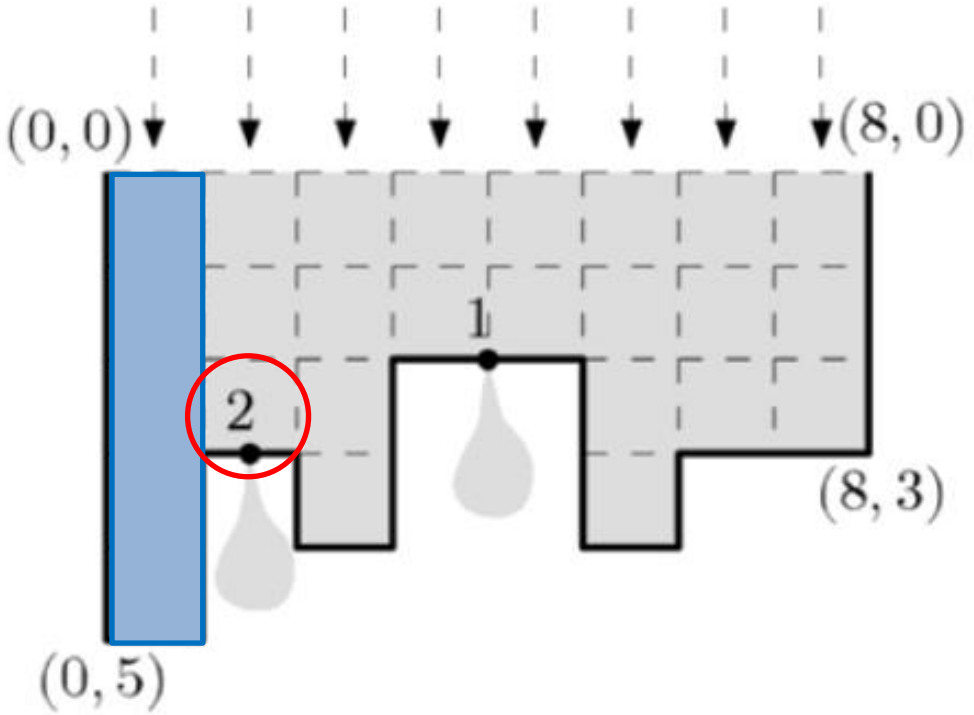
BOJ 8982 풀이

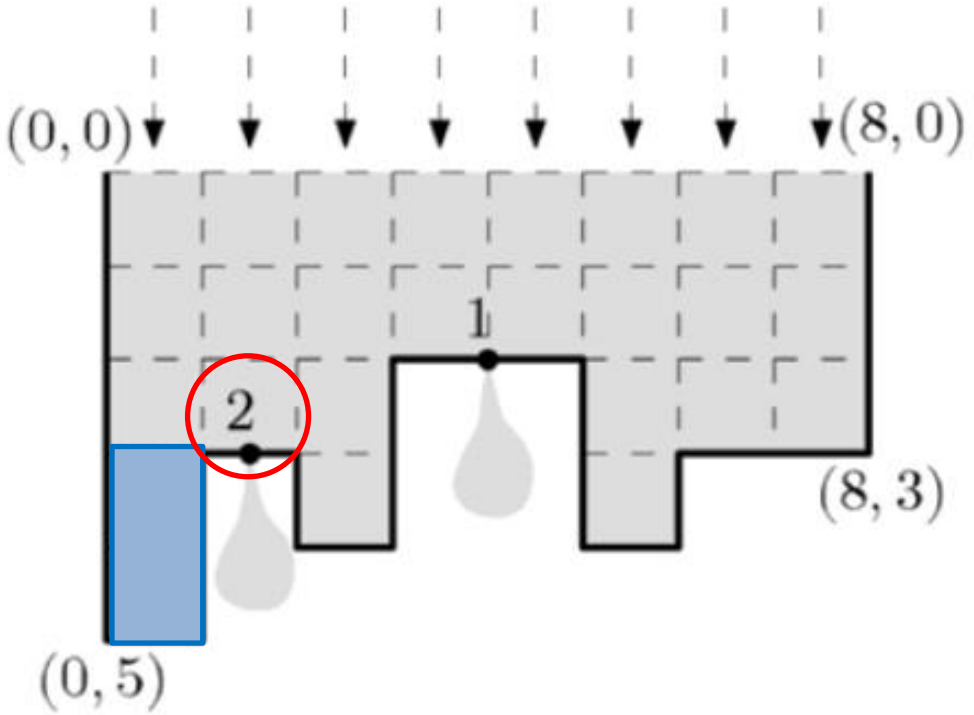
아이디어

