แข่งหุ่นยนต์

นักวิจัยด้าน AI ของมหาวิทยาลัยแซแก็ดได้จัดการแข่งขันเขียนโปรแกรมหุ่นยนต์ขึ้น เพื่อนของคุณชื่อฮังก้าได้เข้าร่วมการแข่งขันนี้ โดยเป้าหมายของการแข่งขันคือการเขียนโปรแกรมให้กับ *พูลิบอท* เพื่อเชิดชูความฉลาดของสุนัขต้อนแกะพันธุ์พูลิของฮังการี

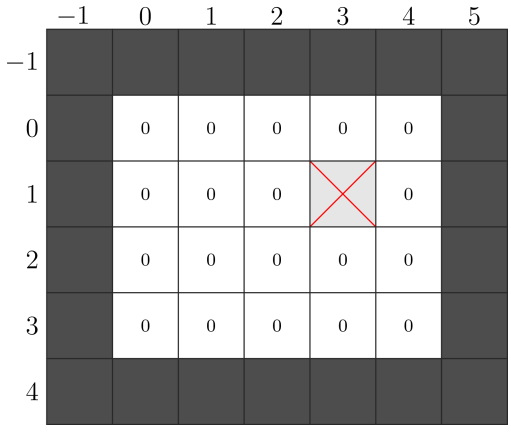
เราจะทดสอบการทำงานของพูลิบอทบนตารางเขาวงกตที่ประกอบด้วยช่องจำนวน $(H+2) \times (W+2)$ ช่อง แถวของตารางนี้กำหนดด้วยหมายเลขตั้งแต่ -1 ถึง H จากบนลงล่าง และคอลัมน์ของตารางนี้กำหนดด้วยหมายเลข -1 ถึง W จากซ้ายไปขวา เราเรียกช่องที่อยู่ ณ แถว r และ คอลัมน์ c ของตาราง ($-1 \le r \le H, -1 \le c \le W$) ว่าช่อง (r, c).

สำหรับช่อง (r, c) ที่ $0 \le r < H$ และ $0 \le c < W$ กำหนดให้มีช่อง 4 ช่องที่ **อยู่ติดกัน** กับช่อง (r, c) ได้แก่

- ช่อง $(r, c-1)$ ซึ่งเรียกว่าช่องด้าน **ซ้าย** ของช่อง (r, c);
- ช่อง $(r+1, c)$ ซึ่งเรียกว่าช่องด้าน **ล่าง** ของช่อง (r, c);
- ช่อง $(r, c+1)$ ซึ่งเรียกว่าช่องด้าน **ขวา** ของช่อง (r, c);
- ช่อง $(r-1, c)$ ซึ่งเรียกว่าช่องด้าน **บน** ของช่อง (r, c).

ช่อง (r, c) เป็น ช่อง**ขอบ** ของเขาวงกตนี้ถ้า $r = -1$ หรือ $r = H$ หรือ $c = -1$ หรือ $c = W$ เป็นจริง แต่ละช่องที่ไม่ใช่ช่องขอบจะเป็น ช่อง**สิ่งกีดขวาง** หรือ ช่อง**ว่าง** นอกจากนี้ช่องว่างแต่ละช่องจะมี **สี** กำกับอยู่ ซึ่งถูกระบุได้ด้วยจำนวนเต็มไม่ลบตั้งแต่ 0 ถึง Z_{MAX} รวมหัวท้าย โดยในตอนเริ่มต้น สีของช่องว่างแต่ละช่องจะเป็น 0

พิจารณาตัวอย่างเขาวงกตขนาด $H = 4$ และ $W = 5$ ซึ่งมีช่องสิ่งกีดขวางหนึ่งช่องคือช่อง $(1, 3)$



ช่องสิ่งกีดขวางหนึ่งช่องนั้นคือช่องที่มีเครื่องหมายกากบาท ช่องขอบของเขาวงกตคือช่องที่ระบายสีเข้มไว้ ตัวเลขที่เขียนอยู่ในช่องว่างแต่ละช่องคือสีของช่องเหล่านั้น

เส้นทาง ความยาว ℓ ($\ell > 0$) จากช่อง (r_0, c_0) ไปยังช่อง (r_ℓ, c_ℓ) คือลำดับของช่อง*ว่าง*ที่แตกต่างกัน $(r_0, c_0), (r_1, c_1), \ldots, (r_\ell, c_\ell)$ โดยที่ช่อง (r_i, c_i) และ (r_{i+1}, c_{i+1}) นั้นเป็นช่องที่อยู่ติดกันสำหรับแต่ละค่า i (

