

Vision Program

Trebuie să implementați un program de interpretare pentru un robot. De fiecare dată când robotul face o poză, aceasta este stocată în memoria robotului ca o imagine albnegru. Fiecare imagine este o matrice cu $H \times W$ pixeli, cu rândurile numerotate de la 0 la H-1 și coloanele numerotate de la 0 la W-1. Există **exact doi** pixeli negri în fiecare imagine si toti ceilalti pixeli sunt albi.

Robotul poate procesa fiecare imagine folosind un program format din instrucțiuni simple. Se dau valorile H, W și un întreg pozitiv K. Scopul vostru este să scrieți o funcție care să producă un program pentru robot care, pentru orice imagine primită, să determine dacă **distanța** dintre cei doi pixeli negri este exact K. Distanța între un pixel de pe linia r_1 și coloana c_1 , și un pixel de pe linia r_2 și coloana c_2 este $|r_1-r_2|+|c_1-c_2|$. În această formulă, |x| reprezintă valoarea absolută a lui x, care este x dacă $x \geq 0$ și -x dacă x < 0.

Acum vom descrie cum functionează robotul.

Memoria robotului este formată dintr-un vector suficient de mare de celule, indexat de la 0. Fiecare celulă este sau 0, sau 1, iar această valoare odată setată, nu poate fi modificată. Imaginea este memorată linie cu linie în celule indexate de la 0 la $H \cdot W - 1$. Prima linie este stocată în celulele de la 0 la W - 1, iar ultima linie este stocată în celulele de la $(H - 1) \cdot W$ la $H \cdot W - 1$. Mai exact, dacă pixelul de pe linia i si coloana i este negru, valoarea celulei $i \cdot W + i$ este i, altfel i.

Programul robotului este format dintr-o serie de **instrucțiuni**, numerotate prin întregi consecutivi începând de la 0. Când programul rulează, instrucțiunile sunt executate una câte una. Fiecare instrucțiune citește valorile din una sau mai multe celule (numim aceste valori **intrările** instrucțiunii) și produc o singură valoare egală cu 0 sau 1 (numim această valoare **ieșirea** instrucțiunii). Ieșirea instrucțiunii i este stocată în celula $H \cdot W + i$. Intrările instrucțiunii i pot fi celule care stochează fie pixeli, fie ieșirile instrucțiunilor precedente, adică celulele de la 0 la $H \cdot W + i - 1$.

Există patru tipuri de instrucțiuni:

- NOT: are o singură intrare. Ieșirea este 1 dacă intrarea e 0, altfel ieșirea este 0.
- AND: are una sau mai multe intrări. Ieșirea este 1 dacă și numai dacă **toate** intrările sunt 1.
- 0R: are una sau mai multe intrări. Ieșirea este 1 dacă și numai dacă **cel puțin una** din intrări este 1.
- XOR: are una sau mai multe intrări. Iesirea este 1 dacă si numai dacă un număr

impar de intrări sunt 1.

Ieșirea ultimei instrucțiuni a programului trebuie să fie 1 dacă distanța dintre cei doi pixeli negri este exact K, altfel ieșirea este 0.

Detalii de implementare

Trebuie să implementați următoarea funcție:

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

- H,W: dimensiunile fiecărei imagini făcute de camera robotului
- *K*: un întreg pozitiv
- Această funcție trebuie să implementeze programul robotului. Pentru orice imagine făcută de camera robotului, acest program trebuie să determine dacă distanța dintre cei doi pixeli negri din imagine este exact K.

Această funcție trebuie să apeleze una sau mai multe funcții din următoarele pentru a adăuga instrucțiuni programului robotului (care inițial e gol):

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

- Adaugă o instrucțiune NOT, AND, OR, respectiv XOR.
- \bullet N (pentru add_not): indicele celulei din care instrucțiunea NOT adăugată își citește intrarea.
- Ns (pentru add_and, add_or, add_xor): un vector ce conține indicii celulelor din care instrucțiunile adăugate AND, OR sau XOR își citesc intrările.
- Fiecare astfel de funcție returnează indicele celulei care stochează ieșirea instrucțiunii. Apelările consecutive ale acestor funcții returnează întregi consecutivi începând de la $H\cdot W$.