1. 왼쪽부터 블록으로 넓이를 구한다.



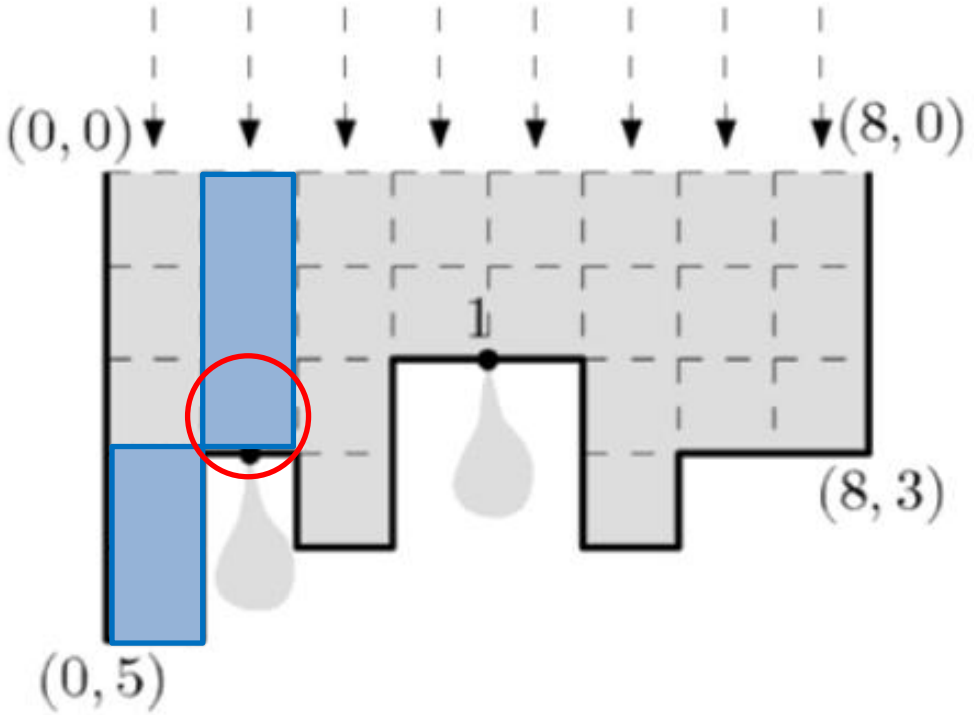
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



