Vision Program

당신은 로봇을 위한 비전 프로그램을 구현하고 있다. 매번 로봇이 사진을 찍을 때 마다 사진은 로봇의 메모리에 흑백 이미지로 저장된다. 각 이미지는 $H \times W$ 크기의 픽셀들이다. 이미지의 행은 0부터 $H-1$까지, 열은 0부터 $W-1$까지 번호가 붙어 있다. 이미지에는 **정확히 2개의** 검은 색 픽셀이 있고 나머지는 모두 흰색이다.

로봇은 간단한 명령들로 구성된 프로그램을 받아서 이미지를 처리할 수 있다. 당신에게는 H, W와 어떤 양의 정수 K가 주어진다. 당신이 할 일은 이미지에서 2개의 검은 색 픽셀간의 **거리**가 정확히 K인지 판단할 수 있는 로봇 용 프로그램을 작성하는 것이다. 여기서, 행 r_1, 열 c_1에 있는 픽셀과 행 r_2 열 c_2에 있는 픽셀 간의 거리는 $|r_1 - r_2| + |c_1 - c_2|$이다.여기서, $|x|$는 x의 절대값이다. 즉, $|x|$는 x가 0 이상이면 x와 같고, x가 음수이면 $-x$와 같다.

로봇이 어떻게 동작하는지 아래에 설명한다.

로봇의 메모리는 충분히 많은 셀들의 배열이다. 배열의 인덱스는 0에서 시작한다. 각 셀은 0 혹은 1을 저장할 수 있고, 한번 값이 저장되면 바뀌지 않는다. 이미지는 0번 셀부터 $H \cdot W - 1$번 셀까지에 한 행씩 저장된다. 이미지의 첫번째 행은 0번 셀부터 $W-1$번 셀까지에 저장되어 있고, 이미지의 마지막 행은 $(H-1) \cdot W$ 번 셀부터 $H \cdot W - 1$번 셀까지에 저장되어 있다. 이미지의 행 i, 열 j에 있는 픽셀이 검은색이면 $i \cdot W + j$번 셀에 있는 값은 1이고, 그렇지 않으면 0이다.

로봇이 수행하는 프로그램은 **명령**의 시퀀스이다. 명령들은 0번부터 순서대로 번호가 붙어 있다. 프로그램이 실행될 때, 명령들은 하나씩 순서대로 수행된다. 각 명령은 하나 혹은 그 이상의 셀들의 값을 읽고 (읽은 값들을 명령의 **입력**이라고 부른다) 0 혹은 1인 하나의 값을 생성한다 (생성된 값을 명령의 **출력**이라고 부른다). i번 명령의 출력은 $H \cdot W + i$번 셀에 저장된다. i번 명령의 입력은 픽셀들이 저장된 셀들 혹은 이전에 수행된 명령들의 출력이 저장된 셀들만 가능하다. 즉, 0번 셀부터 $H \cdot W + i - 1$번 셀까지만 가능하다.

명령은 4가지가 있다.

- `NOT`: 정확히 하나의 입력을 가진다. 입력이 0이면 출력은 1이고, 입력이 1이면 출력은 0이다.
- `AND`: 하나 이상의 입력을 가진다. 출력은 모든 입력이 1인 경우 (그리고 그 경우에만) 1이다.
- `OR`: 하나 이상의 입력을 가진다. 출력은 입력중 적어도 하나가 1이면 1이, 입력이 모두 0이면 0이다.
- `XOR`: 하나 이상의 입력을 가진다. 출력은 입력들 중 1인 것들의 개수가 홀수일 때 (그리고 그 경우에만) 1이다.

프로그램의 마지막 명령의 출력은, 검은 색인 두 픽셀의 거리가 정확히 K인 경우에 1이라야 하고, 그렇지 않은 경우 0이라야 한다.

Implementation details

다음 함수를 구현해야 한다.

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

- H, W: 로봇의 카메라가 찍은 이미지의 크기
- K: 양의 정수
- 이 함수는 로봇을 위한 프로그램을 생성해야 한다. 그 프로그램은 로봇의 카메라가 찍은 이미지에서 두 검은 색 픽셀 간의 거리가 정확히 K인지 판단해야 한다.

이 함수는 로봇을 위한 프로그램에 명령을 (제일 뒤에) 추가하기 위해 다음 함수를 호출할 수 있다. 처음에는 로봇을 위한 프로그램은 비어 있다.

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

- 각각 NOT, AND, OR, or XOR 명령을 추가한다.
- N (add_not에서): 추가하는 NOT 명령이 읽을 셀의 번호이다.
- Ns (add_and, add_or, add_xor에서): 추가하는 AND, OR, XOR 명령들이 읽을 셀들의 번호들을 저장한 배열이다.
- 각 함수 호출은 추가된 명령어의 출력을 저장하는 셀의 번호를 리턴한다. 즉, 이 함수들을 호출할 때마다 $H \cdot W$에서 시작해서 연속으로 증가하는 번호들이 리턴된다.

