Цифрова схема

Є схема, що складається з $N + M$ **воріт** пронумерованих від 0 до $N + M - 1$. Ворота від 0 до $N - 1$ - це **порогові ворота**, а ворота від N до $(N + M - 1)$ - це **початкові ворота**.

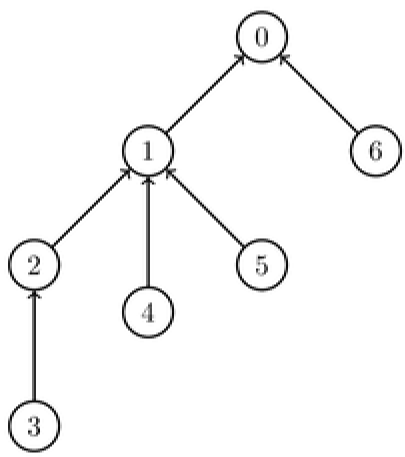
Кожні ворота, окрім 0, це **вхідні дані** (інпут) до рівно одних порогових воріт. Більш конкретно, для кожного i такого, що $1 \le i \le N + M - 1$, ворота i - це вхідні дані для воріт $P[i]$, де $0 \le P[i] \le N - 1$. Крім цього, $P[i] < i$. Більш того, ми припускаємо, що $P[0] = -1$. Кожні порогові ворота мають принаймні одні вхідні дані. Початкові ворота не мають жодних вхідних даних.

Кожні ворота мають **стан**, який або 0, або 1. Початкові стани початкових воріт зазначаються у масиві A з M цілих чисел. Тобто, для кожного j такого, що $0 \le j \le M - 1$, початковий стан початкових воріт $(N + j)$ - це $A[j]$.

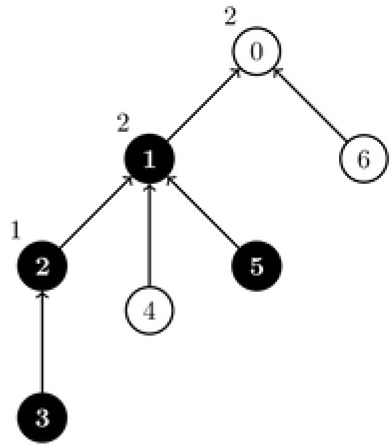
Стан кожних порогових воріт залежить від стану вхідних даних цих воріт і визначається так. Спочатку, кожним пороговим воротам присвоюють пороговий **параметр**. Параметр, присвоєний пороговим воротам з c вхідними даними, має бути цілим числом від 1 до c (включно). Потім стан порогових воріт з параметром p стає 1, якщо принаймні p з його вхідних даних мають стан 1; і стає 0 інакше.

Наприклад, припустимо, що всього є $N = 3$ порогових воріт, а також $M = 4$ початкових воріт. Вхідними даними до воріт 0 є ворота 1 та 6; вхідними даними до воріт 1 є ворота 2, 4 та 5; вхідними даними до воріт 2 є ворота 3.

Цей приклад зображений на наступному малюнку.



Припустимо, що початкові ворота 3 та 5 мають стан 1, а початкові ворота 4 та 6 мають стан 0. Припустимо, що ми присвоюємо параметри 1, 2 та 2 до порогових воріт 2, 1 та 0 відповідно. У цьому випадку ворота 2 мають стан 1, ворота 1 мають стан 1, а ворота 0 мають стан 0. Присвоєння параметрів і станів зображено на наступному малюнку. Ворота зі станом 1 зображені чорним.



Стани початкових воріт будуть змінюватися відповідно до Q запитів. Кожен запит описується двома цілими числами L та R ($N \le L \le R \le N + M - 1$), які змінюють стани усіх початкових воріт від L до R включно. Тобто, для кожного i такого, що $L \le i \le R$, початкові ворота i змінять свій стан на 1, якщо до цього стан був 0; або на 0, якщо початковий стан був 1. Новий стан кожних змінених воріт залишається незмінним до наступної, можливої, зміни у запиті.

Ваше завдання полягає у тому, щоб після кожного оновлення, знайти скільки різних присвоєнь параметрів у порогових воротах дають в результаті, що ворота 0 мають стан 1. Два присвоєння вважаються різними, якщо є принаймні одні порогові ворота, які мають різні параметри у різних присвоювань. Оскільки кількість може бути дуже великою, вам потрібно порахувати за модулем $1\;000\;002\;022$.

Зверніть увагу, що у прикладі вище є всього 6 різних присвоювань оскільки ворота 0, 1 та 2 мають 2, 3 та 1 вхідних даних відповідно. У 2 з 6 можливих присвоювань, ворота 0 мають стан 1.

