



벽

지안지아는 똑같은 크기의 벽돌을 쌓아서 벽을 만들고 있다. 이 벽은 n 열의 벽돌로 되어 있는데, 각 열은 왼쪽부터 오른쪽으로 차례대로 0부터 $n - 1$ 까지 번호가 매겨져 있다. 각 열의 높이는 서로 다를 수 있다. 열의 높이는 이 열에 쌓인 벽돌의 수이다.

지안지아는 다음과 같이 벽을 만든다. 처음에는 어느 열에도 벽돌이 없다. 다음, 지안지아는 k 단계에 걸쳐 벽돌을 **더하거나** 또는 **빼거나** 한다. k 단계가 다 끝나면 벽을 다 쌓은 것이다. 매 단계마다 지안지아는 연속된 벽돌 열의 범위와 높이 h 를 받고, 다음과 같은 절차에 따라 정해진 일을 한다.

- 벽돌을 **더하는** 단계에서는, 지안지아는 주어진 범위에 해당하는 열들 중 h 장 미만의 벽돌이 쌓인 열들에 벽돌을 더해서 정확히 벽돌 h 장이 쌓이게 한다. h 장 이상 벽돌이 있는 열에는 아무 일도 하지 않는다.
- 벽돌을 **빼는** 단계에는, 지안지아는 주어진 범위에 해당하는 열들 중 h 장 초과 벽돌이 쌓인 열들에서 벽돌을 빼서 정확히 벽돌 h 장이 쌓이게 한다. h 장 이하 벽돌이 있는 열에는 아무 일도 하지 않는다.

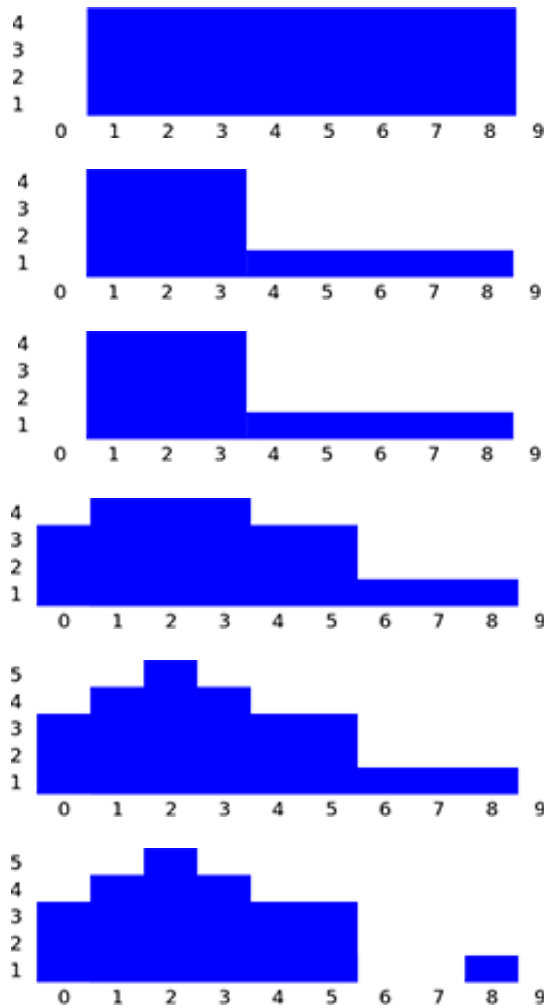
당신이 할 일은 벽의 최종 모양을 결정하는 것이다.

예제

10열의 벽돌이 있고 6단계를 거쳐 벽을 만든다고 가정하자. 아래 표의 모든 범위는 양 끝을 포함한다. 각 단계가 끝났을 때 벽의 모양은 아래 그림과 같다.

| 단계 | 하는 일 | 범위 | 높이 |
|----|------|-----------|----|
| 0 | 더하기 | 1열부터 8열까지 | 4 |
| 1 | 빼기 | 4열부터 9열까지 | 1 |
| 2 | 빼기 | 3열부터 6열까지 | 5 |
| 3 | 더하기 | 0열부터 5열까지 | 3 |
| 4 | 더하기 | 2열 | 5 |
| 5 | 빼기 | 6열부터 7열까지 | 0 |

