კედელი

ჯიან–ჯია აშენებს კედელს ერთი ზომის აგურებისაგან. კედელი შედგება აგურთა n რაოდენობის სვეტისაგან, რომლებიც გადანომრილია 0-დან $(n-1)$-მდე მარცხნიდან მარჯვნივ. სვეტებს შეიძლება განსხვავებული სიმაღლეები ჰქონდეთ. სვეტის სიმაღლედ მასში აგურების რაოდენობა ითვლება.

ჯიან-ჯია კედელს შემდეგნაირად აშენებს: თავიდან ყველა სვეტი ცარიელია. შემდეგ ჯიან-ჯია ასრულებს აგურების *დამატების* ან *აღების* k რაოდენობის ფაზას. მშენებლობის პროცესი მთავრდება მაშინ, როცა ყველა k ფაზა შესრულებულია. ყოველ ფაზაზე ჯიან-ჯიას ეძლევა მიმდევრობით განლაგებული სვეტების ნომერთა დიაპაზონი და h სიმაღლე, რის შემდეგაც იგი ასრულებს შემდეგ მოქმედებებს:

- *დამატების* ფაზაში ჯიან-ჯია მოცემულ დიაპაზონში განლაგებული სვეტებიდან ამატებს აგურებს იმ სვეტებში, რომლებშიც მათი რაოდენობა h-ზე ნაკლებია ისე, რომ თითოეულ მათგანში აგურების რაოდენობა ზუსტად h -ის ტოლი გახდეს. იგი ამ დიაპაზონში არ ეხება იმ სვეტებს, რომლებშიც აგურების რაოდენობა h-ის ტოლია ან h-ზე მეტია.
- *აღების* ფაზაში ჯიან-ჯია მოცემულ დიაპაზონში განლაგებული სვეტებიდან იღებს აგურებს იმ სვეტებიდან, რომლებშიც მათი რაოდენობა h-ზე მეტია ისე, რომ თითოეულ მათგანში ზუსტად h რაოდენობის აგური დარჩეს. იგი ამ დიაპაზონში არ ეხება იმ სვეტებს, რომლებშიც აგურების რაოდენობა h-ის ტოლია ან h-ზე ნაკლებია.

თქვენი ამოცანაა დაადგინოთ კედლის საბოლოო ფორმა.

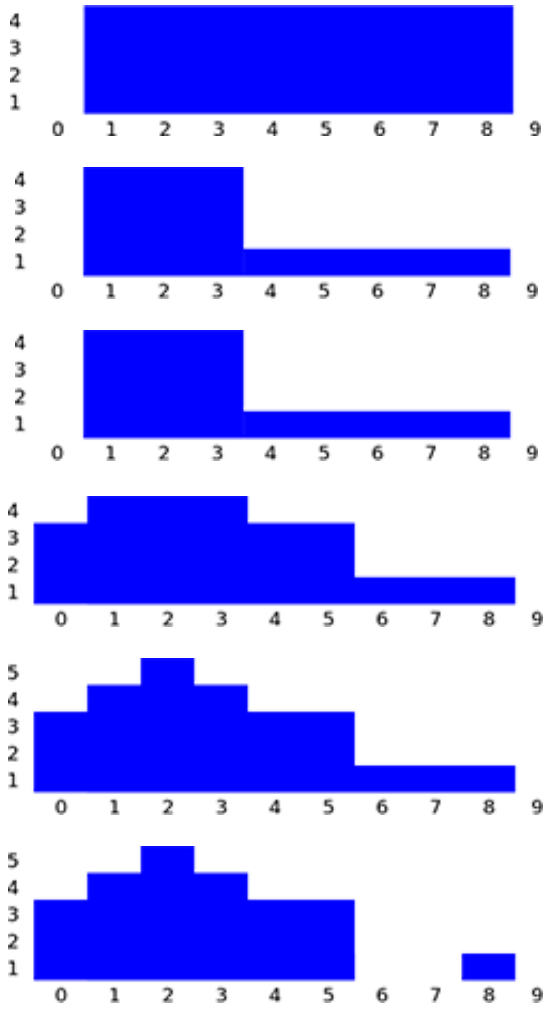
მაგალითი

დავუშვათ, მოცემულია აგურთა 10 სვეტი და კედლის მშენებლობის 6 ფაზა. ქვემოთ მოცემულია კედლის ფორმები თითოეული შესრულებული ფაზის შემდეგ.

