Състезание за роботи

Изследователи на ИИ (изкуствен интелект) в университета на Сегед организират състезание за програмиране на роботи. Вашият приятел, Ханга, е решил да се включи в това състезание. Целта е да се програмира най-добрият *Пулибот* в чест на завидния интелект на (не)известната порода унгарски кучета Пули.

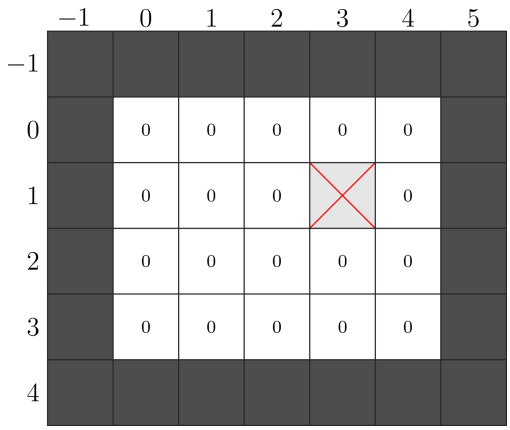
Пулиботът ще бъде тестван на лабиринт, който представлява таблица с размери $(H + 2) \times (W + 2)$. Редовете на таблицата са номерирани с числата от -1 до H в посока от север към юг, а колоните на таблицата са номерирани с числата от -1 до W от запад към изток. Когато говорим за клетката на ред r и колона c на таблицата ($-1 \leq r \leq H$, $-1 \leq c \leq W$), ще използваме означението (r, c).

Нека разгледаме клетката (r, c), за която $0 \leq r < H$ и $0 \leq c < W$. Тя има 4 **съседни** клетки:

- клетка $(r, c - 1)$ ще наричаме **на запад** от клетка (r, c);
- клетка $(r + 1, c)$ ще наричаме **на юг** от клетка (r, c);
- клетка $(r, c + 1)$ ще наричаме **на изток** от клетка (r, c);
- клетка $(r - 1, c)$ ще наричаме **на север** от клетка (r, c).

Клетка (r, c) наричаме **гранична** клетка за лабиринта, ако $r = -1$ или $r = H$, или $c = -1$ или $c = W$. Всяка клетка, която не е гранична клетка за лабиринта, е или клетка с **препятствие**, или **празна** клетка. Допълнително, всяка празна клетка има **цвят**, който означаваме с неотрицателно цяло число между 0 и Z_{MAX}, включително. В началото, цветът на всяка празна клетка е 0.

Нека разгледаме за пример лабиринт, за който $H = 4$ и $W = 5$, съдържащ единствено препятствие в клетка $(1, 3)$:



Единствената клетка с препятствие е означена с кръстче. Граничните клетки са затъмнени. Числата във всяка празна клетка са съответните цветове.

Път с дължина ℓ ($\ell > 0$) от клетка (r_0, c_0) до клетка (r_ℓ, c_ℓ) е редица от различни *празни* клетки $(r_0, c_0), (r_1, c_1), \ldots, (r_\ell, c_\ell)$, така че за всяко i ($0 \leq i < \ell$) клетките (r_i, c_i) и (r_{i+1}, c_{i+1}) са съседни.

Обърнете внимание, че път с дължниа ℓ се състои от точно $\ell + 1$ клетки.

По време на състезанието, изследователите са подготвили лабиринт, в който има поне един път от клетка $(0, 0)$ до клетка $(H - 1, W - 1)$. Това означава и, че клетките $(0, 0)$ и $(H - 1, W - 1)$ са със сигурност празни.

Ханга не знае кои клетки на лабиринта са празни и в кои има препятствия.

Вашата задача е да помогнете на Ханга да програмира Пулибота, така че да е възможно да намери *най-кратък път* (такъв с минимална дължина) от клетка $(0, 0)$ до клетка $(H - 1, W - 1)$ в лабиринта, подготвен от изследователите. Спецификациите на Пулибота и правилата на състезанието са описани по-долу.

Също отбелязваме, че последната секция в условието, описва инструмент за визуализация, който може да използвате за визуализацията на Пулибота.