$0 \le i < \ell$)

สังเกตว่าเส้นทางความยาว ℓ มีช่องจำนวน $\ell + 1$ ช่องพอดี

ในการแข่งขันนั้น นักวิจัยจะสร้างเขาวงกตให้มีอย่างน้อยหนึ่งเส้นทางจากจากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$ ซึ่งหมายความว่าช่อง $(0, 0)$ และ $(H - 1, W - 1)$ จะต้องเป็นช่องว่างแน่นอน

ฮังก้าไม่รู้ว่าช่องใดในเขาวงกตเป็นช่องว่างและช่องใดเป็นช่องสิ่งกีดขวาง

งานของคุณคือช่วยฮังก้าเขียนโปรแกรมให้กับพูลิบอทให้มีความสามารถในการหา *เส้นทางสั้นสุด* (ซึ่งคือเส้นทางที่มีความยาวน้อยที่สุด) จากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$ ในเขาวงกตที่ไม่ทราบข้อมูลที่ถูกสร้างขึ้นโดยนักวิจัย ข้อกำหนดของของพูลิบอทและกฎการแข่งขันเป็นดังต่อไปนี้

สังเกตว่า ส่วนสุดท้ายของคำอธิบายโจทย์นี้ จะอธิบายเครื่องมือสำหรับแสดงภาพการทำงานของพูลิบอท

ข้อกำหนดของของพูลิบอท

กำหนดให้**สถานะ**ของช่อง (r, c) สำหรับ $-1 \le r \le H$ และ $-1 \le c \le W$ เป็นค่าจำนวนเต็มดังต่อไปนี้

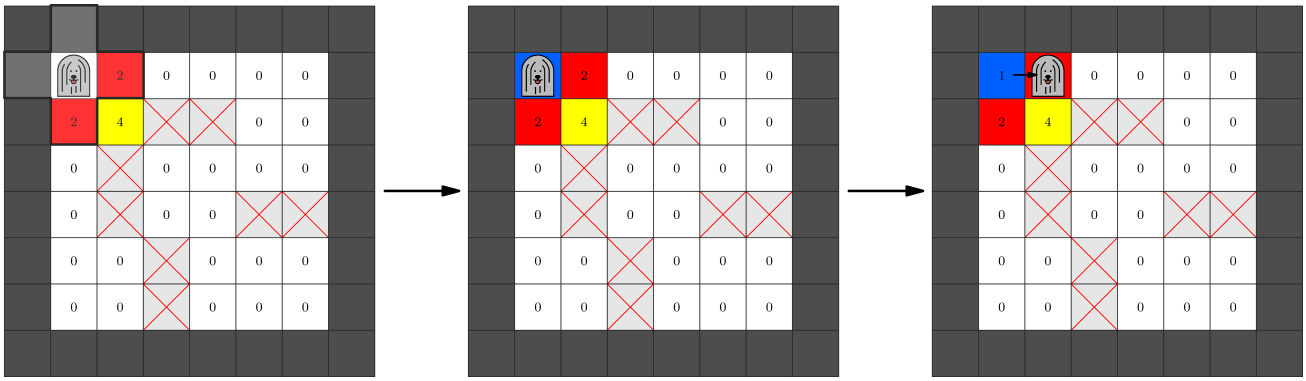
- ถ้าช่อง (r, c) เป็นช่องขอบ ให้สถานะของช่องเป็น -2
- ถ้าช่อง (r, c) เป็นช่องสิ่งกีดขวาง ให้สถานะของช่องเป็น -1
- ถ้าช่อง (r, c) เป็นช่องว่าง ให้สถานะของช่องเป็นสีของช่องนั้น

โปรแกรมของพูลิบอทจะทำงานเป็นขั้นตอนตามลำดับ ในแต่ละขั้นตอนพูลิบอทจะอ่านค่าสถานะของช่องที่อยู่ติดกันและทำงานตามคำสั่ง โดยคำสั่งนั้นจะขึ้นอยู่กับสถานะที่พูลิบอทอ่านค่ามาได้ คำอธิบายโดยละเอียดเป็นดังต่อไปนี้

