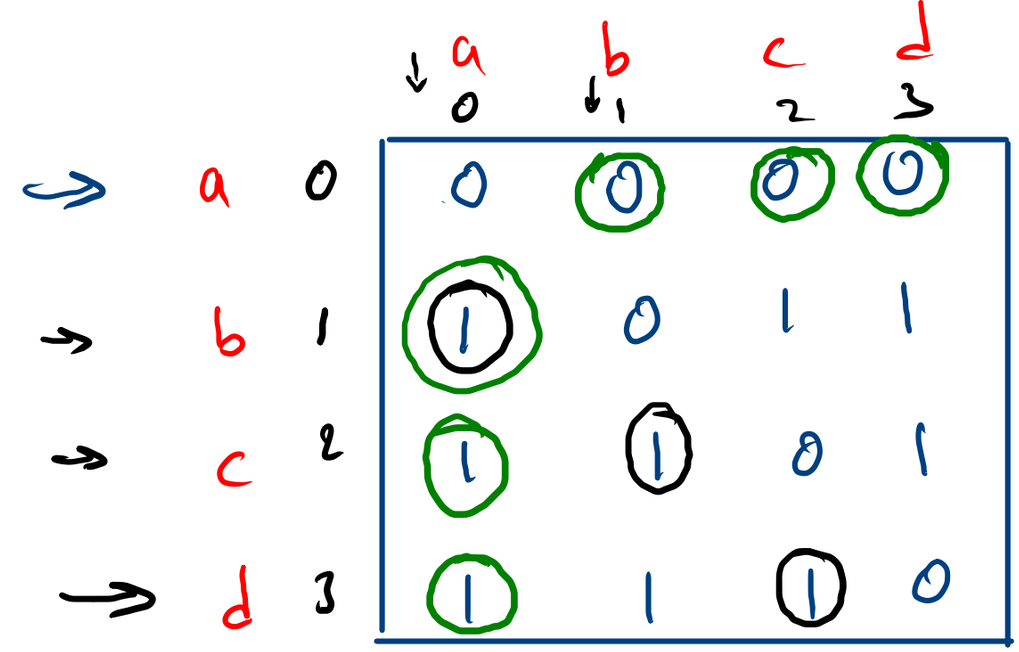
$[b]\ [d] = 1$

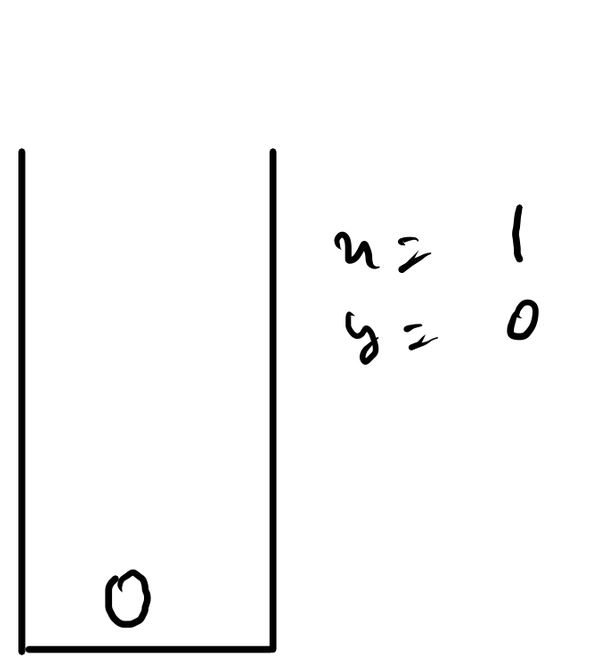
$[a]\ [d] =$

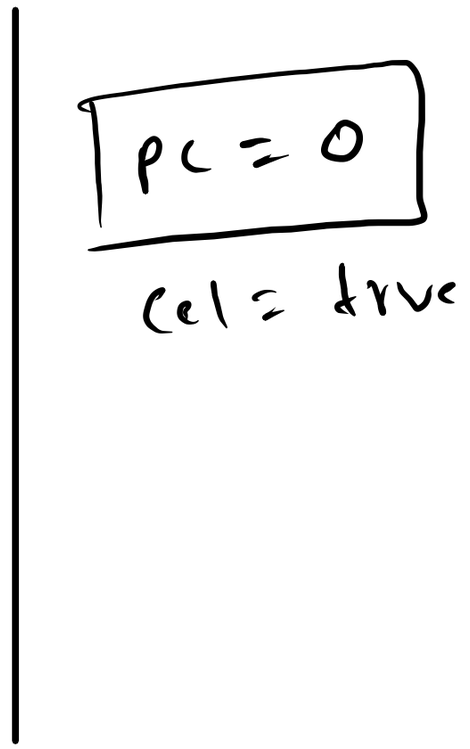
y

x 1

```java
for(int i=0;i<arr.length;i++){
    st.push(i);
}

while(st.size() >=2){
    int x = st.pop();
    int y = st.pop();
    if(arr[x][y] == 0){
        st.push(x);
    }else{
        st.push(y);
    }
}

int pc = st.pop();

boolean cel = true;
for(int i=0;i<arr.length;i++){
    if(pc == i)continue;
    if(arr[pc][i] == 1 || arr[i][pc] == 0){
        cel = false;
        break;
    }
}

if(cel){
    System.out.println(pc);
}else{
    System.out.println("none");
}
```