Спецификация на Пулибота

Нека дефинираме **състоянието** на клетка (r, c) за всяко $-1 \leq r \leq H$ и $-1 \leq c \leq W$ като цяло число, така че:

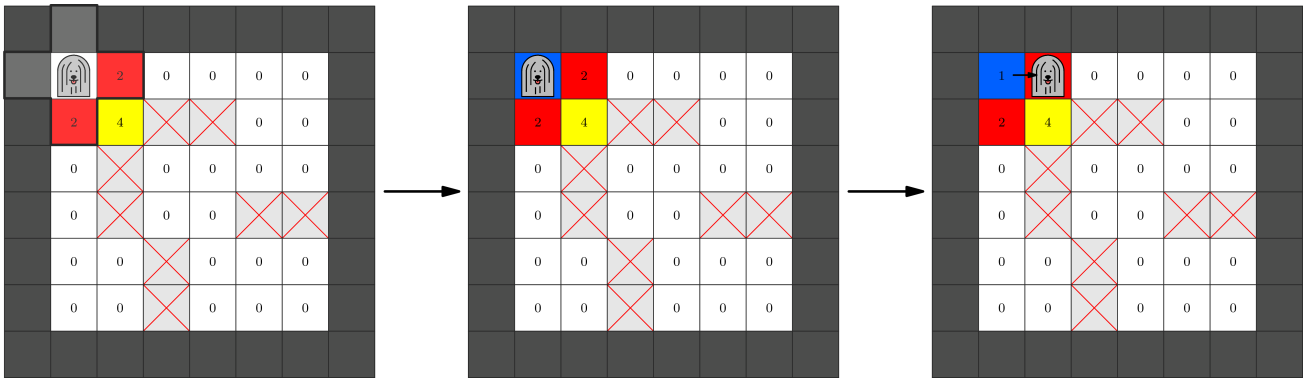
- ако клетката (r, c) е гранична клетка, то състоянието е -2;
- ако клетката (r, c) е клетка с препятствие, то състоянието е -1;
- ако клетката (r, c) е празна, то състоянието е цвета на клетката.

Програмата на Пулибота се изпълнява като поредица от стъпки. На всяка стъпка, Пулибота разпознава състоянията на близките клетки и изпълнява инструкция. Инструкцията, която изпълнява се определя от разпознатите състояния. Следва по-подробно описание.

Нека да предположим, че в началото на текущата стъпка, Пулиботът е в клетка (r, c), която е празна. Стъпката протича по следния начин:

1. Първо, Пулиботът разпознава **масив на състоянията**, което е масивът $S = [S[0], S[1], S[2], S[3], S[4]]$, съставен от състоянието на клетка (r, c) и състоянията на всички съседни клетки:
   - $S[0]$ е състоянието на клетка (r, c).
   - $S[1]$ е състоянието на клетката на запад.
   - $S[2]$ е състоянието на клетката на юг.
   - $S[3]$ е състоянието на клетката на изток.
   - $S[4]$ е състоянието на клетката на север.
2. След което, Пулиботът определя **инструкция** от вида (Z, A), която е в резултат на разпознатия масив на състоянията.
3. Последно, Пулиботът изпълнява определената инструкция: слага цвета на клетка (r, c) на цвят Z и изпълнява действие A, което е едно от следните:
   - *оставане* в клетка (r, c);
   - *придвижване* към една от 4-те съседни клетки;
   - *приключване на програмата*.

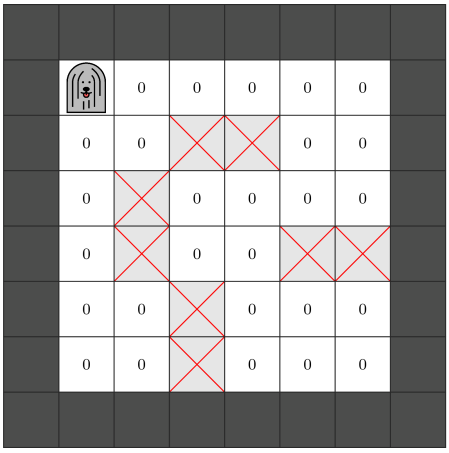
Например, нека да разгледаме случая, който е показан в лявата част на следващата фигура. Пулиботът в момента е в клетка $(0, 0)$ с цвят 0. Той разпознава масива на състоянията $S = [0, -2, 2, 2, -2]$. Пулиботът може да има програма, в която при разпознаване на този масив, слага цвета на текущата клетка на $Z = 1$ и се придвижва на изток, както е показано в средната и дясната част на фигурата:

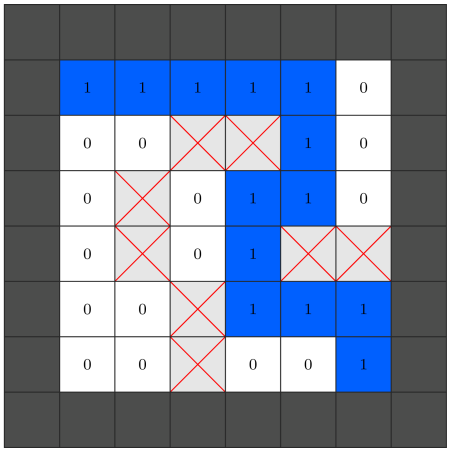


Правила на състезанието за роботи

- В началото, Пулиботът е поставен в клетка $(0, 0)$ и започва изпълнение на програмата си.
- Не е позволено Пулиботът да се движи към клетка, която не е празна.
- Програмата на Пулибота трябва да приключи след най-много $500\,000$ стъпки.
- След приключването на програмата на Пулибота, празните клетки в лабиринта трябва да бъдат оцветени по следния начин:
  - Трябва да има най-кратък път от клетка $(0, 0)$ до клетка $(H - 1, W - 1)$, за който цветът на всяка клетка по пътя е 1.
  - Цветът на всички други празни клетки е 0.
- Пулиботът може да приключи програмата си във всяка празна клетка.

Например, следната фигура показва възможен лабиринт с $H = W = 6$. Началната конфигурация е показана в лявата част, а едно вярно оцветяване на празни клетки след приключване е показано в дясната част:





Детайли по имплементацията

Трябва да напишете следната функция.

```
void program_pulibot()
```

- Тази функция конструира програмата на Пулибота. Тази програма трябва да работи вярно за всички възможни стойности на H и W и всеки възможен лабиринт, който спазва ограниченията на задачата.
- Тази функция се извиква точно веднъж за всеки тест.

Тази функция може да прави извиквания към следната функция, за да конструира програмата на Пулибота:

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

- S: масив с големина 5, описващ масив на състоянията.
- Z: неотрицателно цяло число, което задава цвят.
- A: символ, който задава действие на Пулибота, както следва:
  - `H`: оставане;
  - `W`: преместване на запад;
  - `S`: преместване на юг;
  - `E`: преместване на изток;
  - `N`: преместване на север;
  - `T`: приключване на програмата.
- Извикването на тази функция задава на Пулибота, че когато разпознае масива на състоянията S, трябва да изпълни инструкцията (Z, A).

Ако извикате многократно тази функция за един и същ масив на състоянията S, то ще получите резултат `Output isn't correct`.

Не е задължително да извикате функцията `set_instruction` с всеки възможен масив на състоянията S. Но ако се случи така, че Пулибота разпознае масив на състоянията, за който няма инструкция, то също ще получите резултат `Output isn't correct`.

След като функцията `program_pulibot` завърши, грейдърът изпълнява програмата на Пулибота върху един или повече лабиринти. Тези изпълнения *не се* броят към времето на изпълнение на вашето решение. Грейдърът *не е* адаптивен, така че лабиринтите са предварително фиксирани за всеки тест.

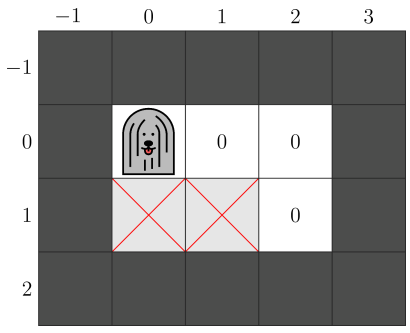
Ако Пулиботът не спазва някое от правилата на състезанието за роботи в тази секция преди приключване на програмата си, то ще получите резултат `Output isn't correct`.