Деталі реалізації

Вам потрібно реалізувати наступні функції:

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

- N: кількість порогових воріт.
- M: кількість початкових воріт.
- P: масив довжини $N + M$, який описує вхідні дані до порогових воріт.
- A: масив довжини M, який описує початкові стани початкових воріт.

- Ця функція викликається рівно один раз перед викликами функції `count_ways`.

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

- L, R: межі діапазонів початкових воріт, чиї стани змінюються.
- Ця функція має спочатку виконати описане оновлення, а потім повернути кількість різних способів присвоювань параметрів так, щоб ворота 0 мали стан 1, за модулем $1\ 000\ 002\ 022$.
- Ця функція викликається рівно Q разів.

Приклад

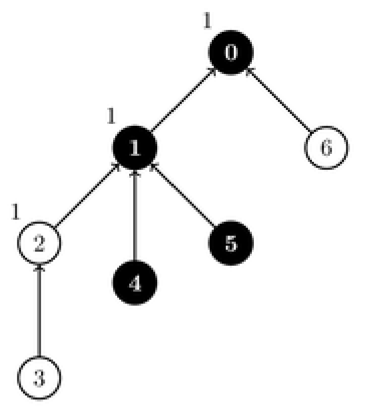
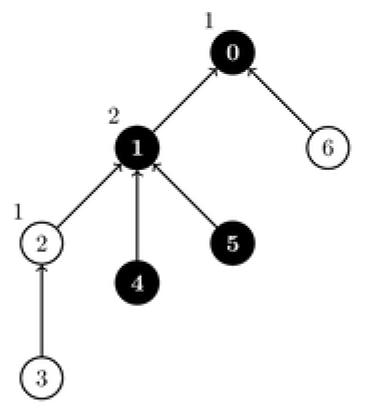
Розглянемо наступну послідовність викликів:

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

Цей приклад проілюстровано в описі завдання вище.

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

Ця функція змінює стани воріт 3 та 4; тобто, стан воріт 3 стане 0, а стан воріт 4 стане 1. Два способи присвоювань, які призведуть до того, що ворота 0 мають стан 1, зображені нижче.

<table>
<tr><th>Спосіб 1</th><th>Спосіб 2</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

В усіх інших присвоювань ворота 0 мають стан 0. Тому функція поверне 2.

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

Ця функція змінює стани воріт 4 та 5. Як результат, всі початкові ворота матимуть стан 0, і для будь-якого присвоювання ворота 0 мають стан 0. Тому функція поверне 0.

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

Ця функція змінює стани усіх початкових воріт на 1. Як результат, для будь-якого присвоювання ворота 0 мають стан 1. Тому функція поверне 6.

Обмеження

- $1 \leq N, M \leq 100\,000$
- $1 \leq Q \leq 100\,000$
- $P[0] = -1$
- $0 \leq P[i] < i$ та $P[i] \leq N - 1$ (для всіх i таких, що $1 \leq i \leq N + M - 1$)
- Кожні порогові ворота мають як мінімум одні вхідні дані (для кожного i такого, що $0 \leq i \leq N - 1$, існує індекс x такий, що $i < x \leq N + M - 1$ та $P[x] = i$).
- $0 \leq A[j] \leq 1$ (для кожного j такого, що $0 \leq j \leq M - 1$)
- $N \leq L \leq R \leq N + M - 1$

Підзадачі

1. (2 бали) $N = 1, M \leq 1000, Q \leq 5$
2. (7 балів) $N, M \leq 1000$, $Q \leq 5$, кожні порогові ворота мають рівно два інпута (вхідних даних).
3. (9 балів) $N, M \leq 1000, Q \leq 5$
4. (4 бали) $M = N + 1$, $M = 2^z$ (для певного додатного цілого числа z), $P[i] = \lfloor \frac{i-1}{2} \rfloor$ (для кожного i такого, що $1 \leq i \leq N + M - 1$), $L = R$
5. (12 балів) $M = N + 1$, $M = 2^z$ (для певного додатного цілого числа z), $P[i] = \lfloor \frac{i-1}{2} \rfloor$ (для кожного i такого, що $1 \leq i \leq N + M - 1$)
6. (27 балів) Кожні проміжні ворота мають рівно два інпута (вхідних даних).
7. (28 балів) $N, M \leq 5000$
8. (11 балів) Без додаткових обмежень.

Приклад градера

Градер зчитує вхідні дані у наступному форматі:

- 1-й рядок: $N \; M \; Q$
- 2-й рядок: $P[0] \; P[1] \; \ldots \; P[N + M - 1]$
- 3-й рядок: $A[0] \; A[1] \; \ldots \; A[M - 1]$
- $(4 + k)$-й рядок ($0 \leq k \leq Q - 1$): $L \; R$ для запиту k

Градер виводить ваші відповіді у наступному форматі:

- $(1+k)$-й рядок ($0 \le k \le Q-1$): повертає значення `count_ways` для запиту k