Wall (Muro)

Jian-Jia está construyendo un muro con ladrillos todos del mismo tamaño. El muro tiene n columnas de ladrillos numeradas de 0 a $n-1$ de izquierda a derecha. Las columnas pueden tener distintas alturas dependiendo del número de ladrillos en ellas.

Jian-Jia construye el muro de la siguiente forma. Inicialmente no hay ladrillos en ninguna columna. Después, Jian-Jia realiza k fases de *agregar* o *quitar* ladrillos. El proceso de construcción termina cuando las k fases han sido completadas. En cada fase Jian-Jia recibe un rango de columnas contiguas y una altura h con los que realiza el siguiente procedimiento:

- En las fases de *agregar*, Jian-Jia agrega ladrillos a aquellas columnas del rango que tienen menos de h ladrillos de modo que queden con exactamente h ladrillos. Él no agrega ningún ladrillo a las columnas que tienen h o más ladrillos.
- En las fases de *quitar*, Jian-Jia quita ladrillos de aquellas columnas del rango que tienen más de h ladrillos de modo que queden con exactamente h ladrillos. Él no quita ningún ladrillo de las columnas que tienen h o menos ladrillos.

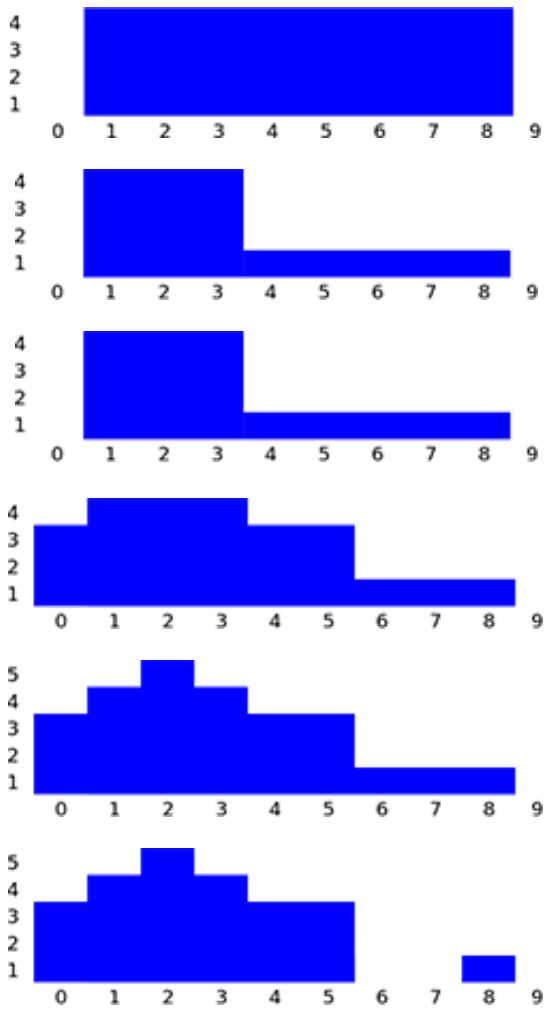
Tu tarea es determinar la forma final del muro.

Ejemplo

Supón que hay 10 columnas de ladrillos y 6 fases de construcción. Todos los rangos de la siguiente tabla son intervalos cerrados. Los diagramas de cada fase se muestran debajo.

fase	tipo	rango	altura
0	agregar	columnas 1 a 8	4
1	quitar	columnas 4 a 9	1
2	quitar	columnas 3 a 6	5
3	agregar	columnas 0 a 5	3
4	agregar	columna 2	5
5	quitar	columnas 6 a 7	0

Al inicio todas las columnas están vacías, después de la fase 0 cada una de las columnas de la 1 a la 8 tiene 4 ladrillos, las columnas 0 y 9 permanecen vacías. En la fase 1, se quitan ladrillos de las columnas 4 a la 8 hasta que cada una de ellas tenga 1 ladrillo, la columna 9 permanece vacía y las columnas de la 0 a la 3 quedan sin cambios. La fase 2 no cambia el muro dado que las columnas de la 3 a la 6 no tienen más de 5 ladrillos. Después de la fase 3 el número de ladrillos en las columnas 0, 4 y 5 se incrementa a 3. Al finalizar la fase 4 hay 5 ladrillos en la columna 2. La fase 5 quita todos los ladrillos de las columnas 6 y 7.



Problema

Escribe un programa que dada la descripción de las k fases, calcule el número de ladrillos en cada columna al finalizar todas las fases. Necesitarás implementar la función `buildWall`.

- `buildWall(n, k, op, left, right, height, finalHeight)`
  - `n`: el número de columnas en el muro.
  - `k`: el número de fases.
  - `op`: un arreglo de longitud k donde `op[i]` es el tipo de la fase i: 1 para una fase de *agregar* y 2 para una fase de *quitar*, para $0 \leq i \leq k-1$.
  - `left` y `right`: arreglos de longitud k representando que el rango de columnas en la fase i inicia en la columna `left[i]` y termina en la columna `right[i]` (incluyendo los extremos `left[i]` y `right[i]`), para $0 \leq i \leq k-1$. Siempre se cumple `left[i]` $\leq$ `right[i]`.
  - `height`: un arreglo de longitud k donde `height[i]` es la altura para la fase i, para $0 \leq i \leq k-1$.
  - `finalHeight`: un arreglo de longitud n; deberás representar tus resultados colocando el número final de ladrillos de la columna i `finalHeight[i]`, para $0 \leq i \leq n-1$.

Subproblemas

Para todos los subproblemas las alturas de las fases serán enteros no negativos menores o iguales a $100,000$.

subproblema	puntos	n	k	nota
1	8	$1 \leq n \leq 10,000$	$1 \leq k \leq 5,000$	sin límites adicionales
2	24	$1 \leq n \leq 100,000$	$1 \leq k \leq 500,000$	todas las fases de *agregar* se encuentran antes de las fases de *quitar*
3	29	$1 \leq n \leq 100,000$	$1 \leq k \leq 500,000$	Sin límites adicionales
4	39	$1 \leq n \leq 2,000,000$	$1 \leq k \leq 500,000$	sin límites adicionales

Detalles de implementación

Debes enviar sólo un archivo, llamado `wall.c`, `wall.cpp` o `wall.pas`. Este archivo debe implementar la función descrita arriba usando el siguiente prototipo. Además necesitarás incluir el archivo `wall.h` para el caso de programas en C/C++.

Programa en C/C++

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

Programa en Pascal

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

Evaluador de prueba

El evaluador de prueba lee la entrada en el siguiente formato:

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