สมมติให้ ณ ก่อนการทำงานในขั้นตอนปัจจุบันนั้น พูลิบอทอยู่ที่ช่อง (r, c) ซึ่งเป็นช่องว่าง ขั้นตอนการทำงานจะเป็นดังนี้

1. ขั้นแรกพูลิบอทจะอ่านค่า **อาร์เรย์สถานะ** ซึ่งคืออาร์เรย์ $S = [S[0], S[1], S[2], S[3], S[4]]$ ที่ประกอบด้วยสถานะของช่อง (r, c) และช่องที่อยู่ติดกัน
    - $S[0]$ คือสถานะของช่อง (r, c)
    - $S[1]$ คือสถานะของช่องด้านซ้าย
    - $S[2]$ คือสถานะของช่องด้านล่าง
    - $S[3]$ คือสถานะของช่องด้านขวา
    - $S[4]$ คือสถานะของช่องด้านบน
2. หลังจากนั้นพูลิบอทจะเลือก **คำสั่ง** (Z, A) ตามค่าของอาร์เรย์สถานะที่อ่านค่ามาได้
3. สุดท้ายนี้ พูลิบอทจะทำงานตามคำสั่งดังกล่าว โดยตั้งค่าสีของช่อง (r, c) ให้เป็น Z และทำงาน A ซึ่งเป็นการทำงานอย่างใดอย่างหนึ่งต่อไปนี้
    - *อยู่กับที่* ณ ช่อง (r, c)
    - *เดิน* ไปยังช่องใดช่องหนึ่งจาก 4 ช่องที่อยู่ติดกัน
    - *จบการทำงาน*

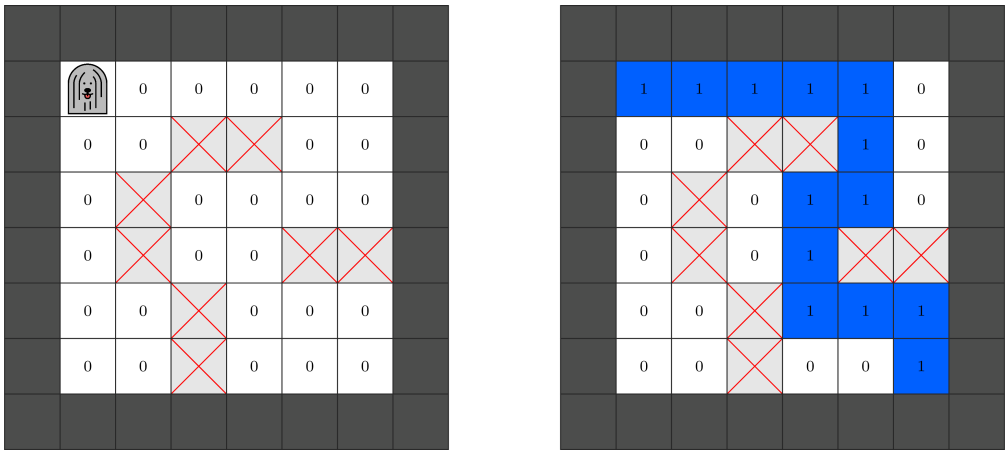
ให้พิจารณาสถานการณ์ที่แสดงในด้านซ้ายของรูปต่อไปนี้เป็นตัวอย่าง พูลิบอทอยู่ ณ ช่อง $(0, 0)$ ซึ่งมีสีเป็น 0 พูลิบอทอ่านค่าอาร์เรย์สถานะได้เป็น $S = [0, -2, 2, 2, -2]$ พูลิบอทอาจถูกตั้งโปรแกรมให้เมื่อรับรู้อาร์เรย์นี้แล้ว ทำการตั้งค่าสีของช่องปัจจุบันให้เป็น $Z = 1$ และเดินไปทางขวา ดังแสดงในรูปกลางและรูปขวาในรูปต่อไปนี้



กฎการแข่งขันหุ่นยนต์

- ตอนเริ่มต้นพูลิบอทจะอยู่ที่ช่อง $(0, 0)$ และเริ่มทำงานตามโปรแกรม
- พูลิบอทไม่สามารถเดินไปยังช่องที่ไม่ใช่ช่องว่างได้
- โปรแกรมของพูลิบอทจะต้องจบการทำงานหลังจากทำงานไปมากสุด $500\,000$ ขั้นตอน
- หลังจากโปรแกรมของพูลิบอทจบการทำงานแล้ว ช่องว่างในเขาวงกตจะต้องมีสีตามรูปแบบต่อไปนี้
  - มีเส้นทางสั้นสุดเส้นทางหนึ่งจากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$ ที่สีของแต่ละช่องในเส้นทางนี้เป็น 1
  - ช่องว่างอื่น ๆ ในตารางมีสีเป็น 0
- พูลิบอทสามารถจบการทำงาน ณ ช่องว่างใด ๆ ก็ได้