처음에 모든 열에는 벽돌이 없기 때문에, 단계 0이 끝나면 1열부터 8열까지는 모두 4장의 벽돌이 있다. 0열과 9열은 비어 있다. 단계 1에서는, 4열부터 8열까지는 벽돌이 빠져서 모든 열에 각각 벽돌이 1장이 있고, 9열은 계속 비어 있다. 주어진 범위 밖인 0열부터 3열은 아무 변화가 없다. 단계 2는 아무 변화가 없는데, 3열부터 6열까지 5장을 초과하여 벽돌이 있는 열이 없기 때문이다. 단계 3이 끝나면 0, 4, 5열의 벽돌은 3장으로 늘어난다. 단계 4가 끝나면 2열에는 벽돌이 5장 있다. 단계 5는 6열과 7열의 모든 벽돌을 없앤다.



문제

각 k 단계에서 하는 일이 주어졌을 때, 모든 단계가 끝난 다음 각 열에 남아 있는 벽돌의 수를 계산 하시오. 이를 위해서 함수 `buildWall`을 구현해야 한다.

- `buildWall(n, k, op, left, right, height, finalHeight)`
 - n : 벽에 있는 열의 수.
 - k : 단계의 수 .
 - `op`: 크기 k 인 배열; $0 \leq i \leq k - 1$ 일 때 `op[i]`는 단계 i 에서 하는 일을 나타낸다. 이 값이 1이면 더하는 단계이고 2이면 빼는 단계이다.
 - `left`와 `right`: 크기 k 인 배열; $0 \leq i \leq k - 1$ 일 때 단계 i 에 해당하는 열의 범위는 `left[i]` 열에서 시작하고 `right[i]` 열에서 끝난다. (양 끝점 `left[i]`와 `right[i]`가 포함된다.) 항상 `left[i] ≤ right[i]`이다.
 - `height`: 크기 k 인 배열; $0 \leq i \leq k - 1$ 일 때 `height[i]`는 단계 i 에서 주어지는 높이 파라미터이다.
 - `finalHeight`: 크기 n 인 배열; $0 \leq i \leq n - 1$ 일 때 모든 단계를 수행한 다음 열 i 에 벽돌이 몇 장 있는지 `finalHeight[i]`에 저장하여 리턴해야 한다.

부분문제

모든 부분문제에서 모든 단계의 높이 파라미터는 **100,000** 이하인 음이 아닌 정수이다.

| 부분문제 | 점수 | n | k | 참고 |
|------|----|---------------------------|-------------------------|----------------------------|
| 1 | 8 | $1 \leq n \leq 10,000$ | $1 \leq k \leq 5,000$ | 추가 제한 조건 없음 |
| 2 | 24 | $1 \leq n \leq 100,000$ | $1 \leq k \leq 500,000$ | 모든 더하기 단계는 모든 빼기 단계보다 앞에 옴 |
| 3 | 29 | $1 \leq n \leq 100,000$ | $1 \leq k \leq 500,000$ | 추가 제한 조건 없음 |
| 4 | 39 | $1 \leq n \leq 2,000,000$ | $1 \leq k \leq 500,000$ | 추가 제한 조건 없음 |

구현 내용

정확히 하나의 파일 `wall.c`, `wall.cpp` 또는 `wall.pas`를 제출해야 한다. 이 파일은 다음과 같은 선언을 사용하여 위에서 설명한 서브프로그램을 구현한다. C/C++ 프로그램의 경우 헤더 파일 `wall.h` 파일을 `#include` 해야 한다.

C/C++ 프로그램

```
void buildWall(int n, int k, int op[], int left[], int right[], int height[], int finalHeight[]);
```

Pascal 프로그램

```
procedure buildWall(n, k : longint; op, left, right, height : array of longint; var finalHeight : array of longint);
```

Sample grader

Sample grader는 다음 양식에 따라 입력을 읽어들이는다.

- line 1: n, k .
- line $2 + i$ ($0 \leq i \leq k - 1$): $op[i], left[i], right[i], height[i]$.