Пример

Функцията `program_pulibot` може да направи следните извиквания на `set_instruction`:

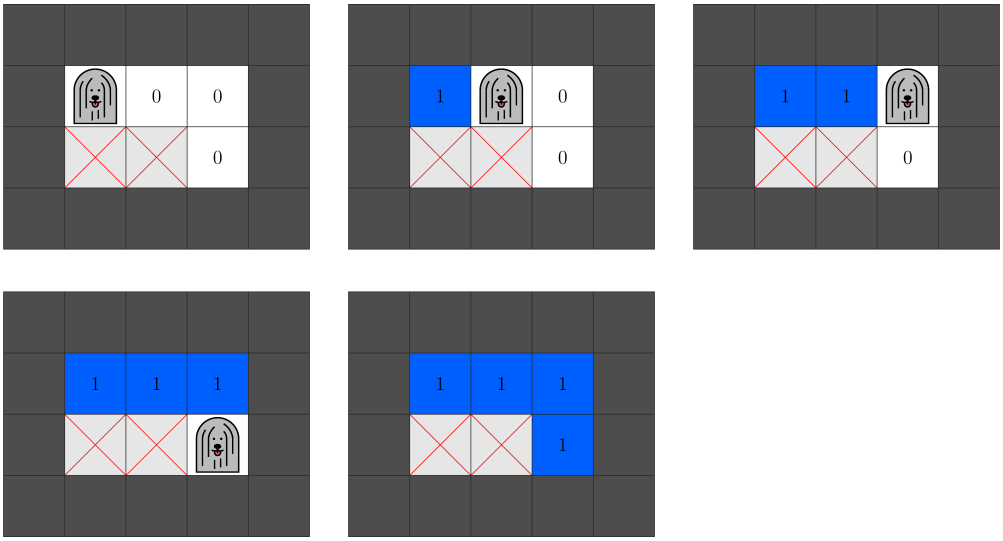
Извикване	Инструкция за масива на състоянията S
`set_instruction([0, -2, -1, 0, -2], 1, E)`	Задаване на цвета на 1 и преместване на изток
`set_instruction([0, 1, -1, 0, -2], 1, E)`	Задаване на цвета на 1 и преместване на изток
`set_instruction([0, 1, 0, -2, -2], 1, S)`	Задаване на цвета на 1 и преместване на юг
`set_instruction([0, -1, -2, -2, 1], 1, T)`	Задаване на цвета на 1 и приключване на програмата

Нека имаме следния случай, където $H = 2$ и $W = 3$, и лабиринтът е показан на следната фигура.



За този конкретен лабиринт програмата на Пулибота работи за четири стъпки. Разпознатите масиви на състоянията и инструкциите, които се изпълняват, съответстват точно на четирите извиквания на `set_instruction`, направени по-горе, в същия ред. Последната инструкция приключва програмата.

Следните фигури показват лабиринта преди всяка от четирите стъпки и цветовете след приключването.



Имайте предвид, че тази програма от 4 инструкции може да не намери най-краткия път в други валидни лабиринти. Затова, ако се събмитне, ще получи резултат `Output isn't correct`.

Ограничения

$Z_{MAX} = 19$. Това означава, че Пулиботът може да използва цветовете от 0 до 19, включително.

За всеки лабиринт, на който се тества Пулибота:

- $2 \leq H, W \leq 15$
- Има поне един път от клетка $(0, 0)$ до клетка $(H - 1, W - 1)$.

Подзадачи и оценяване

1. (6 точки) Няма клетка с препятствие в лабиринта.
2. (10 точки) $H = 2$
3. (18 точки) Има точно един път между всеки две празни клетки.
4. (20 точки) Всеки най-кратък път от клетка $(0, 0)$ до клетка $(H - 1, W - 1)$ има дължина $H + W - 2$.
5. (46 точки) Няма допълнителни ограничения.

Ако за някой тест, извикванията към функцията `set_instruction` нарушават някое от описаните ограничения, или при изпълнението на програмата на Пулибота, не се спазят описаните ограничения, то Вашето решение ще получи 0 точки за съответната подзадача.

На всяка подзадача може да изкарате частичен резултат, ако направите оцветяване, което е почти вярно.

Формално:

- Решението на даден тест е **пълно**, ако финалното оцветяване на празните клетки спазва правилата на състезанието за роботи.
- Решението на даден тест е **частично**, ако финалното оцветяване изглежда по следния начин:
  - Има най-кратък път от $(0,0)$ до $(H-1,W-1)$, за който цвета на всяка клетка, включена в пътя е 1.
  - Няма друга празна клетка в табллицата с цвят 1.
  - Някои празни клетки в таблицата имат цвят различен от 0 и 1.

Ако вашето решение на даден тест не е нито пълно, нито частично, то ще получите 0 за теста.

В подзадачи 1-4, резултатът за пълно решение е 100%, а резултатът за частично решение на даден тест е 50% от точките на подзадачата, от която е част този тест.

