Robot Contest

Szeged 大學的 AI 研究者們正在舉辦一個機器人程式比賽。你的朋友 Hanga 決定參賽。目標是要設計一隻叫 Pulibot 的強大機器人，取名的緣由來自於知名的匈牙利放牧犬 Puli。

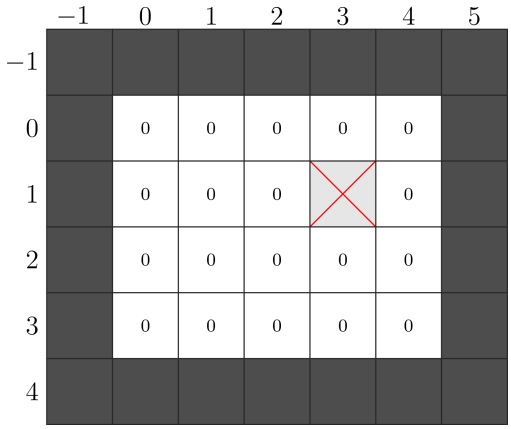
Pulibot 的測試環境是一個由 $(H+2) \times (W+2)$ 個方格組成的矩陣迷宮。矩陣的列 (rows) 標號由北至南依序標上 -1 至 H，矩陣的行 (columns) 標號由西至東依序標上 -1 至 W。在第 r 列 ($-1 \leq r \leq H$) 和第 c 行 ($-1 \leq c \leq W$) 的格子叫做方格 (r, c)。

對於方格 (r, c) 而言 ($0 \leq r < H, 0 \leq c < W$)，有四個**相鄰**方格：

- 方格 $(r, c-1)$ 稱為方格 (r, c) 的**西鄰居**;
- 方格 $(r+1, c)$ 稱為方格 (r, c) 的**南鄰居**;
- 方格 $(r, c+1)$ 稱為方格 (r, c) 的**東鄰居**;
- 方格 $(r-1, c)$ 稱為方格 (r, c) 的**北鄰居**。

如果矩陣迷宮內的方格 (r, c) 滿足 $r = -1$ 或 $r = H$ 或 $c = -1$ 或 $c = W$ 四個其中之一的條件，那麼方格 (r, c) 被稱為**邊界方格**。矩陣迷宮內其他不是邊界方格的方格，只有下列兩種可能：(1) **障礙方格** (2) **空白方格**。每一個空白方格會被塗上顏色。顏色用一個介於 0 和 Z_{MAX} (包含 0 和 Z_{MAX})之間的非負整數表示。每個空白方格的顏色一開始全初始化成 0。

舉例來說，下面這個矩陣迷宮的 $H = 4$，$W = 5$，且包含一個障礙方格 $(1, 3)$。



這個唯一的障礙方格在圖裡被打上一個叉。邊界方格全被塗黑。每一個空白方格上有一個數字，這個數字就代表這個方格的顏色。

一條從方格 (r_0, c_0) 至方格 (r_ℓ, c_ℓ) 且長度為 ℓ ($\ell > 0$) 的**路徑**，可以表示成一串空白方格的位置 $(r_0, c_0), (r_1, c_1), \ldots, (r_\ell, c_\ell)$，其中每一對空白方格 (r_i, c_i)、空白方格 (r_{i+1}, c_{i+1}) 必須是相鄰 ($0 \leq i < \ell$)。

留意一條長度為 ℓ 的路徑，由恰好 $\ell + 1$ 個方格組成。

在比賽中，研究者們會建構一個矩陣迷宮。在迷宮中，至少存在一條路徑從方格 $(0, 0)$ 至方格 $(H - 1, W - 1)$。留意這意謂著方格 $(0, 0)$ 和方格 $(H - 1, W - 1)$ 保證是空白方格。

Hanga 並不知道矩陣迷宮中的哪些方格是空白方格、哪些是障礙方格。

你的任務是幫助 Hanga 設計 Pulibot 程式，使 Pulibot 能夠在一個未知的矩陣迷宮中找到任何一條從方格 $(0, 0)$ 至方格 $(H - 1, W - 1)$的最短長度路徑。Pulibot 的使用規範和比賽規則如下所述。

留意這題目文件的最後一段會介紹一個工具讓你用來視覺化 Pulibot。

Pulibot 使用規範 (Pulibot's Specification)