로봇을 위한 프로그램은 최대 $10\,000$개의 명령을 가질 수 있다. 명령들은 (모두 합해서) $1\,000\,000$개의 값들을 읽을 수 있다. 즉, add_and, add_or, add_xor 함수의 모든 호출에서 사용된 인자 배열들의 크기의 합과 add_not 함수의 호출 횟수의 합은 $1\,000\,000$을 넘을 수 없다.

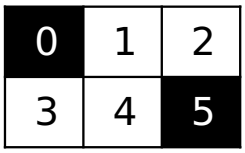
제일 마지막 명령을 추가한 후, construct_network 함수는 리턴해야 한다. 그 이후, 서버는 생성된 프로그램을 몇개의 이미지에 수행해 볼 것이다. 시험해 본 모든 이미지들에 대해서 생성된 프로그램이 정확한 결과(검은 색 픽셀 간의 거리가 정확히 K인 경우에만 마지막 명령이 1을 출력)를 만들어 내는 경우에 테스트를 통과한 것으로 간주될 것이다.

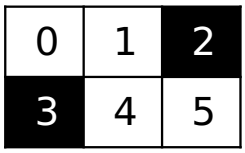
프로그램을 수행한 결과로 다음의 오류 메시지 중 하나가 생성될 수 있다.

- Instruction with no inputs: add_and, add_or, add_xor의 인자로 빈 배열이 주어졌다.
- Invalid index: add_and, add_or, add_xor, add_not의 인자로 잘못된 (음수 번호일 수도 있는) 셀 번호가 주어졌다.
- Too many instructions: $10\,000$개를 초과하는 명령을 수행하려는 시도가 있었다.
- Too many inputs: 입력으로 읽은 셀의 개수가 $1\,000\,000$를 넘었다.

Example

$H = 2, W = 3, K = 3$이라고 하자. 두 검은 색 픽셀 간의 거리가 정확히 3인 이미지는 2개 밖에 없다.





- Case 1: 0과 5가 검은 색이다.
- Case 2: 2와 3이 검은 색이다.

가능한 해결책 중 하나는 다음과 같은 호출로 명령들을 만드는 것이다.

1. `add_and([0, 5])`, 이렇게 만든 명령은 첫번째 경우에만 1을 출력할 것이다. 출력은 6번 셀에 저장된다.
2. `add_and([2, 3])`, 이렇게 만든 명령은 두번째 경우에만 1을 출력할 것이다. 출력은 7번 셀에 저장된다.
3. `add_or([6, 7])`, 이렇게 만든 명령은 위의 두 명령 중 1을 출력한 것이 하나라도 있는 경우에만 1을 출력할 것이다.

Constraints

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

Subtasks

1. (10 points) $\max(H, W) \le 3$
2. (11 points) $\max(H, W) \le 10$
3. (11 points) $\max(H, W) \le 30$
4. (15 points) $\max(H, W) \le 100$
5. (12 points) $\min(H, W) = 1$
6. (8 points) 모든 이미지에서 행 0, 열 0에 있는 픽셀은 검은 색이다.
7. (14 points) $K = 1$
8. (19 points) 추가 제약 조건이 없음.

Sample grader

샘플 그레이더는 다음의 형식으로 입력을 읽는다.

- line 1: H W K
- line $2 + i$ $(i \ge 0)$: $r_1[i]$ $c_1[i]$ $r_2[i]$ $c_2[i]$
- last line: -1

첫 줄과 마지막 줄을 제외한 중간의 줄들은 각각 2개의 검은 색 픽셀을 포함한 이미지를 표현한다. $2 + i$

번째 줄에 주어진 이미지를 i번째 이미지라고 하자. 그 이미지에서, 행 $r_1[i]$, 열 $c_1[i]$과 행 $r_2[i]$, 열 $c_2[i]$에 있는 픽셀들이 검은 색이다.

샘플 그레이더는 우선 `construct_network(H, W, K)`를 호출한다. 만약 `construct_network`이 문제의 조건을 만족하지 않는 경우 Implementation details에 있는 오류 메시지 중 하나를 출력하고 종료한다.

오류가 없는 경우, 샘플 그레이더는 두가지 출력을 생성한다.

첫번째 출력은 다음과 같은 방식이며, 로봇을 위한 프로그램이 수행된 결과들이다.

- line $1 + i$ $(0 \le i)$: i번째 이미지에 대해 로봇을 위한 프로그램이 수행된 결과 (0 혹은 1).

두번째 출력은 현재 디렉토리의 `log.txt` 파일에 다음 형식으로 저장된다.

- line $1 + i$ $(0 \le i)$: $m[i][0]$ $m[i][1]$ $\ldots$ $m[i][c-1]$

$1 + i$번째 줄의 내용은 i번째 이미지에 대해서 프로그램이 수행된 이후 메모리 셀들에 저장된 값들이다. 정확히 말하면, $m[i][j]$는 j번 셀에 저장된 값이다. 한 줄에 있는 값들의 개수인 c는 $H \cdot W$과 프로그램의 명령 개수의 합이다.