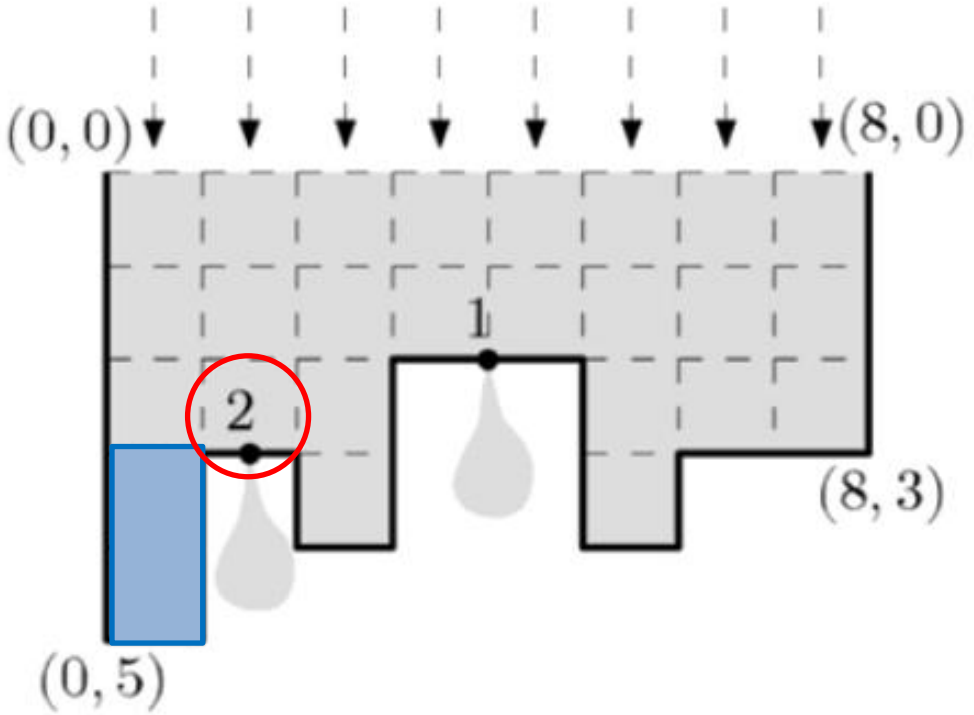
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



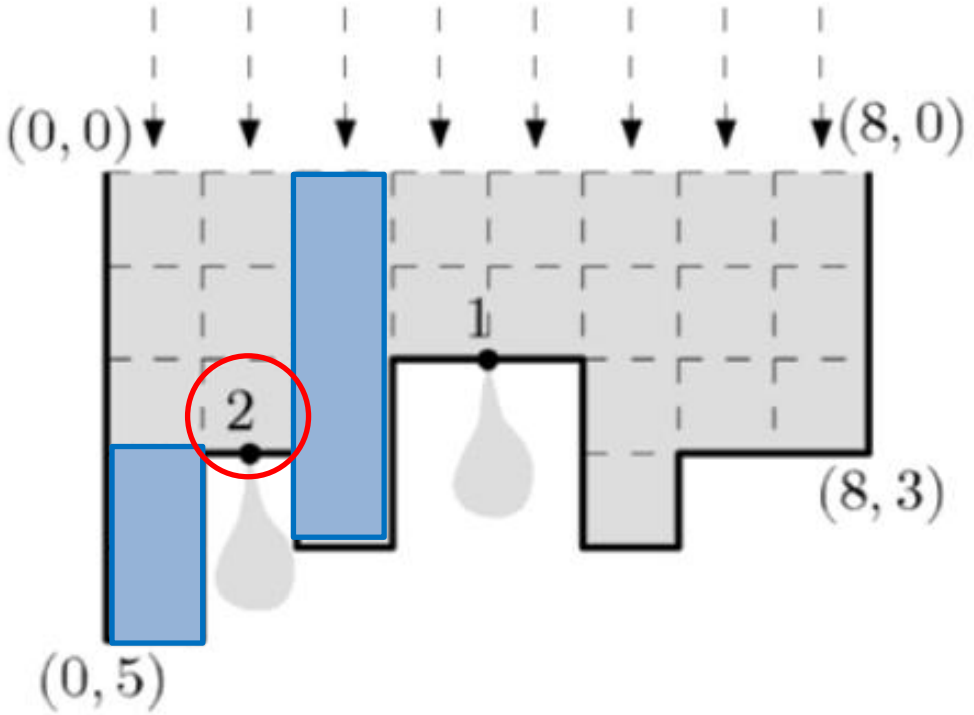
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