В подзадача 5, вашият резултат зависи от броя използвани цветове, използвани в програмата на Пулибота. По-точно, нека означим с $Z^\star$ максималната стойност на Z измежду всички извиквания на `set_instruction`. Резултатът на теста се изчислява спрямо таблицата:

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

Резултатът на всяка подзадача е минималният брой точки измежду тестовете на подзадачата.

Локално тестване

Локалният грейдър чете входа в следния формат:

- ред 1: H W
- ред $2 + r$ ($0 \le r < H$): $m[r][0]$ $m[r][1]$ $\ldots$ $m[r][W-1]$

Тук, m е масив от H масива с по W цели числа, описващ неграничните клетки на лабиринта. $m[r][c] = 0$, ако клетката (r, c) е празна, и $m[r][c] = 1$, ако клетката (r, c) е клетка с препятствие.

Локалният грейдър първо извиква `program_pulibot()`. Ако той засече нарушение на протокола, то ще се отпечата `Protocol Violation: <MSG>` и ще приключи работа, като `<MSG>` е едно от следните съобщение за грешка:

- `Invalid array`: $-2 \le S[i] \le Z_{MAX}$ не е спазено за някое i или дължината на S не е 5.
- `Invalid color`: $0 \le Z \le Z_{MAX}$ не е спазено.
- `Invalid action`: символът A не е някой от `H`, `W`, `S`, `E`, `N` или `T`.
- `Same state array`: `set_instruction` е било извикано с един и същ масив на състоянията S повече от веднъж.

В противен случай, когато функцията `program_pulibot` завърши, локалният грейдър изпълнява програмата на Пулибота на лабиринта, описан на входа.

Локалният грейдър има два изхода.

Първо, локалният грейдър отпечатва лог на действията на Пулибота във файла `robot.bin` във вашата работна директория. Файлът служи за вход на инструмента за визуализация, описан в следващата секция.

Второ, ако програмата на Пулибота не приключи успешно, то локалният грейдър отпечатва някое от следните съобщения:

- `Unexpected state`: Пулиботът разпознава масив на състоянията, за който функцията `set_instruction` не е била извикана.
- `Invalid move`: изпълняване на някое действие, премества Пулибота в непразна клетка.
- `Too many steps`: Пулиботът изпълнява $500\,000$ стъпки без да приключи програмата си.

В противен случай, нека $e[r][c]$ е състоянието на клетката (r, c) след като програмата на Пулибота приключи. Локалният грейдър отпечатва H реда в следния формат:

- Ред $1 + r$ ($0 \le r < H$): $e[r][0]$ $e[r][1]$ $\ldots$ $e[r][W-1]$

Инструмент за визуализация

Прикаченият архив на задачата съдържа файл, който се казва `display.py`. Когато този файл се изпълни, Python скрипта визуализира действията на Пулибота в лабиринта, както е описано във файла, генериран от локалния грейдър. За тази цел, трябва двоичния файл `robot.bin` да е в работната директория, в която е и скрипта.

За да пуснете скрипта, изпълнете следната команда.

```
python3 display.py
```

Ще се покаже прост графичен интерфейс. Основните характеристики са следните:

- Може да виждате състоянието на целия лабиринт. Текущата позиция на Пулибота е показана чрез правоъгълник.
- Може да се движите между различните стъпки на Пулибота, като натискате бутоните със стрелки или съответните бутони на клавиатурата. Също така, може да отидете на конкретна стъпка.
- Следващата стъпка на програмата на Пулибота се показва най-отдолу. Там е показан текущият масив на състоянията и инструкцията, която ще се изпълни. След последната стъпка това, което се показва, е някое от съобщенията за грешка на грейдъра или `Terminated`, ако програмата е завършила успешно.
- За всяко число, което съответства на цвят, може да зададете съответен фонов цвят на визуализация, както и текст за визуализация. Текстът за визуализация е кратък низ, който ще се появи на всяка клетка от съответния цвят. Може да зададете фоновите цветове и текстовете за визуализация по някой от следните начини:
  - Задавате ги в диалоговия прозорец, който се отваря след натискане на бутона `Colors`.
  - Редактирате съдържанието на файла `colors.txt`.
- За да презаредите файла `robot.bin`, използвайте бутона `Reload`. Това е нужно, ако съдържанието на `robot.bin` е променено.