



## Muur

Jian-Jia bouwt een muur door bakstenen van dezelfde grootte op elkaar te stapelen. De muur bestaat uit  $n$  kolommen van bakstenen, die we nummeren van  $0$  tot  $n - 1$  van links naar rechts. De kolommen kunnen verschillende hoogtes hebben. De hoogte van een kolom is gelijk aan het aantal bakstenen erin.

Jian-Jia bouwt de muur als volgt. In het begin bevat geen enkele kolom bakstenen. Daarna gaat Jian-Jia door  $k$  fases van *toevoegen* of *verwijderen* van bakstenen. Het bouwproces is gedaan wanneer alle  $k$  fases zijn afgelopen. In elke fase krijgt Jian-Jia een interval van opeenvolgende kolommen en een hoogte  $h$ , en volgt hij de volgende procedure:

- In een fase *toevoegen*, zal Jian-Jia bakstenen toevoegen in die kolommen binnen het gegeven interval die minder dan  $h$  bakstenen hebben, zodanig dat die exact  $h$  bakstenen bevatten. Hij doet niets in kolommen die  $h$  of meer bakstenen bevatten.
- In een fase *verwijderen*, zal Jian-Jia bakstenen verwijderen van die kolommen binnen het gegeven interval die meer dan  $h$  bakstenen bevatten, zodanig dat die exact  $h$  bakstenen bevatten. Hij doet niets in kolommen die  $h$  of minder bakstenen bevatten.

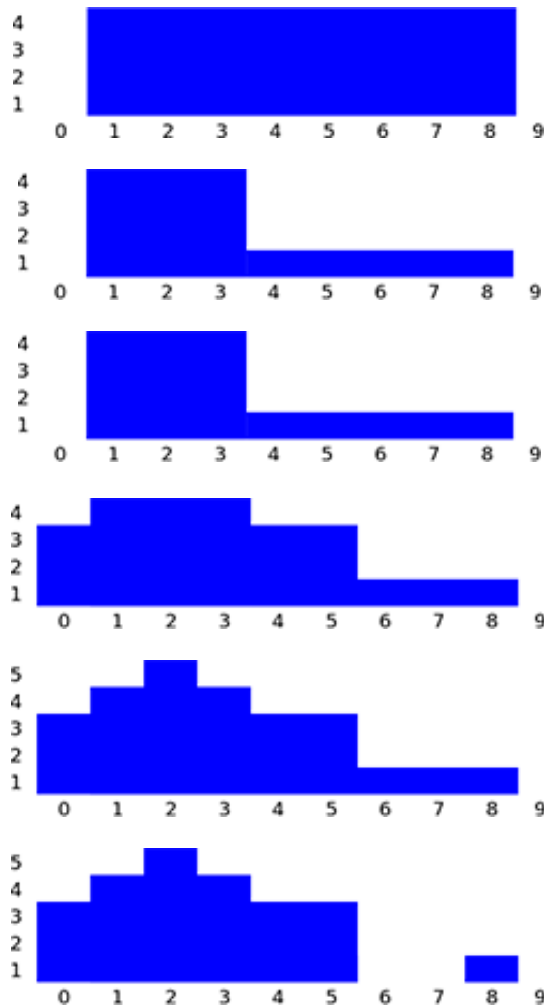
Jouw taak is om de uiteindelijke vorm van de muur te bepalen.

## Voorbeeld

We nemen aan dat er 10 bakstenen kolommen zijn en 6 bouwfasen. Alle intervallen in de volgende tabel zijn inclusief eindpunten. Een voorstelling van de muur na elke fase wordt daaronder weergegeven.

fase	type	interval	hoogte
0	toevoegen	kolommen 1 t/m 8	4
1	verwijderen	kolommen 4 t/m 9	1
2	verwijderen	kolommen 3 t/m 6	5
3	toevoegen	kolommen 0 t/m 5	3
4	toevoegen	kolom 2	5
5	verwijderen	kolommen 6 t/m 7	0

Gezien alle kolommen in het begin leeg zijn, hebben kolommen 1 t/m 8 na fase 0 exact 4 bakstenen. Kolommen 0 en 9 blijven leeg. In fase 1 worden bakstenen weggenomen van kolommen 4 t/m 8 totdat die allemaal uit 1 baksteen bestaan, en kolom 9 blijft leeg. Kolommen 0 t/m 3, die buiten het gegeven interval liggen, blijven ongewijzigd. Fase 2 verandert niets, gezien kolommen 3 t/m 6 niet meer dan 5 bakstenen bevatten. Na fase 3 is het aantal bakstenen in kolommen 0, 4 en 5 verhoogd tot 3. Er zijn 5 bakstenen in kolom 2 na fase 4. Fase 5 verwijdert alle bakstenen uit kolommen 6 en 7.



## Taak

Gegeven de beschrijving van de  $k$  fases, moet je berekenen hoeveel bakstenen iedere kolom bevat nadat alle fases afgelopen zijn. Je moet de functie `buildWall` implementeren.

- `buildWall(n, k, op, left, right, height, finalHeight)`
  - $n$ : het aantal kolommen in de muur.
  - $k$ : het aantal fases.
  - `op`: array van lengte  $k$ ; `op[i]` is het type van fase  $i$ : 1 voor een fase van toevoegen en 2 voor een fase van verwijderen, waarbij  $0 \leq i \leq k - 1$ .
  - `left` en `right`: arrays van lengte  $k$ ; het interval van kolommen in fase  $i$  begint met kolom `left[i]` en eindigt met kolom `right[i]` (inclusief beide eindpunten `left[i]` en `right[i]`), waarbij  $0 \leq i \leq k - 1$ . Altijd geldt dat `left[i] ≤ right[i]`.
  - `height`: array van lengte  $k$ ; `height[i]` is de hoogte-parameter van fase  $i$ , met  $0 \leq i \leq k - 1$ .
  - `finalHeight`: array van lengte  $n$ ; je moet jouw resultaat teruggeven door het uiteindelijke aantal bakstenen in kolom  $i$  te schrijven in `finalHeight[i]`, voor  $0 \leq i \leq n - 1$ .

## Subtaken

Voor alle subtaken zijn de hoogteparameters van alle fases niet-negatieve integers kleiner of gelijk aan 100,000.

subtaak	punten	$n$	$k$	opmerking
1	8	$1 \leq n \leq 10,000$	$1 \leq k \leq 5,000$	geen bijkomende beperkingen
2	24	$1 \leq n \leq 100,000$	$1 \leq k \leq 500,000$	alle toevoegingsfases gebeuren voor alle verwijderingsfases
3	29	$1 \leq n \leq 100,000$	$1 \leq k \leq 500,000$	geen bijkomende beperkingen
4	39	$1 \leq n \leq 2,000,000$	$1 \leq k \leq 500,000$	geen bijkomende beperkingen

## Implementatiedetails

Je moet exact één bestand indienen, genaamd `wall.c`, `wall.cpp` of `wall.pas`. Dit bestand implementeert de bovenstaande subroutine volgens de declaraties hieronder. Bij C/C++ implementatie moet dat ook een header-bestand `wall.h` "includeren".

### C/C++ programma

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

### Pascal programma

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

## Voorbeeldgrader

De voorbeeldgrader leest input in het volgende formaat:

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