Digital krets

Det finns en krets som består av $N + M$ **grindar** numrerade från 0 till $N + M - 1$. Grindarna 0 till $N - 1$ är **tröskelgrindar**, medan grindarna N till $N + M - 1$ är **källgrindar**.

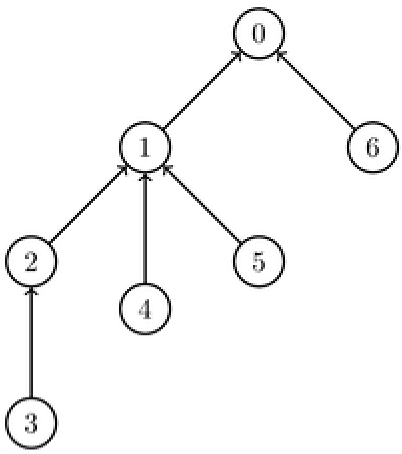
Varje grind, förutom grind 0, är en **ingång** till exakt en tröskelgrind. Mer exakt, för varje i så att $1 \leq i \leq N + M - 1$, så är grind i en ingång till grind $P[i]$, där $0 \leq P[i] \leq N - 1$. Det gäller alltid att $P[i] < i$. Dessutom låter vi $P[0] = -1$. Varje tröskelgrind har en eller flera ingångar. Källgrindar har inga ingångar.

Varje grind har ett **tillstånd** som antingen är 0 eller 1. Det ursprungliga tillståndet för varje källgrind ges av en vektor A med M heltal. Det vill säga, för varje j så att $0 \leq j \leq M - 1$, så är det ursprungliga tillståndet för källgrind $N + j$ lika med $A[j]$.

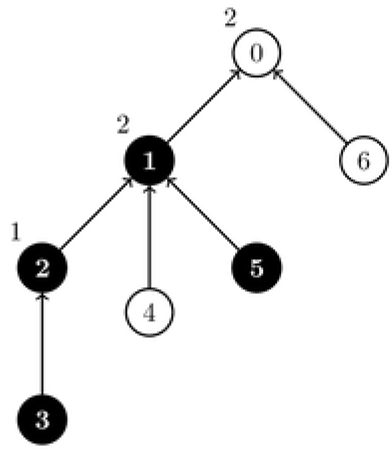
Tillståndet för varje tröskelgrind beror på tillstånden av dess ingångar samt bestäms på följande vis. Först ges varje tröskelgrind en **tröskelparameter**. Parametern som ges till en grind med k ingångar måste vara ett heltal från 1 till k (inklusive). Tillståndet för en tröskelgrind med parameter p är 1 om minst p av dess ingångar har tillståndet 1, annars är det 0.

Till exempel, antag att det finns $N = 3$ tröskelgrindar samt $M = 4$ källgrindar. Ingångar till grind 0 är grindarna 1 samt 6, samt ingångarna till grind 1 är grindarna 2, 4, samt 5, medan den enda ingången till grind 2 är grind 3.

Detta exempel illustreras i den följande bilden.



Antag att källgrindarna 3 samt 5 har tillståndet 1, medan källgrindarna 4 samt 6 har tillstånd 0. Antag att vi tilldelar parametrarna 1, 2, 2 till tröskelgrindarna 2, 1, 0 respektive. I detta fall har grind 2 tillstånd 1, grind 1 har tillstånd 1, medan grind 0 har tillstånd 0. Denna tilldelning av parametervärdena med dess tillstånd illustreras i följande bild. Grindar vars tillstånd är 1 är markerade i svart.



Tillstånden för källgrindarna kommer utsättas för Q uppdateringar. Varje uppdatering beskrivs av två heltal L samt R ($N \leq L \leq R \leq N + M - 1$) som flippar tillståndet på alla källgrindar numrerade från L till R, inklusive. Det vill säga, för varje i sådant att $L \leq i \leq R$ ändrar källgrind i sitt tillstånd till 1 om det var 0, samt 0 om det var 1. Det nya tillståndet för en flippad grind kommer inte ändras förrän det möjligtvis flippas av en senare uppdatering.

Ditt mål är att räkna, efter varje uppdatering, antalet olika tilldelningar av parametrar till tröskelgrindarna som resulterar i att grind 0 får tillstånd 1. Två tilldelningar anses olika om det finns minst en tröskelgrind vars värde på dess tröskelparameter skiljer sig mellan de två tilldelningarna. Eftersom antalet sätt kan vara oerhört stort ska du beräkna resten när talet delas med $1\ 000\ 002\ 022$.

Notera att i exemplet ovan finns det totalt 6 olika tilldelningar av parametrar till tröskelgrindarna, eftersom grind 0, 1, 2 har 2, 3, 1 ingångar. I två av dessa 6 tilldelningar har grind 0 tillståndet 1.

