Robot verseny

Az SzTE MI kutatói robotversenyt rendeznek.

Hanga elhatározta, hogy részt vesz a versenyen. A versenynek az a célja, hogy programot kell készíteni a *Pulibot* robot számára. Ezzel is tisztelegve a híres magyar pásztorkutya inteligenciájának.

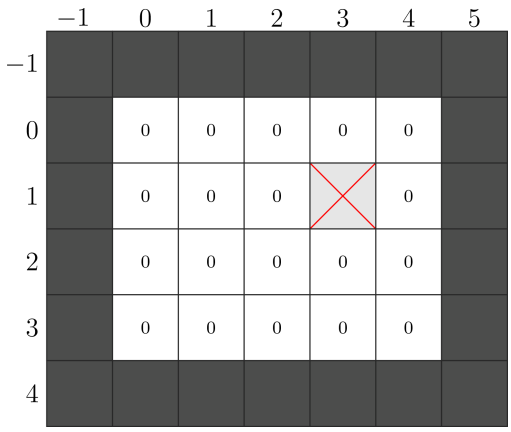
A Pulibot robotot egy $(H + 2) \times (W + 2)$ méretű négyzetrácsos labirintussal tesztelik. A labirintus sorait északról dél felé haladva -1-től H-ig, az oszlopait pedig nyugatról kelet felé haladva -1-től W-ig sorszámozzuk. Az r. sorban és c. oszlopban ($-1 \leq r \leq H$, $-1 \leq c \leq W$) lévő cellára az (r, c) számpárral hivatkozunk.

Tekintsük az (r, c) cellát, ahol $0 \leq r < H$ and $0 \leq c < W$. Az (r, c) cellának 4 **szomszédja** van:

- $(r, c - 1)$ a **nyugati** szomszédja (r, c)-nek;
- $(r + 1, c)$ a **déli** szomszédja (r, c)-nek;
- $(r, c + 1)$ a **keleti** szomszédja (r, c)-nek;
- $(r - 1, c)$ a **északi** szomszédja (r, c)-nek;

Az (r, c) cellát **keretező** cellának nevezünk, ha $r = -1$ vagy $r = H$ vagy $c = -1$ vagy $c = W$ teljesül. Minden, nem keretező cella vagy **akadály** vagy **üres**. Minden üres cellának van egy nemnegatív egész számmal megadott **színe**, amelynek értéke nem nagyobb, mint Z_{MAX}. Kezdetben minden üres cella színe 0.

Tekintsük példaként az alábbi ábrán látható $H = 4$ és $W = 5$ méretű labirintust, amelyben csak egy akadály van a $(1, 3)$ cellában:



Az egyetlen akadály cellát az X kereszt jelöli, a keretezők pedig satírozottak. A cellákban levő számok a cellák színét jelölik. Az (r_0, c_0) cellától az (r_ℓ, c_ℓ) celláig vezető ℓ hosszú **út** egy páronként különböző, üres cellákat tartalmazó $(r_0, c_0), (r_1, c_1), \ldots, (r_\ell, c_\ell)$ sorozat, ahol minden i-re ($0 \leq i < \ell$) az (r_i, c_i) és az (r_{i+1}, c_{i+1}) cellák szomszédosak.

Megjegyzés: minden ℓ hosszú út pontosan $\ell + 1$ cellát tartalmaz.

A versenyen a kutatók olyan labirintust készítenek, amelyben létezik legalább egy út a $(0, 0)$ cellától a $(H - 1, W - 1)$ celláig. A $(0, 0)$ és a $(H - 1, W - 1)$ cellák biztosan üresek.

Hanga nem tudja, hogy hol vannak akadályok, és azt sem tudja, hogy hol vannak az üres cellák. Az a feladatod, hogy segíts Hangának olyan programot készíteni, amely az ismeretlen labirintusban keres egy *legrövidebb utat*, amely a $(0, 0)$ cellától a $(H - 1, W - 1)$ celláig vezet. A Pulibot specifikációja és a verseny szabályzata az alábbi.

