Zpracování obrazu

Vaším úkolem je implementovat program pro robota na zpracování obrazu. Robot snímá kamerou obrázky a ukládá si je do paměti jako černobílé obrazy. Každý obraz má velikost $H \times W$ pixelů, s řádky očíslovanými od 0 do $H-1$ a se sloupci očíslovanými od 0 do $W-1$. V každém obrazu jsou **právě dva** černé pixely, všechny ostatní pixely jsou bílé.

Robot zpracuje každý obraz programem, který je tvořen jednoduchými příkazy. Jsou dány hodnoty H, W a kladné celé číslo K. Vaším úkolem je napsat funkci, která vytvoří program pro robota takový, aby ve zpracovávaném obrazu zjistil, zda je **vzdálenost** mezi dvěma černými pixely rovna přesně K. Vzdálenost mezi pixelem lv řádku r_1 a sloupci c_1 a pixelem v řádku r_2 a sloupci c_2 je definována jako $|r_1 - r_2| + |c_1 - c_2|$. V tomto výrazu $|x|$ značí absolutní hodnotu z x, tzn. $|x|$ je rovno x pro $x \geq 0$ a je rovno $-x$ pro $x < 0$.

Popíšeme si nyní, jak robot pracuje.

Paměť robota je dostatečně velké pole buněk indexované od 0. Každá buňka může obsahovat buď 0 nebo 1 a její hodnota se po nastavení již nemění. Zpracovávaný obraz je v paměti uložen po řádcích v buňkách s indexy 0 až $H \cdot W - 1$. První řádek obrazu je uložen v buňkách 0 až $W-1$, poslední řádek je uložen v buňkách $(H-1) \cdot W$ až $H \cdot W - 1$. Je-li pixel obrazu v řádku i a sloupci j černý, buňka $i \cdot W + j$ má hodnotu 1, v opačném případě má hodnotu 0.

Program robota je posloupnost **příkazů**, které jsou očíslovány po sobě jdoucími celými čísly počínaje od 0. Při výpočtu jsou příkazy programu prováděny jeden za druhým. Každý příkaz přečte hodnoty jedné nebo více buněk (tyto hodnoty nazýváme **vstupy** příkazu) a vytvoří jednu výslednou hodnotu rovnou 0 nebo 1 (nazýváme ji **výstup** příkazu). Výstup příkazu i se uloží do buňky $H \cdot W + i$. Příkaz i smí mít jako své vstupy pouze hodnoty buněk odpovídajích pixelům zpracovávaného obrazu nebo výstupům předchozích příkazů, tzn. buněk s čísly od 0 do $H \cdot W + i - 1$.

Existují čtyři typy příkazů:

- `NOT`: má pouze jeden vstup. Jeho výstupem je 1, pokud vstup je roven 0, v opačném případě je výstupem 0.
- `AND`: má jeden nebo více vstupů. Jeho výstupem je 1, právě když **všechny** jeho vstupy mají hodnotu 1.
- `OR`: má jeden nebo více vstupů. Jeho výstupem je 1, právě když **alespoň jeden** z jeho vstupů má hodnotu 1.

- `XOR`: má jeden nebo více vstupů. Jeho výstupem je 1, právě když **lichý počet** z jeho vstupů má hodnotu 1.

Výstupem posledního příkazu programu bude 1, pokud vzdálenost mezi dvěma černými pixely v obrazu je rovna přesně K, jinak bude výstupem poledního příkazu 0.

Pokyny k implementaci

Implementujte následující funkci:

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

- H, W: rozměry zpracovávaného obrazu
- K: kladné celé číslo
- Tato funkce vytváří program pro robota. Pro každý zpracovávaný obraz bude program robota rozhodovat, zda vzdálenost mezi dvěma černými pixely v obrazu je rovna přesně K.

Vaše funkce bude volat jednou nebo vícekrát následující funkce, aby přidala příkazy do vytvářeného programu pro robota (který je na počátku prázdný):

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

- Přidává příkaz `NOT`, `AND`, `OR`, resp. `XOR`.
- N (pro `add_not`): index buňky, z níž přidaný příkaz `NOT` čte vstup.
- Ns (pro `add_and`, `add_or`, `add_xor`): pole obsahující indexy buněk, z nichž přidaný příkaz `AND`, `OR` nebo `XOR` čte vstup
- Každá funkce vrací index buňky, která obsahuje výstup příslušného příkazu. Postupné volání těchto funkcí vrací posloupnost po sobě jdoucích celých čísel počínaje od $H \cdot W$.

