Көрүү программасы

Сиз роботко көрүү программасын жазгыныз келет. Ар убакытта роботтун камерасы көрүнүштү алат, жана кара ак сүрөт түрүндө роботтун эсине сактайт. Ар бир сүрөт $H \times W$ пикселдердин таблицасы, саптары 0 дөн $(H-1)$ге жана мамычалары 0дөн $(W-1)$.ге чейин номерленген. Бир сүрөттө **так эки** кара пиксел калгандары ак писел.

Робот сүрөттү жөнөкөй буйруктардан турган программа аркылуу иштете алат. Сизге H, W маанилери жана K оң саны берилет. Сиздин максатыңыз робот үчүн программаны жасоочу процедураны жазуу, ал программа каалаган сүрөт үчүн эки кара пикселдин **аралыгы** Kга барабар болоор болбосун аныктайт.

Мында $(r_1$ жана $c_1)$инчи жана$(r_2$ жана $c_2)$инчи пикселдердин аралыгы $|r_1 - r_2| + |c_1 - c_2|$ге барабар.

Роботтун кандай иштегенин түшүндүрөбүз.

Роботтун эси аябай узун, 0дөн баштап номерленген уячалардын массиви түрүндө. Ар бир уяча 0 же 1ге барабар болот жана бир жолу коюлуп андан кийин алмашпайт. Сүрөт саптан сапка 0дөн $(H \cdot W - 1)$ге чейин сакталат. Биринчи сап 0дөн $(W-1)$ге, жана акыркы сапка $((H-1) \cdot W)$дан $(H \cdot W - 1)$ге чейин сактайт. Эгерде i , jигчи пиксел кара болсо анда ошол уяча 1ге барабар болбосо 0гө барабар.

Роботтун программасы **буйруктардын** удаалаштыгы түрүндө.Алар 0дөн баштап номерленген.

Алар бир бирден аткарылат. Ар бир буйрук бир же бирден көп уячалардын маанилерин окуп (алардын маанилери буйруктун **инпуттары** деп аталат) жана 0гө же 1ге барабар болгон жалгыз маанини чыгарат(ал маани буйруктун **аутпуту** деп аталат). i инчи буйруктун **аутпуту** $(H \cdot W + i)$инчи уячага сакталат. iинчи буйруктун инпуттары пикселдерди же алдын ала буйруктардын аутпуттарын кармаган уячалар гана болушу мүмкүн, башкача айтканда 0 инчиден $(H \cdot W + i - 1)$инчиге чейинки уячалар.

Буйруктун төрт түрү бар:

- `NOT`: буйругуна бир гана инпут кирет. Эгерде инпут 0гө барабар болсо, анда аутпут 1ге барабар болот, антпесе аотпут 0гө барабар болот.
- `AND`:буйругуна бир же бирканча инпут кирет. Эгерде бардык инпут 1ге

барабар болсо,анда аутпут 1 болот, болбосо 0 болот.
- `OR`: буйругуна бир же бирканча инпут кирет. Эгерде бардык инпут 0ге барабар болсо,анда аутпут 0 болот, болбосо 1 болот.
- `XOR`: буйругуна бир же бирканча инпут кирет. Эгерде бардык инпутагы 1лердин саны так болсо,анда аутпут 1 болот, болбосо 0 болот.

Эгерде эки кара пикселдин аралыгы K санына барабар болсо анда программанын акыркы буйруктун аутпуту 1ге барабар болуш керек, андай жок болсо акыркы буйруктун аутпуту 0гө барабар болуш керек.

Implementation details

You should implement the following procedure:

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

- H, W: dimensions of each image taken by the robot's camera
- K: a positive integer
- This procedure should produce a robot's program. For any image taken by the robot's camera, this program should determine whether the distance between the two black pixels in the image is exactly K.