Pulibot Specifikáció

Az (r, c) cella ($-1 \leq r \leq H$ és $-1 \leq c \leq W$) **állapota** az alábbiak szerint megadott egész szám:

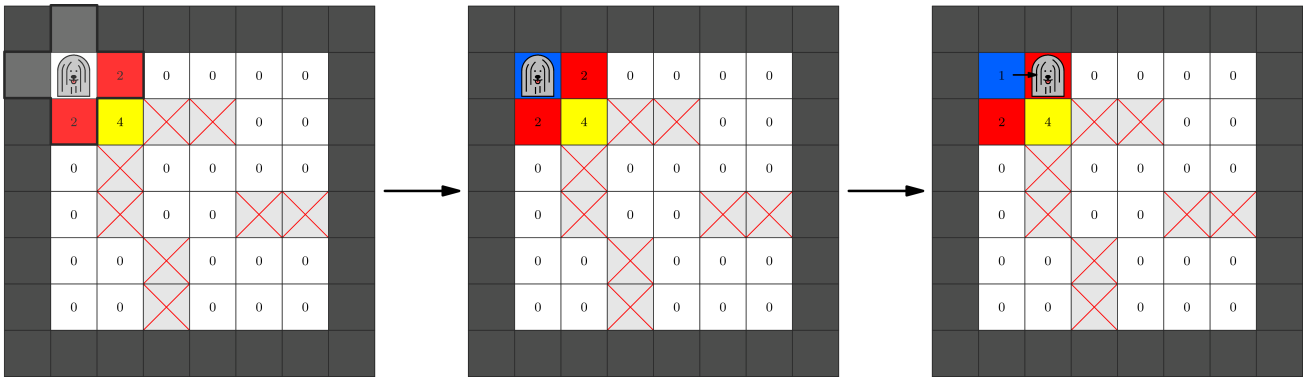
- Ha az (r, c) cella keretező cella, akkor állapota -2;
- Ha az (r, c) cella akadály cella, akkor állapota -1;
- Ha az (r, c) cella üres cella, akkor állapota a cella színének az értéke.

Pulibot programjának végrehajtása az utasításainak egymás után való végrehajtásából áll. Egy utasítás végrehajtása úgy történik, hogy a robot kiolvassa a mostani és a 4 szomszédos cella állapotát, és az ezek által egyértelműen meghatározott műveletet hajtja végre az alábbiak szerint. Tegyük fel, hogy Pulibot az (r, c) üres cellában van. Ekkor a következőt teszi:

1. Először Pulibot kiolvassa saját és a szomszédjainak az állapotat és az S **állapottömbben** tárolja, ahol $S = [S[0], S[1], S[2], S[3], S[4]]$, és az értékek:
    - $S[0]$ az (r, c) cella állapota.
    - $S[1]$ a nyugati szomszédjának állapota.
    - $S[2]$ a déli szomszédjának állapota.
    - $S[3]$ a keleti szomszédjának állapota.
    - $S[4]$ az északi szomszédjának állapota.
2. Ezután Pulibot elkészíti a (Z, A) párral leírható **utasítást**, amelyet az S állapotsor meghatároz.
3. Végül Pulibot végrehajtja az utasítást, ami azt jelenti, hogy az (r, c) cella színét beállítja a Z értékre, majd végrehajtja az A utasítást az alábbiak szerint:
    - *marad* az (r, c) cellában;
    - *átlép* az 4 szomszédos cella valamelyikébe;
    - *befejezi a programot*.

Tekintsük az alábbi ábrán látható elrendezést. Pulibot a $(0, 0)$ cellában van, aminek színe 0. Megállapítja az $S = [0, -2, 2, 2, -2]$ állapotot.

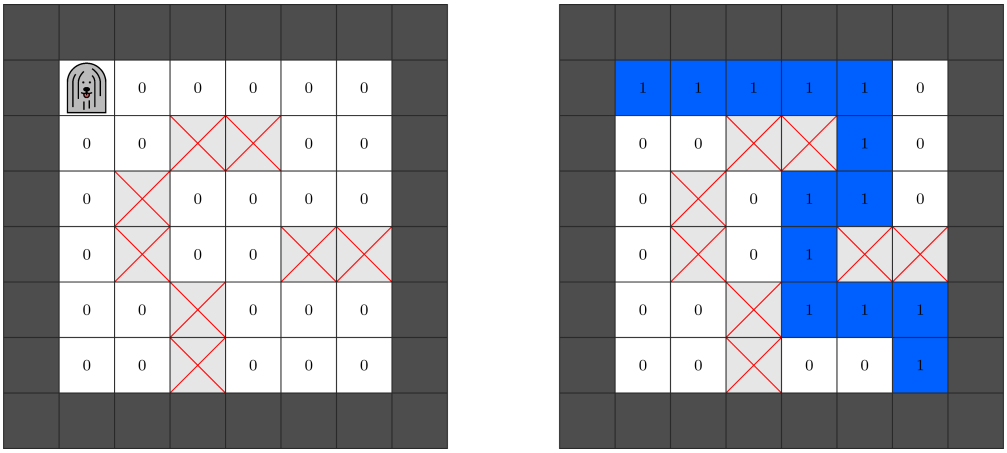
Pulibot programja tartalmaz olyan utasítást, amely az S állapotnak megfelel, és ez az aktuális cella színét $Z = 1$ -re állítja, és keletre lépteti a robotot, amit a középső ábra mutatat.