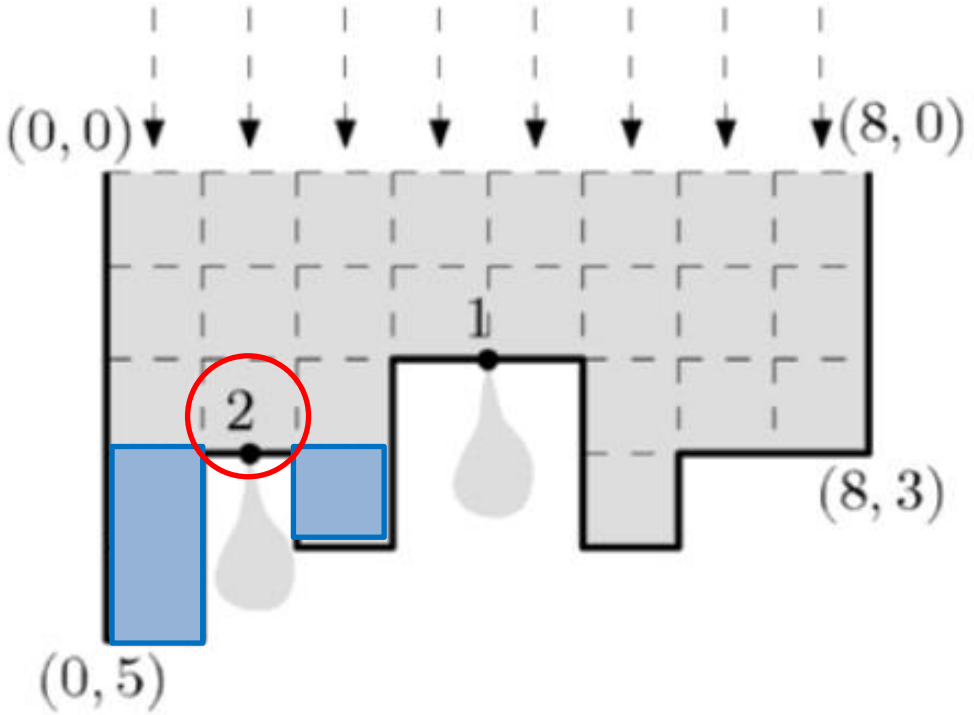
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



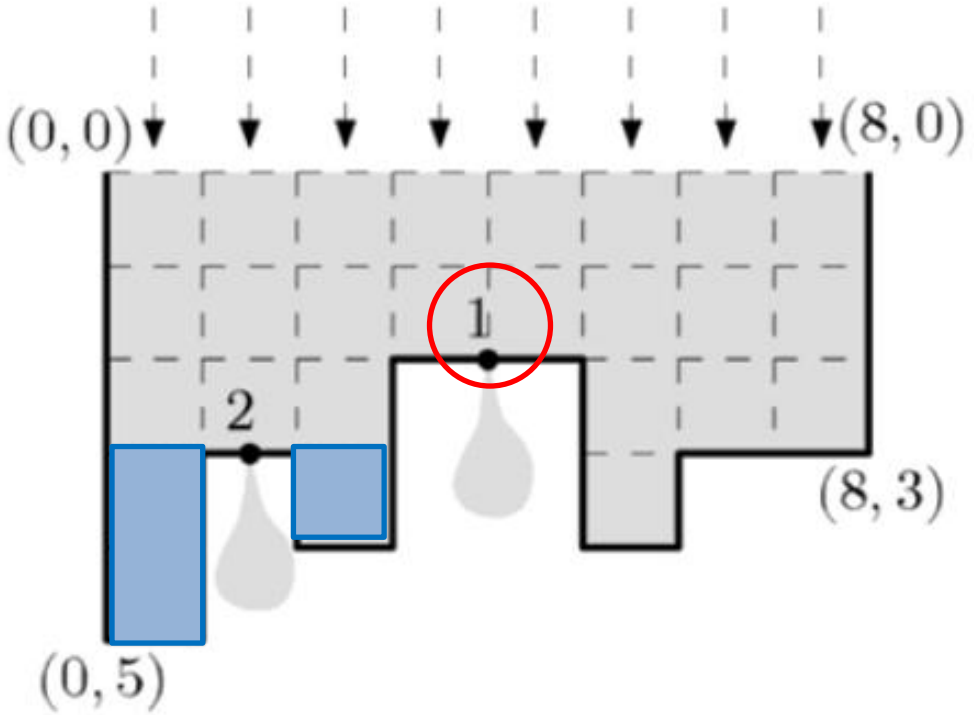
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