This procedure should call one or more of the following procedures to append instructions to the robot's program (which is initially empty):

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

- Append a `NOT`, `AND`, `OR`, or `XOR` instruction, respectively.
- N (for `add_not`): the index of the cell from which the appended `NOT` instruction reads its input
- Ns (for `add_and`, `add_or`, `add_xor`): array containing the indices of the cells from which the appended `AND`, `OR`, or `XOR` instruction reads its inputs
- Each procedure returns the index of the cell that stores the output of the instruction. The consecutive calls to these procedures return consecutive integers starting from $H \cdot W$.

The robot's program can consist of at most $10\,000$ instructions. The instructions can read at most $1\,000\,000$ values in total. In other words, the total length of Ns arrays in all calls to `add_and`, `add_or` and `add_xor` plus the number of calls to `add_not` cannot exceed $1\,000\,000$.

After appending the last instruction, procedure `construct_network` should return. The

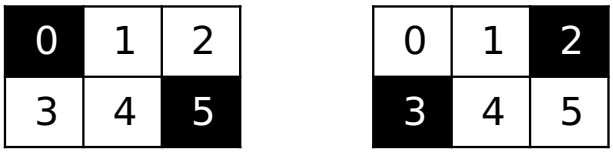
robot's program will then be evaluated on some number of images. Your solution passes a given test case if for each of these images, the output of the last instruction is 1 if and only if the distance between the two black pixels in the image is equal to K.

The grading of your solution may result in one of the following error messages:

- `Instruction with no inputs`: an empty array was given as the input to `add_and`, `add_or`, or `add_xor`.
- `Invalid index`: an incorrect (possibly negative) cell index was provided as the input to `add_and`, `add_or`, `add_xor`, or `add_not`.
- `Too many instructions`: your procedure attempted to add more than $10\,000$ instructions.
- `Too many inputs`: the instructions read more than $1\,000\,000$ values in total.

Example

Assume $H = 2$, $W = 3$, $K = 3$. There are only two possible images where the distance between the black pixels is 3.



- Case 1: black pixels are 0 and 5
- Case 2: black pixels are 2 and 3

A possible solution is to build a robot's program by making the following calls:

1. `add_and([0, 5])`, which adds an instruction that outputs 1 if and only if the first case holds. The output is stored in cell 6.
2. `add_and([2, 3])`, which adds an instruction that outputs 1 if and only if the second case holds. The output is stored in cell 7.
3. `add_or([6, 7])`, which adds an instruction that outputs 1 if and only if one of the cases above holds.

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) $\max(H, W) \leq 3$
2. (11 points) $\max(H, W) \leq 10$
3. (11 points) $\max(H, W) \leq 30$
4. (15 points) $\max(H, W) \leq 100$
5. (12 points) $\min(H, W) = 1$
6. (8 points) Pixel in row 0 and column 0 is black in each image.
7. (14 points) $K = 1$
8. (19 points) No additional constraints.

Sample grader

The sample grader reads the input in the following format:

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

Each line excepting the first and the last line represents an image with two black pixels. We denote the image described in line $2 + i$ by image i. One black pixel is in row $r_1[i]$ and column $c_1[i]$ and the other one in row $r_2[i]$ and column $c_2[i]$.

The sample grader first calls `construct_network(H, W, K)`. If `construct_network` violates some constraint described in the problem statement, the sample grader prints one of the error messages listed at the end of Implementation details section and exits.

Otherwise, the sample grader produces two outputs.

First, the sample grader prints the output of the robot's program in the following format:

- line $1 + i$ ($0 \leq i$): output of the last instruction in the robot's program for image i (1 or 0).

Second, the sample grader writes a file `log.txt` in the current directory in the following format:

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

The sequence on line $1 + i$ describes the values stored in the robot's memory cells after the robot's program is run, given image i as the input. Specifically, $m[i][j]$ gives the value of cell j. Note that the value of c (the length of the sequence) is equal to $H \cdot W$ plus the number of instructions in the robot's program.