Robot Versenyszabályzat

- Induláskor Pulibot a $(0, 0)$ cellában van és elkezdi végrehajtani a programját.
- Pulibot nem léphet nem üres cellába.
- Legfeljebb $500\,000$ lépés végrehajtása után be kell fejeznie a programot.
- A program végrehajtása után az üres cellák színeire teljesülni kell az alábbiaknak:
  - Létezik olyan legrövidebb út $(0, 0)$-tól $(H - 1, W - 1)$-ig, amely olyan cellákat tartalmaz, amelyek mindegyikének színe 1.
  - Minden olyan cella színe 0 legyen, amely nem esik a fenti legrövidebb útba.
- Pulibot a program befejezésekor bármely cellában lehet.

Például az alábbi ábra egy lehetséges labirintust mutat, ahol $H = W = 6$. A baloldali ábra a kezdeti helyzetet, a jobboldali pedig egy megfelelő színezést mutat.



Megvalósítás

A következő függvényt kell megvalósítanod!

```
void program_pulibot()
```

- A függvénynek egy Pulibot programot kell készítenie, amely minden olyan H és W értékre helyesen működik, amely kielégíti a feladat feltételeit.
- A függvényt pontosan egyszer hívják minden tesztesetre.

program_pulibot az alábbi fügvényt hívhatja a kívánt program elkészítéséhez:

```
void set_instruction(int[] S, int Z, char A)
```

- S: 5 elemű tömb, amely az állapottömböt adja meg.
- Z: nem negatív egész szám, a szín értéke.
- A: egy karakter, amely a műveletet határozza meg, az alábbiak szerint:
  - `H`: marad;
  - `W`: nyugatra lép;
  - `S`: délre lép;
  - `E`: keletre lép;
  - `N`: északra lép;
  - `T`: befejezi a programot.
- Ennek a függvénynek a meghívása azt jelenti, hogy minden olyan esetben, amikor Pulibot helyzetét az S állapottömb határozza meg, akkor végre kell hajtania az (Z, A) utasítást.

Ha ezt a függvényt többször hívod az S állapotra, akkor `Output isn't correct` hibaüzenetet kapsz.

Nem kötelező minden S állapotra meghívni a `set_instruction` függvényt. Azonban, ha Pulibot olyan állapotba kerül, amelyre a programja nem tartalmaz utasítást, akkor `Output isn't correct` hibát kapsz

Miután `program_pulibot` befejeződik, az értékelő végrahajtja Pulibot programját egy, vagy több labirintusra. Ezen a hívások futási ideje *nem* számít bele a megoldásod futási idejébe. Az értékelő *nem* adaptív, tehát minden labirintus előre meghatározott egy adott tesztesetben.

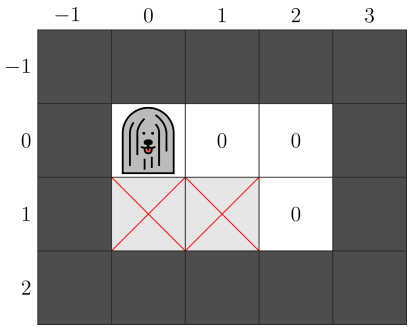
Ha Pulibot megsérti a Versenyszabályzatot, akkor `Output isn't correct` hibaüzenetet kapsz.