รูปต่อไปนี้แสดงตัวอย่างเขาวงกตที่เป็นไปได้เมื่อ $H = W = 6$ โดยที่รูปแบบตอนเริ่มต้นเป็นดังรูปซ้ายและการตั้งค่าสีที่ถูกต้องแบบหนึ่งหลังจบการทำงานเป็นดังรูปขวา



รายละเอียดการเขียนโปรแกรม

คุณต้องเขียนฟังก์ชันต่อไปนี้

```
void program_pulibot()
```

- ฟังก์ชันนี้จะต้องสร้างโปรแกรมของพูลิบอท โปรแกรมดังกล่าวจะต้องทำงานได้อย่างถูกต้องสำหรับค่า H และ W ใด ๆ และสำหรับเขาวงกตใด ๆ ที่ตรงกับเงื่อนไขของโจทย์
- ฟังก์ชันนี้จะถูกเรียกหนึ่งครั้งสำหรับแต่ละข้อมูลทดสอบ

ฟังก์ชันนี้สามารถเรียกใช้ฟังก์ชันต่อไปนี้เพื่อสร้างโปรแกรมของพูลิบอท

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

- S: อาร์เรย์ความยาว 5 ที่ระบุถึงอาร์เรย์สถานะ
- Z: จำนวนเต็มไม่ลบที่ระบุถึงสี
- A: อักขระหนึ่งตัวที่ระบุถึงการทำงานของพูลิบอทดังต่อไปนี้
  - `H`: อยู่กับที่
  - `W`: เดินไปด้านซ้าย
  - `S`: เดินไปด้านล่าง
  - `E`: เดินไปด้านขวา
  - `N`: เดินไปด้านบน
  - `T`: จบการทำงาน
- การเรียกฟังก์ชันนี้เป็นการสร้างคำสั่งให้กับพูลิบอทโดยบอกว่าเมื่ออ่านค่าอาร์เรย์สถานะ S พูลิบอทจะต้องทำงานตามคำสั่ง (Z, A)

การเรียกใช้ฟังก์ชันนี้หลายครั้งโดยที่มีค่าอาร์เรย์สถานะ S เดียวกันจะทำให้ได้ผลการตรวจเป็น `Output isn't correct`

เราไม่จำเป็นต้องเรียก `set_instruction` ด้วยค่าอาร์เรย์สถานะ S ทั้งหมดที่เป็นไปได้ทุกค่าก็ได้ อย่างไรก็ตาม ถ้าพูลิบอทอ่านค่าอาร์เรย์สถานะที่ไม่ได้มีการกำหนดคำสั่งไว้ คุณจะได้ผลการตรวจเป็น `Output isn't correct`

หลังจากฟังก์ชัน `program_pulibot` จบการทำงาน เกรดเดอร์จะเรียกใช้โปรแกรมของพูลิบอทกับเขาวงกตอย่างน้อยหนึ่งเขาวงกต การเรียกใช้งานโปรแกรมของพูลิบอทนี้จะ*ไม่*ถูกนับเป็นเวลาในการทำงานของคำตอบของคุณ เกรดเดอร์จะ*ไม่*ปรับตัว กล่าวคือ เซตของเขาวงกตที่ใช้จะถูกกำหนดไว้ก่อนแล้วในแต่ละข้อมูลทดสอบ

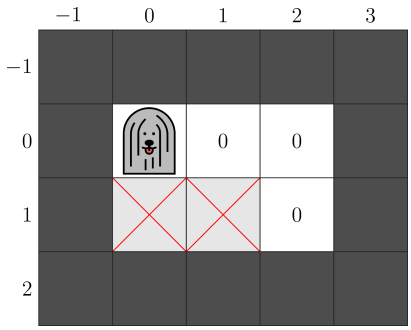
ถ้าหากพูลิบอทละเมิดกฎการแข่งขันหุ่นยนต์ข้อใดก่อนจบการทำงาน คุณจะได้ผลการตรวจเป็น `Output isn't correct`