Program robota může obsahovat nejvýše 10 000 příkazů. Příkazy mohou číst dohromady nejvýše 1 000 000 hodnot. Jinými slovy, celková délka polí Ns ve všech voláních funkcí `add_and`, `add_or` a `add_xor` plus počet volání funkce ``add_not``` nesmí překročit 1 000 000.

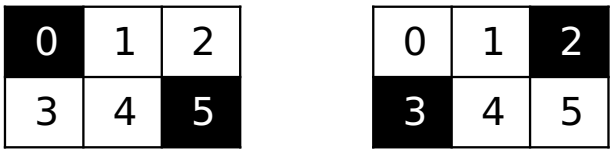
Po přidání posledního příkazu funkce `construct_network` skončí. Vytvořený program pro robota bude následně vyhodnocen na řadě obrazů. Vaše řešení splní úspěšně daná testovací data, pokud pro všechny tyto obrazy bude výstup posledního příkazu správný (tzn. bude 1, právě když vzdálenost dvou černých pixelů v obrazu je rovna K).

Hodnocení vašeho řešení může skončit některou z těchto chybových zpráv:

- `Instruction with no inputs`: vstupem funkce `add_and`, `add_or` nebo `add_xor` je prázdné pole.
- `Invalid index`: vstupem funkce `add_and`, `add_or`, `add_xor` nebo `add_not` je chybný index buňky (možná záporný).
- `Too many instructions`: vaše funkce se pokusila připojit do výsledného programu robota více než $10\,000$ příkazů.
- `Too many inputs`: příkazy čtou dohromady více než $1\,000\,000$ hodnot.

Příklad

Předpokládejme $H = 2$, $W = 3$, $K = 3$. Existují pouze dva obrazy, v nichž je vzdálenost černých pixelů rovna 3.



- Případ 1: černé pixely jsou 0 a 5
- Případ 2: černé pixely jsou 2 a 3

Možným řešením je vytvořit program pro robota pomocí následujících volání:

1. `add_and([0, 5])`, přidává příkaz s výstupem 1, právě když nastane první případ. Výstup se uloží do buňky 6.
2. `add_and([2, 3])`, přidává příkaz s výstupem 1, právě když nastane druhý případ. Výstup se uloží do buňky 7.
3. `add_or([6, 7])`, přidává příkaz s výstupem 1, právě když nastane jeden z výše uvedených případů.

Omezení

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

Podúlohy

1. (10 bodů) $\max(H, W) \leq 3$
2. (11 bodů) $\max(H, W) \leq 10$
3. (11 bodů) $\max(H, W) \leq 30$
4. (15 bodů) $\max(H, W) \leq 100$
5. (12 bodů) $\min(H, W) = 1$

6. (8 bodů) Pixel v řádku 0 a sloupci 0 je černý ve všech obrazech.
7. (14 bodů) $K = 1$
8. (19 bodů) Žádná další omezení.

Ukázkový vyhodnocovač

Ukázkový vyhodnocovač čte vstup v následujíícím formátu:

- řádek 1: $H \;\; W \;\; K$
- řádky $2 + i$ $(i \geq 0)$: $r_1[i] \;\; c_1[i] \;\; r_2[i] \;\; c_2[i]$
- poslední řádek: -1

Každý řádek kromě prvního a posledního představuje obraz se dvěma černými pixely. Obraz popsaný na řádku $2 + i$ označíme jako obraz i. Jeden černý pixel je v řádku r_1 a sloupci c_1, druhý je v řádku r_2 a sloupci c_2.

Ukázkový vyhodnocovač nejprve volá `construct_network(H, W, K)`. Jestliže `construct_network` poruší některé z omezení daných úlohou, ukázkový vyhodnocovač vypíše jedno z chybových hlášení uvedených na konci Pokynů k implementaci a skončí.

Jinak ukázkový vyhodnocovač vytvoří dva výstupy.

Nejprve ukázkový vyhodnocovač vypíše výstup programu robota v následujícím formátu:

- řádek $1 + i$ $(0 \leq i)$: výstup posledního příkazu v programu robota pro obraz i (1 nebo 0).

Poté ukázkový vyhodnocovač vytvoří soubor `log.txt` v aktuálním adresáři:

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

Posloupnost na řádku $1 + i$ popisuje hodnoty v paměti robota po ukončení běhu programu robota pro obraz i na vstupu. Konkrétně, $m[i][j]$ udává hodnotu buňky j. Všimněte si, že hodnota c (délka posloupnosti) se rovná $H \cdot W$ plus počet příkazů v programu robota.