定義方格 (r, c) 的**狀態** ($-1 \leq r \leq H, -1 \leq c \leq W$) 為一個整數如下：

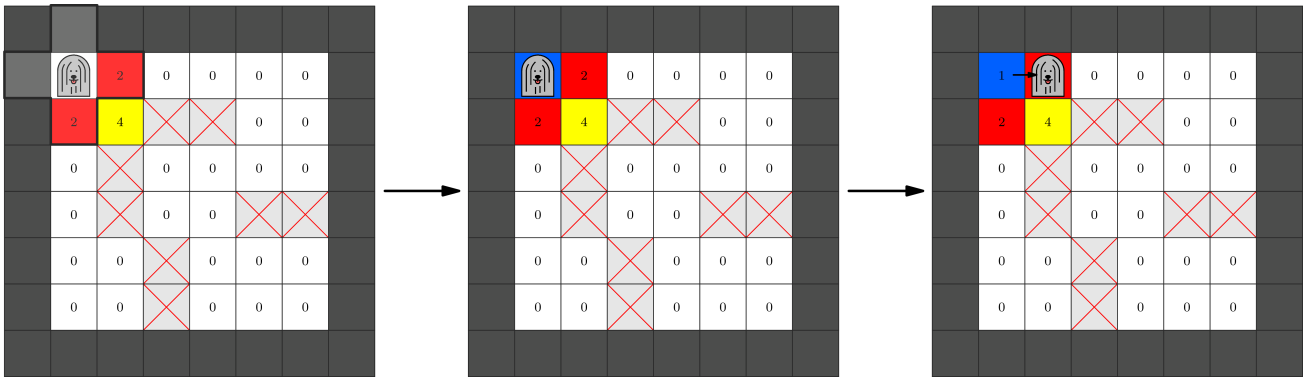
- 如果方格 (r, c) 是邊界方格，那麼它的狀態為 -2;
- 如果方格 (r, c) 是障礙方格，那麼它的狀態為 -1;
- 如果方格 (r, c) 是空白方格，那麼它的狀態為它的顏色號碼;

Pulibot 被執行時，會完成一連串的步驟。在每一步，Pulibot 可以讀相鄰方格的狀態並完成一個指令。這個指令會由讀到的狀態決定。以下是更詳盡的說明。

假設在某一步開始時，Pulibot 站在方格 (r, c) (留意它是一個空白方格)，這一步會這樣完成：

1. 首先 Pulibot 讀取目前所在方格 (r, c) 的**狀態串**，這狀態串是一個陣列 $S = [S[0], S[1], S[2], S[3], S[4]]$，包含方格 (r, c) 的狀態、其他四格相鄰方格的狀態：
    - $S[0]$ 表示方格 (r, c) 的狀態。
    - $S[1]$ 表示方格 (r, c) 西鄰居的狀態。
    - $S[2]$ 表示方格 (r, c) 南鄰居的狀態。
    - $S[3]$ 表示方格 (r, c) 東鄰居的狀態。
    - $S[4]$ 表示方格 (r, c) 北鄰居的狀態。
2. 接著 Pulibot 會根據讀到的狀態串，對應到一個**指令** (Z, A)。
3. 最後 Pulibot 會執行這個指令：把方格 (r, c) 塗成顏色 Z、然後做動作 A，A為下面其中一個：
    - *stay* 待在方格(r, c)；
    - *move* 移動至四個相鄰方格的其中一格；
    - *terminate* 終止整個 Pulibot 程式。