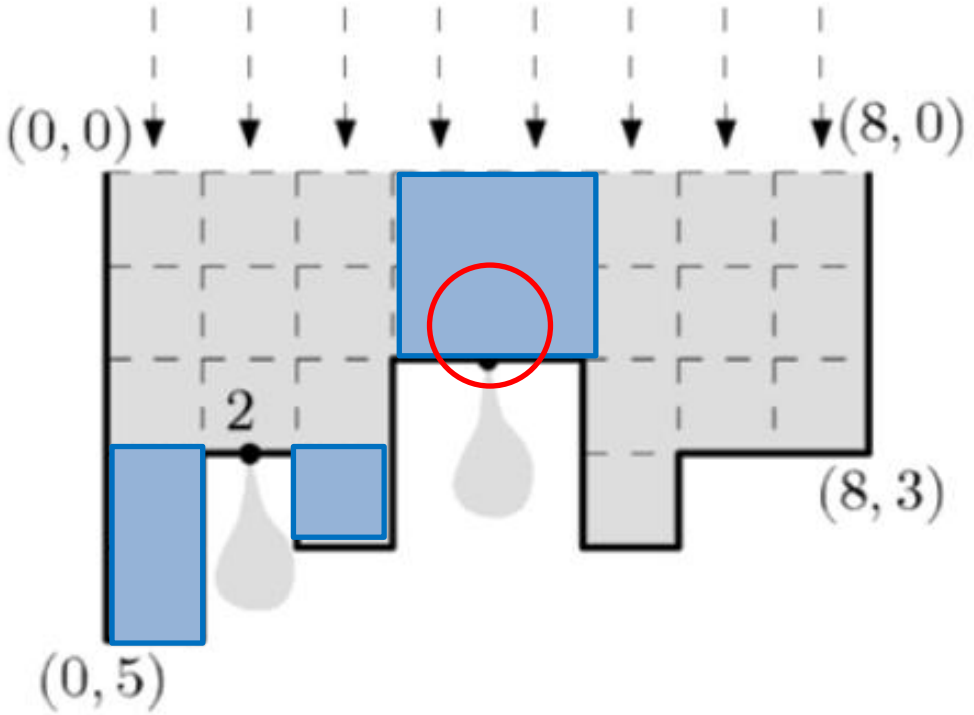
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



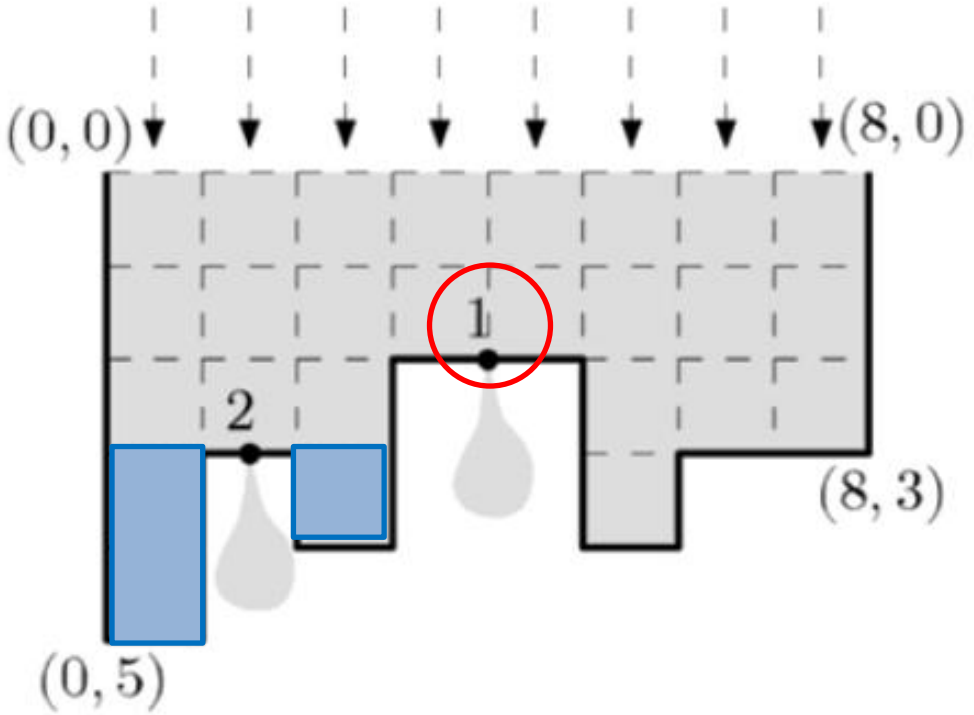
아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