Programul robotului poate să conțină cel mult $10\,000$ de instrucțiuni. Instrucțiunile pot citi cel mult $1\,000\,000$ de valori în total. Cu alte cuvinte, lungimea totală a vectorilor Ns din apelurile funcțiilor add_and, add_or și add_xor, plus numărul de apeluri ale funcției add_not nu poate depăși $1\,000\,000$.

După adăugarea ultimei instrucțiuni, trebuie să ieșiți din funcția $construct_network$. După aceea, programul robotului va fi evaluat pe o serie de imagini. Soluția voastră trece un anumit test dacă pentru fiecare dintre aceste imagini, ieșirea ultimei instrucțiuni este 1 dacă și numai dacă distanța dintre cei doi pixeli negri din imagine este egală cu K.

Evaluarea soluției voastre poate returna unul din următoarele mesaje (scrise în limba engleză):

- Instruction with no inputs: a fost transmis un vector gol ca intrare pentru una din funcțiile add and, add or sau add xor.
- Invalid index: indicele unei celule transmis ca intrare pentru una din funcțiile add and, add or, add xor sau add not este incorect (posibil negativ).
- \bullet Too many instructions: funcția voastră a încercat să adauge mai mult de $10\,000$ de instructiuni.
- Too many inputs: instrucțiunile citesc mai mult de 1000000 de valori în total.

Exemple

Presupunem că H=2, W=3 și K=3. Există doar două posibile imagini pentru care distanța dintre cei doi pixeli negri este 3.

0	1	2
3	4	5

0	1	2
3	4	5

- Cazul 1: pixelii negri sunt 0 și 5
- Cazul 2: pixelii negri sunt 2 și 3

O posibilă soluție este să construim programul robotului prin apelarea următoarelor funcții:

- 1. add_and([0, 5]), va adăuga o instrucțiune care afișează 1 dacă și numai dacă cazul 1 este analizat. Ieșirea este stocată în celula 6.
- 2. add_and([2, 3]), va adăuga o instrucțiune care afișează 1 dacă și numai dacă cazul 2 este analizat. Ieșirea este stocată în celula 7.
- 3. add_or([6, 7]), va adăuga o instrucțiune care afișează 1 dacă și numai dacă unul din cele două cazuri este analizat.

Restricții

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

Subtask-uri

1. (10 puncte) $\max(H, W) \leq 3$

- 2. (11 puncte) $\max(H, W) \le 10$
- 3. (11 puncte) $\max(H, W) < 30$
- 4. (15 puncte) $\max(H, W) \leq 100$
- 5. (12 puncte) $\min(H, W) = 1$
- 6. (8 puncte) Pixelul de pe linia 0 și coloana 0 este negru în fiecare imagine.
- 7. (14 puncte) K = 1
- 8. (19 puncte) Fără restricții suplimentare.

Exemplu de grader

Grader-ul citește intrarea în următorul format:

- linia 1: H W K
- linia 2 + i $(i \ge 0)$: r_1 c_1 r_2 c_2
- ultima linie: -1

Fiecare linie, cu excepția primei și ultimei linii, reprezintă o imagine cu doi pixeli negri. Notăm imaginea descrisă pe linia 2+i, imaginea i. Un pixel negru se află pe linia r_1 și coloana c_1 , iar celălalt pe linia r_2 , coloana c_2 .

Grader-ul mai întăi apelează funcția construct_network(H, W, K). Dacă construct_network nu respectă anumite condiții menționate în enunțul problemei, grader-ul va afișa unul din mesajele de eroare menționat la sfârșitul secțiunii de Implementare, iar apoi iese.

Altfel, grader-ul produce două ieșiri.

În prima fază, grader-ul afișează rezultatul programului robotului în următorul format:

• linia 1+i $(0 \le i)$: ieșirea ultimei instrucțiuni din programul robotului pentru imaginea i (1 sau 0).

În a doua fază, grader-ul scrie în directorul curent un fișier log.txt cu următorul format:

ullet linia $1+i\;(0\leq i)\colon m[0]\;\;m[1]\;\;\dots\;\;m[c-1]$

Secvența de pe linia 1+i descrie valorile stocate în celulele din memoria robotului după ce programul robotului a fost rulat, dându-se imaginea i ca intrare. Mai exact, m[j] reprezintă valoarea celulei j. Remarcați că valoarea lui c (lungimea secvenței) este egală cu $H \cdot W$ plus numărul de instructiuni din programul robotului.