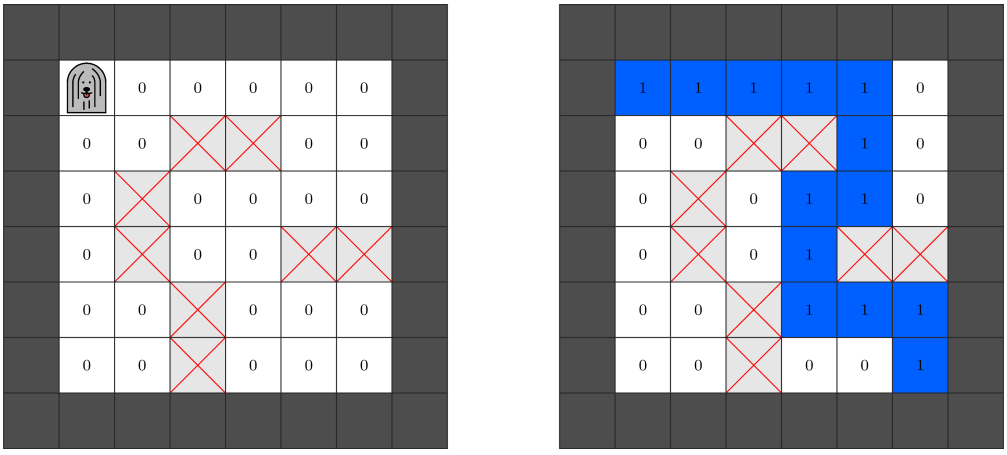
舉例來說，考慮下圖左邊的情境。Pulibot 目前待在方格 $(0, 0)$，方格的顏色是 0。Pulibot 讀取狀態串得到 $S = [0, -2, 2, 2, -2]$。根據讀到的狀態串所對應的指令，Pulibot 將目前所在方格塗成顏色 $Z = 1$、然後移動至東鄰居。就如下圖中間和下圖右邊所示。



比賽規則 (Robot Contest Rules)

- 一開始 Pulibot 被放在方格 $(0, 0)$ 然後啟動 Pulibot。
- Pulibot 禁止移動到不是空白方格的方格。
- Pulibot 必須要終止程式，在移動至多 $500\,000$ 步之後。
- 當 Pulibot 程式終止時，矩陣迷宮中的空白方格必須要如下塗色
  - 存在某一條從方格 $(0, 0)$ 至方格 $(H - 1, W - 1)$的最短長度路徑，這條路徑上的每一個方格都被塗成顏色 1。
  - 其他不在這條路徑上的空白方格都被塗成顏色 0。
- Pulibot 終止時，可以停在任何一個空白方格。

  舉例而言，下圖是一個矩陣迷宮，其 $H = W = 6$。Pulibot 一開始啟動時，如下圖左；Pulibot 終止時，把空白方格塗成下圖右的顏色是一個可接受的答案。



實作細節 (Implementation Details)

你應該實作以下程序。

```
void program_pulibot()
```

- 這程序要用來實現 Pulibot。對於所有的 H 和 W，對於所有滿足條件的矩陣迷宮，Pulibot 都需要正確地執行。
- 對於每一筆測試資料，這程序恰好會被呼叫一次。

這程序可以呼叫以下程序數次來實現 Pulibot。

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

- S： 長度為 5 的陣列，用以描述狀態串。
- Z： 一個非負整數，表示顏色。
- A：一個字元，表示 Pulibot 的動作，如下：
  - `H`：待在原地；
  - `W`：移動至西鄰居；
  - `S`：移動至南鄰居；
  - `E`：移動至東鄰居；
  - `N`：移動至北鄰居；
  - `T`：終止程式。
- 透過呼叫這個程序，教導 Pulibot 以下這件事：當它讀到狀態串 S 時，它需要執行指令 (Z, A)。

如果呼叫這個程序時，同樣的狀態串 S 被設定不只一次會被判成 `Output isn't correct`。

沒有必要對每一個狀態串 S 都呼叫 `set_instruction` 一次。但是，如果 Pulibot 讀到一個沒有被 `set_instruction` 設定過的狀態串，會判成 `Output isn't correct`。

當 `program_pulibot` 結束設定後，評分程式會呼叫 Pulibot 跑在一個或多個矩陣迷宮上。呼叫 Pulibot 的時間不會算在你的解答時限內。評分程式是固定的，也就是說每一筆測試資料所用到的矩陣迷宮集合在每一筆測試資料是固定的。

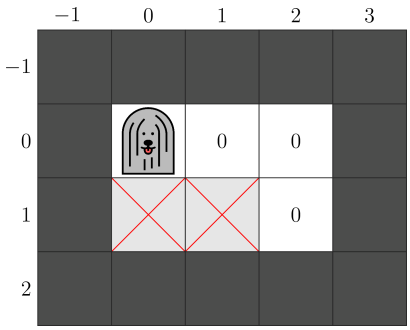
如果 Pulibot 在終止之前，違反任何競賽規則，你會收到判決 `Output isn't correct`。

例子 (Example)

