Програма "Зір"

Ви реалізуєте програму зору робота. Кожен раз, коли камера робота робить знімок, він зберігається як чорно-біле зображення в пам'яті робота. Кожне зображення є сіткою пікселів $H \times W$, рядки пронумеровані від 0 до $H-1$, а стовпці пронумеровані від 0 до $W-1$. У кожному зображенні є **рівно два** чорних пікселя, а всі інші пікселі - білі.

Робот може обробляти кожне зображення програмою, що складається з простих інструкцій. Вам даються значення H, W і додатне ціле число K. Ваша мета - написати процедуру, що буде створювати програму для робота, яка для будь-якого зображення визначає, чи **відстань** між двома чорними пікселями рівна K. Тут відстань між пікселем у рядку r_1 та стовпці c_1 та пікселем у рядку r_2 та стовпці c_2 становить $|r_1 - r_2| + |c_1 - c_2|$. У цій формулі $|x|$ позначає абсолютне значення x, яке дорівнює x, якщо $x \geq 0$ і дорівнює $-x$, якщо $x < 0$.

Зараз ми опишемо, як працює робот.

Пам'ять робота - це досить великий масив комірок, пронумерований від 0. Кожна комірка може зберігати або 0, або 1, і її значення після заповнення вже не міняється. Зображення зберігається рядок за рядком у клітинках, індексованих від 0 до $H \cdot W - 1$. Перший рядок зберігається в клітинках від 0 до $W-1$, а останній рядок зберігається в клітинках від $(H-1)W$ до $H \cdot W - 1$. Зокрема, якщо піксель у рядку i та стовпці j є чорним, значення комірки $i \cdot W + j$ становить 1, інакше - 0.

Програма робота - це послідовність **інструкцій**, які пронумеровані послідовними цілими числами, починаючи з 0. Коли програма запускається, інструкції виконуються по черзі. Кожна інструкція зчитує значення однієї або декількох комірок (ми називаємо ці значення **входами** інструкції) і видає єдине значення, що дорівнює 0 або 1 (це значення ми називаємо **виходом** інструкції). Вихід інструкції i зберігається у комірці $H \cdot W + i$. Входами інструкції i можуть бути лише комірки, які зберігають або пікселі, або виходи попередніх інструкцій, тобто комірки від 0 до $H \cdot W + i - 1$.

Існує чотири типи інструкцій:

- `NOT`: має рівно один вхід. Його вихід становить 1, якщо вхід становить 0, інакше його вихід становить 0.
- `AND`: має один або кілька входів. Його вихід становить 1, тоді і тільки тоді, коли **всі** значення входів є 1.
- `OR`: має один або кілька входів. Його вихід становить 1, тоді і тільки тоді, коли

принаймні один із входів становить 1.
- XOR: має один або кілька входів. Його вихід становить 1, тоді і тільки тоді, коли **непарне число** входів становить 1.

Вихід останньої інструкції програми повинен становити 1, якщо відстань між двома чорними пікселями рівна K, а в іншому випадку - 0.

Деталі реалізації

Вам слід реалізувати таку процедуру:

```
void construct_network (int H, int W, int K)
```

- H, W: розміри кожного зображення, зробленого камерою робота
- K: додатне ціле число
- Ця процедура повинна створити програму робота. Для будь-якого зображення, зробленого камерою робота, ця програма повинна визначити, чи дорівнює K відстань між двома чорними пікселями на зображенні.

Ця процедура повинна викликати одну або кілька з наступних процедур для додавання інструкцій до програми робота (яка спочатку порожня):

```
int add_not (int N) int add_and (int [] Ns) int add_or (int [] Ns) int add_xor (int [] Ns)
```

- Додає інструкцію NOT, AND, OR або XOR відповідно.
- N (для add_not): індекс комірки, з якої додана інструкція NOT, читає свій вхід
- Ns (для add_and, add_or, add_xor): масив, що містить індекси комірок, з яких додана інструкція AND, OR або XOR читає свої входи
- Кожна процедура повертає індекс комірки, в якій зберігається вихід інструкції. Послідовні звернення до цих процедур повертають послідовні цілі числа, починаючи з $H \cdot W$.

Програма робота може складатися з максимум $10\,000$ інструкцій. Інструкції можуть прочитати щонайбільше $1\,000\,000$ значень. Іншими словами, загальна довжина масивів Ns у всіх викликах до add_and, add_or та add_xor плюс кількість викликів до add_not не може перевищувати $1\,000\,000$.