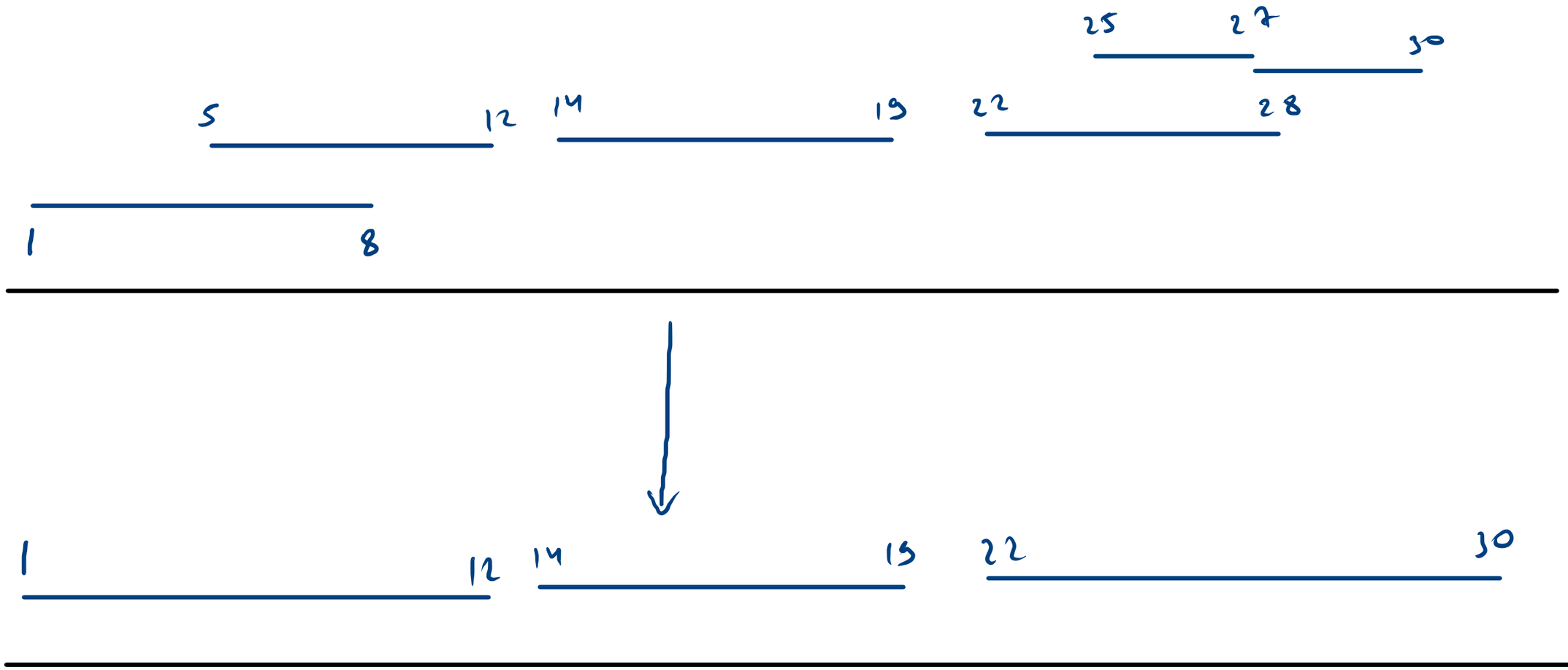






1 8
5 12
14 19
22 28
25 27
27 30

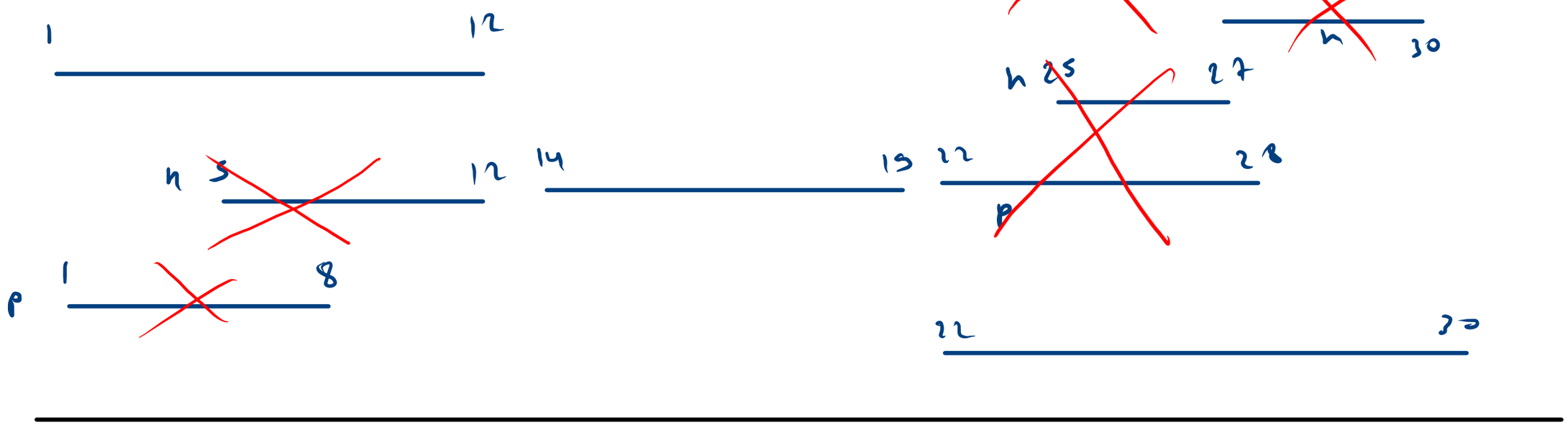
Then the
1 12
14 19
22 30



Pair can

- 1 8
- 5 12
- 14 19
- 22 28
- 25 27
- 27 30

Then the



Check

Correct

1 12
14 19
22 30

overlap → h.start ≤ p.end

27 ≤ 28

- merged.start = p.start
- merged.end = max(h.end, p.end)

1 8
5 12
14 19
22 28
25 27
27 30

→

1	8
22	28
27	30
5	12

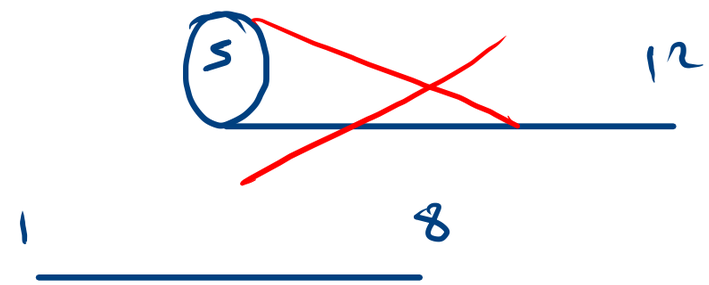
$5 \le 30$

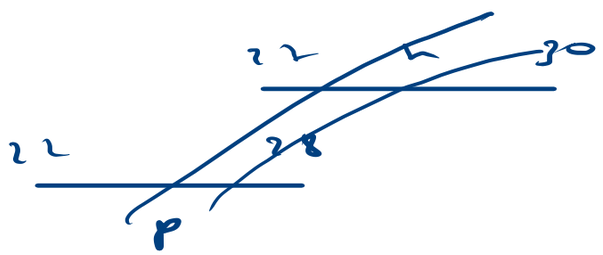
{
int start
int end
}

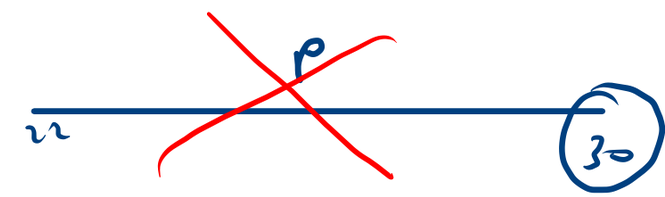
- 1 8
- 5 12
- 14 19
- 22 28
- 25 27
- 27 30

Then the

1 12
14 19
22 30







- 1 8
- 5 12
- 14 19 n
- 22 28
- 25 27
- 27 30

Then the
1 12
14 19
22 30

p → st. pack

25 27

p →

{27, 30}

{14 19}

{1, 12}

1	12
14	19
27	30

output

Check

overlap → n.start ≤ p.end

27 ≤ 28

Correct

merged.start = p.start

- merged.end = max(n.end, p.end)

~~{27, 30}~~

~~{14 19}~~

~~{1, 12}~~

{ 1, 12}

{ 14, 19}

{ 27, 30}