아이디어

2. 가까운 구멍의 위치를 파악해 물을 뺀다.



블록(꼭짓점), 구멍

```
pos=new int[N-2][2];

for(int i=0;i<N-2;i++) { //첫,끝 꼭짓점은 저장 할 필요 없어서
    st=new StringTokenizer(br.readLine());
    pos[i][0]=Integer.parseInt(st.nextToken());
    pos[i][1]=Integer.parseInt(st.nextToken());
}
br.readLine(); // 끝 꼭짓점 필요없어서 버림
```

POS에 꼭짓점 저장

→POS[X][Y]

두개씩 꺼내어, 한 블록의 물 체크

물을 빼주는 drain()

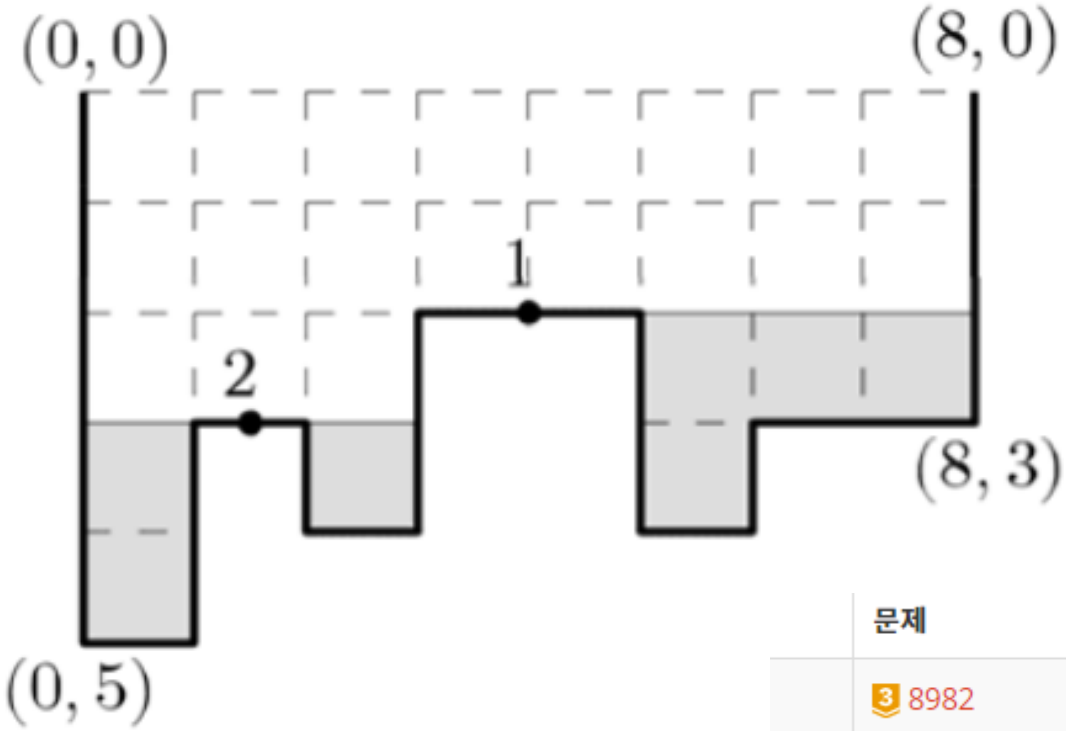
```
public static void drain(int x1, int y1, int x2, int y2) { //y2는 사실상 필요없고 저장할 필요x
    int L=(x2-x1)*y1; //블럭 한 줄의 면적.

    int mid=(x1+x2)/2;
    float holeY=-1;
    float min=40000;
    for(int i=0;i<holeCnt;i++) {//가까운 블럭만 물빠짐에 정확한 영향을 주므로,찾아내서 활용한다.
        if(min > Math.abs(mid-hole[i][0])) { //mid와 가장 절댓값 차가 작은 구멍:  가까이에 있는 구멍
            min=Math.abs(mid-hole[i][0]);
            holeY=hole[i][1];
```