Példa

A `program_pulibot` az alábbi `set_instruction` hívásokat végzi:

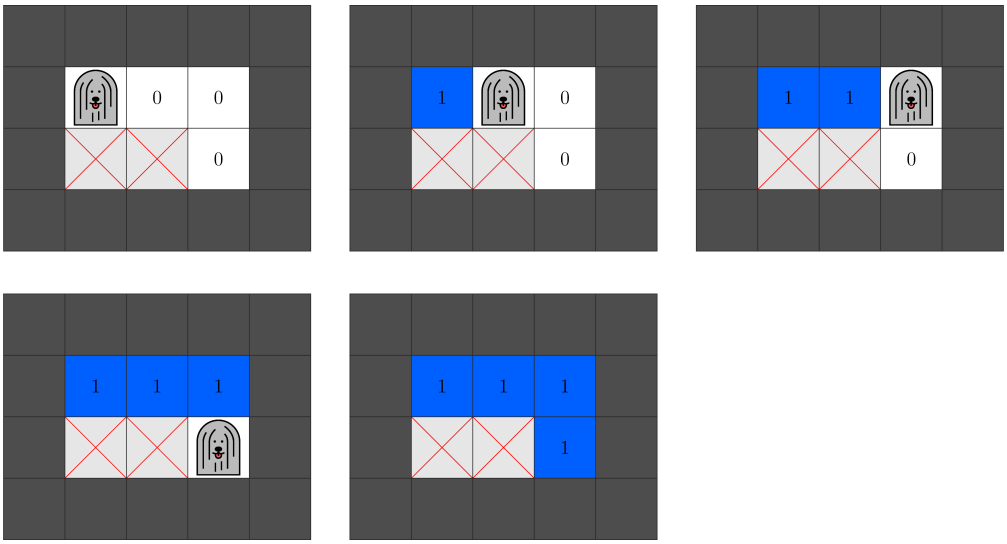
Hívás	Utasítás S-re
`set_instruction([0, -2, -1, 0, -2], 1, E)`	Szín legyen 1 és keletre lép
`set_instruction([0, 1, -1, 0, -2], 1, E)`	Szín legyen 1 és keletre lép
`set_instruction([0, 1, 0, -2, -2], 1, S)`	Szín legyen 1 és délre lép
`set_instruction([0, -1, -2, -2, 1], 1, T)`	Befejez

Tekintsük az alábbi ábrán látható labirintust.



Erre a labirintusra Pulibot programja a táblázatban megadott utasításokat hajtja végre, a sorrendjükben. Az utolsó utasítás hatására befejeződik a program.

A következő négy ábra rendre az utasítások elötti helyzetet mutatja, az utolsó pedig a befejeződés utánit.



Vegyük észre, hogy van olyan labirintus, amelyre ebben a sorrendben végrehajtva az utasításokat, nem eredményez legrövidebb utat, tehát `Output isn't correct` hibaüzenet keletkezik.

Feltételek

$Z_{MAX} = 19$. Tehát Pulibot csak 0 és 19 közötti szín értékeket alkalmaznat .

Labirintus méretek:

- $2 \leq H, W \leq 15$
- Létezik legalább egy út $(0, 0)$-tól $(H-1, W-1)$-ig.

Részfeladatok

1. (6 pont) Nincs akadály a labirintusban.
2. (10 pont) $H = 2$
3. (18 pont) Bármely két üres cella között pontosan egy út van.
4. (20 pont) A $(0, 0)$-tól $(H-1, W-1)$-ig vezető legrövidebb út hossza $H + W - 2$.
5. (46 points) Nincs egyéb feltétel.

Ha bármely tesztesere a `set_instruction` hívása, vagy a Pulibot program végrehajtása nem felel meg a Megvalósításban megadott feltételeknek, akkor 0 pontot kapsz a részfeladatra.

Részpontot is kaphatsz, ha a színezés majdnem helyes.

Formálisan:

- A teszteset megoldása **teljes**, ha a keletkező színezés teljesíti a Robot Versenyszabályzatát.
- A teszteset megoldása **részleges**, ha a keletkező színezés a következően néz ki:
  - Van olyan legrövidebb út $(0, 0)$-tól $(H-1, W-1)$-ig, hogy az útban lévő minden cella színe 1.
  - Nincs olyan nem útban lévő üres cella, amelynek a színe 1.
  - Van olyan üres cella, aminek színe nem 0 és nem 1.

Ha a megoldásod valamely tesztesetre se nem teljes, se nem részleges, akkor 0 pontot kapsz a tesztesetre.

Az 1-4 részfeladatok esetén 50% pontot kapsz, ha a megoldásod részleges.