ფაზა	ტიპი	დიაპაზონი	სიმაღლე
0	დამატება	სვეტები 1-დან 8-მდე	4
1	აღება	სვეტები 4-დან 9-მდე	1
2	აღება	სვეტები 3-დან 6-მდე	5
3	დამატება	სვეტები 0-დან 5-მდე	3
4	დამატება	სვეტები 2	5
5	აღება	სვეტები 6-დან 7-მდე	0

რადგანაც თავიდან ყველა სვეტი ცარიელია, ამიტომ 0-ვანი ფაზის შემდეგ თითოეულ სვეტში 1-დან 8-მდე 4 აგური გვექნება, 0-ვანი და მე-9 სვეტები კი

ცარიელი დარჩება. 1-ლი ფაზის შემდეგ აგურები აღებული იქნება მე-4-დან მე-8 სვეტის ჩათვლით მანამ, სანამ თითოეულ მათგანში თითო აგური არ დარჩება და მე-9 სვეტი ისევ ცარიელი რჩება. 0-დან 3-მდე სვეტებში, რომლებიც მოცემული დიაპაზონის გარეთაა, არაფერი არ იცვლება. მე-2 ფაზის შემდეგ ცვლილებები არ ხდება, რადგან არცერთი სვეტი 3-დან 6-მდე არ შეიცავს 5 აგურზე მეტს. მე-3 ფაზის შემდეგ აგურების რაოდენობა 0-ვანი, მე-4 და მე-5 სვეტებიდან თითოეულში სამამდე იზრდება. მე-4 ფაზის შემდეგ მე-2 სვეტში აგურების რაოდენობა 5-ის ტოლი გახდება. და საბოლოოდ, მე-5 ფაზის შემდეგ მე-6 და მე-7 სვეტებიდან ყველა აგური აღებული იქნება.



ამოცანა

k რაოდენობის ფაზის მოცემული აღწერისათვის გამოთვალეთ აგურების რაოდენობა თითოეულ სვეტში ყველა ფაზის შესრულების შემდეგ. თქვენ უნდა შექმნათ ფუნქცია `buildWall`.

- `buildWall(n, k, op, left, right, height, finalHeight)`
  - `n`: კედელში სვეტების რაოდენობა.
  - `k`: ფაზების რაოდენობა.
  - `op`: k სიგრძის მასივი; `op[i]` არის i-ური ფაზის ტიპი: იგი ტოლია 1-ის დამატების ფაზისათვის და 2-ის - აღების ფაზისათვის ($0 \leq i \leq k-1$).

- `left` და `right`: k სიგრძის მასივები; სვეტების დიაპაზონი i-ურ ფაზაში იწყება `left[i]` სვეტიდან და მთავრდება `right[i]` სვეტში (`left[i]` და `right[i]` ბოლოების ჩათვლით) ($0 \le i \le k-1$). ყველა ფაზაში `left[i]` $\le$ `right[i]`.
- `height`: k სიგრძის მასივი; `height[i]` წარმოადგენს i-ური ფაზის პარამეტრს ($0 \le i \le k-1$).
- `finalHeight`: n სიგრძის მასივი; თქვენ უნდა დააბრუნოთ თქვენი შედეგები i-ურ სვეტში აგურების საბოლოო რაოდენობის `finalHeight[i]`-ში მოთავსებით ($0 \le i \le n-1$).

ქვეამოცანები

თითოეული ქვეამოცანის ყველა ფაზაში სიმაღლეების მნიშვნელობები $100,000$-ზე ნაკლები ან ტოლი არაუარყოფითი მთელი რიცხვებია.

ქვეამოცანა	ქულები	n	k	შენიშვნა
1	8	$1 \le n \le 10,000$	$1 \le k \le 5,000$	დამატებითი შეზღუდვების გარეშე
2	24	$1 \le n \le 100,000$	$1 \le k \le 500,000$	დამატების ყველა ფაზა აღების ყველა ფაზის წინაა
3	29	$1 \le n \le 100,000$	$1 \le k \le 500,000$	დამატებითი შეზღუდვების გარეშე
4	39	$1 \le n \le 2,000,000$	$1 \le k \le 500,000$	დამატებითი შეზღუდვების გარეშე

იმპლემენტაციის დეტალები

თქვენ უნდა წარმოადგინოთ ზუსტად ერთი ფაილი, სახელით `wall.c`, `wall.cpp` ან `wall.pas`. ეს ფაილი იმპლემენტაციას უკეთებს ზემოთ აღწერილ ქვეპროგრამას მითითებული შაბლონების გამოყენებით. თქვენ ასევე გჭირდებათ `wall.h` ფაილი (C/C++)-ში იმპლემენტაციისათვის.

C/C++ პროგრამა

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

Pascal პროგრამა

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

სანიმუშო გრადერი

სანიმუშო გრადერი კითხულობს შესატან მონაცემებს შემდეგ ფორმატში:

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