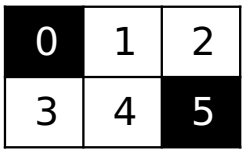
Після додавання останньої інструкції процедура construct_network повинна завершити виконання. Потім програма робота буде оцінена на деякій кількості зображень. Ваше розв’язання проходить даний тест, якщо для кожного з цих зображень результат останньої інструкції становить 1, тоді і лише тоді, коли відстань між двома чорними пікселями на зображенні дорівнює K.

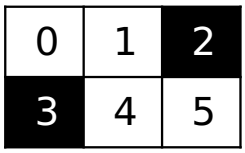
Оцінювання вашого розв'язку може призвести до одного з наступних повідомлень англійською мовою, які пояснюються нижче:

- `Instruction with no inputs`: до `add_and`, `add_or` або `add_xor` було передано порожній масив.
- `Invalid index`: неправильний (можливо від'ємний) номер комірки було передано до `add_and`, `add_or`, `add_xor` або `add_not`.
- `Too many instructions`: ваша процедура спробувала додати більше ніж $10\,000$ інструкцій.
- `Too many inputs`: інструкції сумарно зчитали більше ніж $1\,000\,000$ значень.

Приклад

Припустимо, що $H = 2$, $W = 3$, $K = 3$. Існує лише 2 зображення на яких відстань між чорними пікселями становитеме 3.





- Випадок 1: чорні пікселі 0 і 5
- Випадок 2: чорні пікселі 2 і 3

Можливим варіантом розв’язання буде побудова програми робота роблячи наступні виклики:

1. `add_and([0, 5])`, що додає інструкцію, яка виводить 1 тоді і лише тоді коли має місце перший випадок. Вихід записується до клітинки 6.
2. `add_and([2, 3])`, що додає інструкцію, яка виводить 1 тоді і лише тоді коли має місце другий випадок. Вихід записується до клітинки 7.
3. `add_or([6, 7])`, що додає інструкцію, яка виводить 1 тоді і лише тоді коли має місце один з перечислених вище випадків.

Обмеження

- $1 \leq H \leq 200$
- $1 \leq W \leq 200$
- $2 \leq H \cdot W$
- $1 \leq K \leq H + W - 2$

Підзадачі

1. (10 балів) $\max(H, W) \leq 3$
2. (11 балів) $\max(H, W) \leq 10$
3. (11 балів) $\max(H, W) \leq 30$
4. (15 балів) $\max(H, W) \leq 100$

5. (12 балів) $\min(H, W) = 1$
6. (8 балів) Піксель в рядку 0 та стовпці 0 чорний на всіх зображеннях.
7. (14 балів) $K = 1$
8. (19 балів) Без додаткових обмежень.

Приклад модуля перевірки

Модуль перевірки з прикладу зчитує вхідні дані в наступному форматі:

- рядок 1: $H \;\; W \;\; K$
- рядок $2 + i$ $(i \geq 0)$: $r_1 \;\; c_1 \;\; r_2 \;\; c_2$
- останній рядок: -1

Кожен із рядків за винятком першого і останнього представляє собою зображення з двома чорними пікселями. Позначатимемо зображення в рядку $2 + i$, як зображення i. Перший чорний піксель знаходиться в рядку $r_1[i]$ і стовпці $c_1[i]$, а другий у рядку $r_2[i]$ і стовпці $c_2[i]$.

Модуль перевірки з прикладу спочатку викликає `construct_network(H, W, K)`. Якщо `construct_network` порушує деякі обмеження, описанні в умові задачі, модуль перевірки з прикладу друкує одне із повідомлень про помилку, перелічених в кінці секції Деталі реалізації, та завершує виконання.

В іншому випадку, модуль перевірки з прикладу генерує два результати.

По-перше, він друкує вихід програми робота у наступному форматі:

- рядок $1 + i$ $(0 \leq i)$: вихід останньої інструкції програми робота для зображення i (1 або 0).

По-друге, він записує до поточної папки файл `log.txt` у наступному форматі:

- рядок $1 + i$ $(0 \leq i)$: $m[i][0] \;\; m[i][1] \;\; \ldots \;\; m[i][c-1]$

Послідовність у рядку $1 + i$ описує величини, записані в пам'яті робота після роботи програми із зображенням i. А саме, $m[i][j]$ описує значення величини в комірці j. Зауважимо, що значення c (довжина послідовності) дорівнює $H \cdot W$ плюс кількість інструкцій у програмі робота.