程序 `program_pulibot` 可以呼叫 `set_instruction` 數次，如下：

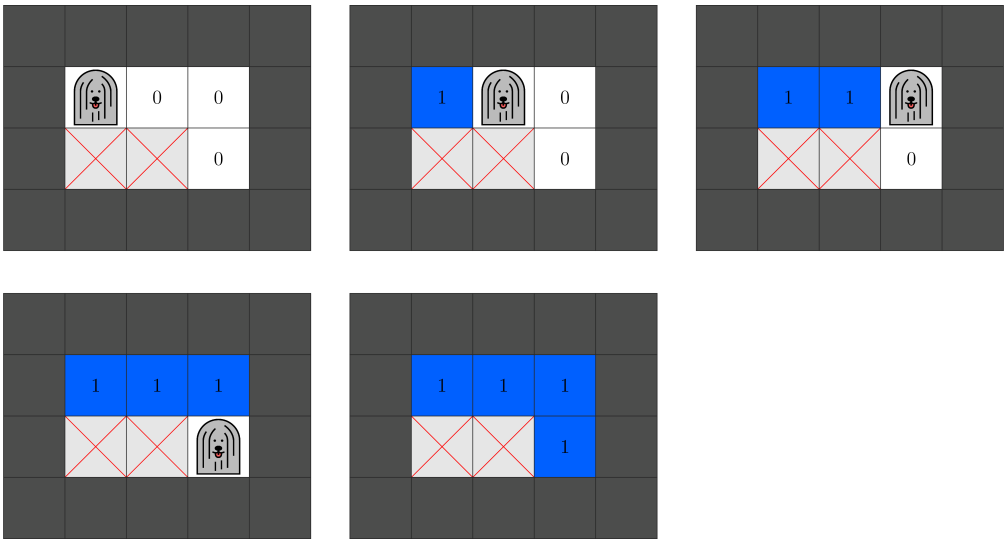
Call	Instruction for state array S
`set_instruction([0, -2, -1, 0, -2], 1, E)`	Set color to 1 and move east
`set_instruction([0, 1, -1, 0, -2], 1, E)`	Set color to 1 and move east
`set_instruction([0, 1, 0, -2, -2], 1, S)`	Set color to 1 and move south
`set_instruction([0, -1, -2, -2, 1], 1, T)`	Set color to 1 and terminate program

考慮下圖中的情境，其 $H = 2$ 且 $W = 3$。



在上面這個矩陣迷宮，Pulibot 會走四步。Pulibot 讀到的狀態串、Pulibot 根據讀到的狀態串所執行的指令，如上表所示。這些指令的最後一個會終止 Pulibot。

在這四步的每一步之前的矩陣迷宮狀態、Pulibot 終止後的狀態，一一表示在下圖。



然後，如果換成其他的矩陣迷宮，使用上面四個指令的 Pulibot 不見得可以找到符合要求的某一條最短長度路徑。因此，直接使用這範例指令，會收到判決 `Output isn't correct`。

限制 (Constraints)

$Z_{MAX} = 19$. 因此，Polibot 只能使用 20 種顏色，從 0 至 19。

每一個用來測試 Pulibot 的矩陣迷宮，滿足下列所有條件：

- $2 \leq H, W \leq 15$
- 至少有一條路徑從方格$(0, 0)$ 至方格 $(H - 1, W - 1)$。

子任務 (Subtasks)

1. (6 points) 矩陣迷宮內沒有障礙方格。
2. (10 points) $H = 2$。
3. (18 points) 任兩個空白方格，恰好有一條路徑連接。
4. (20 points) 任何從方格 $(0, 0)$ 至方格 $(H - 1, W - 1)$ 的最短路徑的長度為 $H + W - 2$。
5. (46 points) 沒有額外限制。

如果子任務中有任何一筆測試資料，呼叫 `set_instruction` 時或是 Pulibot 執行時，沒有遵守實作細節的規範，這子任務的得分為 0。

在每一個子任務，如果 Pulibot 所塗的顏色差不多是正確的你可以得到部分分數。

精確來說：

- 如果最終的塗色滿足競賽規則，那麼這一筆測資的解就是**complete**的。
- 如果最終的塗色滿足以下條件，那麼這一測資的解就是**partial**的：
  - 存在某條從方格$(0, 0)$ 至方格 $(H-1, W-1)$的最短路徑，上面的每個方格都被塗成顏色 1。
  - 其他任何空白方格都不是顏色 1。
  - 某些空白方格被塗成 0 和 1 以外的顏色。