Implementation Details

Your task is to implement two procedures.

```
void init(int N, int M, int[] P, int[] A)
```

- N: the number of threshold gates.
- M: the number of source gates.
- P: an array of length $N + M$ describing the inputs to the threshold gates.
- A: an array of length M describing the initial states of the source gates.
- This procedure is called exactly once, before any calls to `count_ways`.

```
int count_ways(int L, int R)
```

- L, R: the boundaries of the range of source gates, whose states are toggled.
- This procedure should first perform the specified update, and then return the number of ways, modulo $1\ 000\ 002\ 022$, of assigning parameters to the threshold gates, which result in gate 0 having state 1.
- This procedure is called exactly Q times.

Example

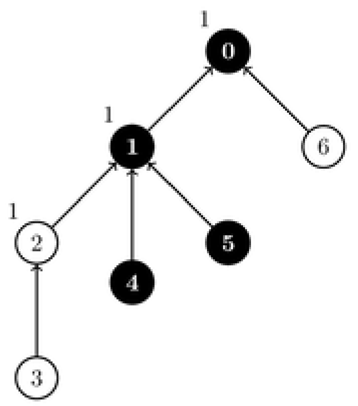
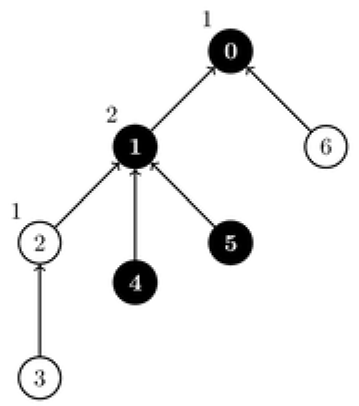
Consider the following sequence of calls:

```
init(3, 4, [-1, 0, 1, 2, 1, 1, 0], [1, 0, 1, 0])
```

This example is illustrated in the task description above.

```
count_ways(3, 4)
```

This toggles the states of gates 3 and 4, i.e. the state of gate 3 becomes 0, and the state of gate 4 becomes 1. Two ways of assigning the parameters which result in gate 0 having state 1 are illustrated in the pictures below.

Way 1	**Way** 2
	

In all other assignments of parameters, gate 0 has state 0. Thus, the procedure should return 2.

```
count_ways(4, 5)
```

This toggles the states of gates 4 and 5. As a result, all source gates have state 0, and for any assignment of parameters, gate 0 has state 0. Thus, the procedure should return 0.

```
count_ways(3, 6)
```

This changes the states of all source gates to 1. As a result, for any assignment of parameters, gate 0 has state 1. Thus, the procedure should return 6.

Constraints

- $1 \le N, M \le 100\,000$
- $1 \le Q \le 100\,000$
- $P[0] = -1$
- $0 \le P[i] < i$ and $P[i] \le N - 1$ (for each i such that $1 \le i \le N + M - 1$)
- Each threshold gate has at least one input (for each i such that $0 \le i \le N - 1$ there exists an index x such that $i < x \le N + M - 1$ and $P[x] = i$).
- $0 \le A[j] \le 1$ (for each j such that $0 \le j \le M - 1$)
- $N \le L \le R \le N + M - 1$

Subtasks

1. (2 points) $N = 1$, $M \le 1000$, $Q \le 5$
2. (7 points) $N, M \le 1000$, $Q \le 5$, each threshold gate has exactly two inputs.
3. (9 points) $N, M \le 1000$, $Q \le 5$
4. (4 points) $M = N + 1$, $M = 2^z$ (for some positive integer z), $P[i] = \lfloor \frac{i-1}{2} \rfloor$ (for each i such that $1 \le i \le N + M - 1$), $L = R$
5. (12 points) $M = N + 1$, $M = 2^z$ (for some positive integer z), $P[i] = \lfloor \frac{i-1}{2} \rfloor$ (for each i such that $1 \le i \le N + M - 1$)
6. (27 points) Each threshold gate has exactly two inputs.
7. (28 points) $N, M \le 5000$
8. (11 points) No additional constraints.

Sample Grader

The sample grader reads the input in the following format:

- line 1: $N \; M \; Q$
- line 2: $P[0] \; P[1] \; \ldots \; P[N + M - 1]$
- line 3: $A[0] \; A[1] \; \ldots \; A[M - 1]$
- line $4 + k$ ($0 \le k \le Q - 1$): $L \; R$ for update k

The sample grader prints your answers in the following format:

- line $1 + k$ ($0 \le k \le Q - 1$): the return value of `count_ways` for update k