Az 5. részfeladatban a pontszám attól függ, hogy hány színt használt a Pulibot program. Pontosabban, jelölje $Z^\star$ a `set_instruction` utasításokban használt Z értékek maximumát. A pontszámot az alábbi táblázat szerint kapod:

Feltétel	teljes	részleges
$11 \leq Z^\star \leq 19$	$20 + (19 - Z^\star)$	$12 + (19 - Z^\star)$
$Z^\star = 10$	31	23
$Z^\star = 9$	34	26
$Z^\star = 8$	38	29
$Z^\star = 7$	42	32
$Z^\star \leq 6$	46	36

A részfeladatra kapott pont a tesztesetekre kapott pontszámok minimuma.

Mintaértékelő

A mintaértékelő az alábbi formában olvassa a bemenetet:

- 1. sor: H W
- $2+r$ ($0 \leq r < H$). sor: $m[r][0]$ $m[r][1]$ $\ldots$ $m[r][W-1]$

Itt m olyan H elemű tömb, amelynek minden eleme egy W egész számot tartalmazó tömb, amely a labirintust írja le.
$m[r][c] = 0$ha az (r, c) cella üres és $m[r][c] = 1$ ha az (r, c) cella akadály.

A mintaértékelő elöször hívja a `program_pulibot()` függvényt. Ha protokol sértést detektál, akkor kiírja a standard kimenetre, hogy `Protocol Violation: <MSG>` , ahol `<MSG>` a következö hibaüzenet lehet:

- `Invalid array`: $-2 \leq S[i] \leq Z_{MAX}$ nem teljesül valamely i-re, vagy S hossza nem 5.
- `Invalid color`: $0 \leq Z \leq Z_{MAX}$ nem teljesül.
- `Invalid action`: az A karakter nem `H`, `W`, `S`, `E`, `N` vagy `T`.
- `Same state array`: `set_instruction` ismételten lett híva az S értékre.

A `program_pulibot` szabályos befejezése után a meghívja a Pulibot programot a mintában megadott labirintusra. A mintaértékelő két kimenetet készít. Az aktuális könyvtárban a `robot.bin` nevű fájlba írja ki a végrahajtott műveletek naplózását. Ez a fájl a következő fejezetben ismertetett vizualizáló alkalmazés számára kell. Ha a Pulibot program nem szabályosan fejeződik be. akkor az alábbi hibaüzeneteket írhatja ki:

- `Unexpected state`: Pulibot olyan állapotba kerül, amelyre nincs `set_instruction` álta készített utasítás.
- `Invalid move`: Pulibot nem üres cellába lépne.
- `Too many steps`: $500\,000$ lépésre sem fejeződik be a program.

Egyébként, legyen $e[r][c]$ az (r, c) cella állapota a Pulibot program befejeződésekor. A mintaértékelő H sort ír ki az alábbi formában:

- $1+r$ ($0 \leq r < H$). sor: $e[r][0]$ $e[r][1]$ $\ldots$ $e[r][W-1]$

Vizualizáló alkalmazás

A melléklet tartalmaz egy `display.py` programot. Ezt végrehajtva megjeleníti Pulibot műveleteit. Ehhez szükséges, hogy az adott könyvtárban ott legyen a `robot.bin` bináris fájl.

A vizualizálót így kell végrehajtani:

```
python3 display.py
```

Egy egyszerű grafikus felület jelenik meg, amely az alábbiakat mutatja.

- Láthatod a labirintus állapotát. A Polibot aktuális helyzetét fényes téglalap jelöli.
- Végrehajthatod a lépéseket a nyílra kattintva, vagy megnyomva a megfelelő billentyűt. Ugorhatsz adott lépésre is.
- Alul mutatja a következő lépést. Mutatja az aktuális állapottömböt és a megfelelő utasítást. Az utolsó lépés után vagy a hibaüzenetet adja, vagy a `Terminated` üzenetet, ha a program sikeresen befejeződött.
- Minden színt reprezentáló számhoz rendelhetsz vizuális háttérszínt, és szín megnevezést is. Ezt a hozzárendelést az alábbiak szerint teheted:
  - A `Colors` gombra kattintva, a megjelenő ablakban add meg az értéket.
  - Szerkesztheted a `colors.txt` fájlt.
- A `robot.bin` fájlt a `Reload` gombbal tudod újraolvasni, ha az közben változott.