1. POS의 꼭짓점을 2개씩 꺼내서
블록마다 물의 양 계산.

2. 가장 가까운 구멍: X축 기준
|블록 중간-구멍| 가장 작을 때
➔그때의 구멍의 Y를 기준으로 물을 뺀다.

아이디어

이 방식을 통한 결과



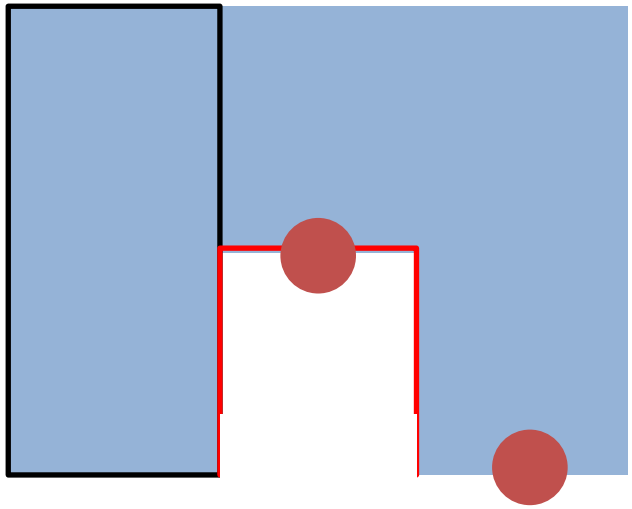
	문제	결과
	3 8982	틀렸습니다

아이디어

생각해 보니까. . .

재도 가까운 구멍

➔ 물을 다 빼버림. 결과가 0나옴.



(구멍 / 현재 블록) 사이에 벽이 있는가?
➔ 구멍높이=벽 높이

블록

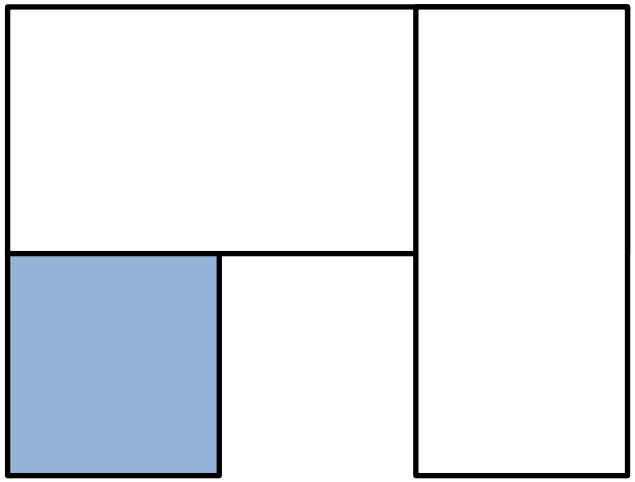
```
for(int j=0;j<N-2;j++) {
    if(mid<pos[j][0] && pos[j][0]<hole[i][0]) { //현재블럭과 구멍 사이에 튀어나온 벽이 있는지 체크.
        if(pos[j][1]<holeY) holeY=pos[j][1]; //벽이 튀어나와있다면
    }
}
```

현재 (가장 가까운 구멍/블록) 사이의 벽?

➔ 둘 사이의 POS 중 Y값이 작은 게 있는가

아이디어

➔ 수정하여 물을 뺀 결과



	문제	결과
	3 8982	틀렸습니다