如果有任何一筆測資，你的解既不是complete也不是partial，那麼那筆測資的得分為 0。

在子任務 $1-4$，每一筆測資，如果輸出 complete solution ，該筆測資得分 100%。每一筆測資，如果輸出 partial solution，該筆測資得分 50%。

在子任務 5，你的分數取決於 Pulibot 使用的顏色數量。精確地說，令 $Z^\star$ 為在呼叫 `set_instruction` 時，曾經用過的最大 Z 值。 每一筆測資的分數按照下表計算：

Condition	Score (complete)	Score (partial)
$11 \le Z^\star \le 19$	$20 + (19 - Z^\star)$	$12 + (19 - Z^\star)$
$Z^\star = 10$	31	23
$Z^\star = 9$	34	26
$Z^\star = 8$	38	29
$Z^\star = 7$	42	32
$Z^\star \le 6$	46	36

每一子任務的得分為，該子任務中的所有測資得分的最低分。

(範例評分程式) Sample Grader

範例評分程式的輸入為下：

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

這裡，m 是一個陣列，其擁有 H 個擁有 W 個整數的陣列。m 描述了矩陣迷宮中的非邊界方格。$m[r][c] = 0$ 如果方格 (r, c) 是空白方格，$m[r][c] = 1$ 如果 (r, c) 是障礙方格。

範例評分程式首先會呼叫 `program_pulibot()`。 如果範例評分程式偵測到任何違反規範的行為，它會印出 `Protocol Violation: <MSG>`然後終結程式。其中 `<MSG>`會是下面錯誤訊息的其中一種：

- `Invalid array`: 存在某些 i，$-2 \leq S[i] \leq Z_{MAX}$ 不滿足，或S的長度不是5。
- `Invalid color`: $0 \leq Z \leq Z_{MAX}$ 不滿足。
- `Invalid action`: 字元 A 不是 `H`, `W`, `S`, `E`, `N` 或 `T`。
- `Same state array`: 對於同一個 S ，`set_instruction`被呼叫至少兩次。

否則，當 `program_pulibot` 結束時，範例評分程式會將 Pulibot 跑在輸入的矩陣迷宮。

範例評分程式有兩個輸出。

第一個，範例評分程式會將 Pulibot 的動作紀錄輸出在一個叫 `robot.bin` 的檔案，放在工作目錄下。這個檔案可以被拿來當成視覺化工具的輸入。視覺化工具會在下一段介紹。

第二個，如果 Pulibot 沒有成功結束，那麼範例評分程式會印出以下錯誤訊息的其中之一個：

- `Unexpected state`: Pulibot 讀到一個狀態串，它沒有在 `set_instruction` 呼叫中設定動作。
- `Invalid move`: Pulibot 移動到一個非空白格。
- `Too many steps`: Pulibot 跑了 $500\,000$ 步卻沒有停下來。

否則，令$e[r][c]$為方格(r, c)在 Pulibot 結束時的狀態。範例評分程式會按照以下格式印出 H 行：

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

視覺化工具 (Display Tool)

這個題目的附件包包含下面這個檔案： `display.py`. 如果執行它，它會顯示 Pulibot 的動作在範例評分程式所使用的輸入矩陣迷宮。為了做這件事，`robot.bin`這個二進位檔一定要放在工作目錄底下。

跑下面這行執行這個 python 腳本

```
python3 display.py
```

一個簡單的圖形化介面會跳出來，主要的功能如下：

- 你可以看到整個矩陣迷宮的狀態。Pulibot 目前的位置會用一個矩形突顯。
- 你可以透過點擊方向鍵或他們的熱鍵，一步一步地重現 Pulibot 的動作。你也可以直接跳到某一步。
- 即將出現的下一步，會出現在下方。他會顯示目前的狀態串以及這狀態串所對應的動作指令。到了最後一步，他會顯示其中一個的錯誤訊息或是顯示`Terminated`如果程式正常結束。
- 對於每一個顏色數字，你可以設定一個背景色給他以及一個顯示文字。這個顯示文字是一個短字串，會被寫入每一個同顏色的格子。你可以用下面其中一個方式完成設定：
  - 在對話窗中透過點擊`Colors`按鈕設定。
  - 編輯`colors.txt`檔案內的內容。

- 要重新載入robot.bin，請使用Reload按鈕。當robot.bin的內容被改過，請使用這個按鈕。