ตัวอย่าง

ฟังก์ชัน `program_pulibot` สามารถเรียก `set_instruction` ดังต่อไปนี้

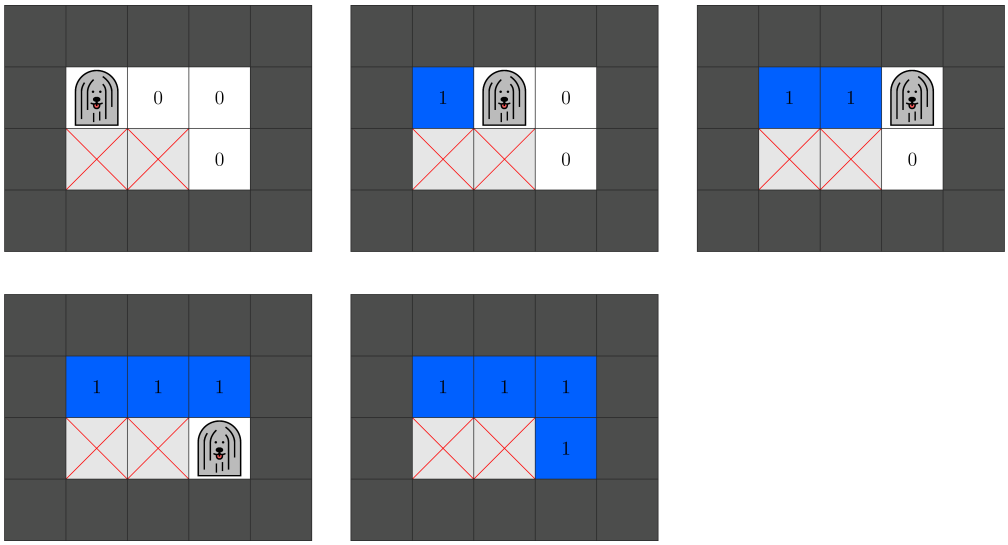
การเรียกใช้	คำสั่งสำหรับอาร์เรย์สถานะ S
`set_instruction([0, -2, -1, 0, -2], 1, E)`	ตั้งค่าสีเป็น 1 แล้วเดินไปด้านขวา
`set_instruction([0, 1, -1, 0, -2], 1, E)`	ตั้งค่าสีเป็น 1 แล้วเดินไปด้านขวา
`set_instruction([0, 1, 0, -2, -2], 1, S)`	ตั้งค่าสีเป็น 1 แล้วเดินไปด้านล่าง
`set_instruction([0, -1, -2, -2, 1], 1, T)`	ตั้งค่าสีเป็น 1 แล้วจบการทำงาน

พิจารณาสถานการณ์ที่ $H = 2$ และ $W = 3$ และเขาวงกตเป็นดังรูปด้านล่างนี้



สำหรับเขาวงกตนี้โปรแกรมของพูลิบอทจะทำงานสี่ขั้นตอน อาร์เรย์สถานะที่พูลิบอทอ่านค่าได้และคำสั่งที่จะทำจะตรงกับการเรียก `set_instruction` ทั้งสี่ครั้งข้างต้นตามลำดับ คำสั่งสุดท้ายจะจบการทำงานของโปรแกรม

รูปต่อไปนี้แสดงเขาวงกต ณ ก่อนการทำงานในแต่ละขั้นและสีสุดท้ายหลังจบการทำงาน



อย่างไรก็ตาม ให้สังเกตว่าโปรแกรมที่มี 4 คำสั่งนี้อาจจะไม่สามารถหาเส้นทางสั้นสุดในเขาวงกตอื่น ๆ ที่เป็นไปได้ก็ได้ ดังนั้นหากส่งโปรแกรมโปรแกรมข้างต้นมาก็จะได้ผลการตรวจเป็น `Output isn't correct`

ข้อจำกัด

$Z_{MAX} = 19$ ดังนั้นพูลิบอทจะใช้สีได้ตั้งแต่ 0 ถึง 19 รวมหัวท้าย

สำหรับแต่ละเขาวงกตที่ใช้ทดสอบพูลิบอท

- $2 \leq H, W \leq 15$
- มีอย่างน้อย 1 เส้นทางจากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$.

ปัญหาย่อย

1. (6 คะแนน) ไม่มีช่องสิ่งกีดขวางในเขาวงกต
2. (10 คะแนน) $H = 2$
3. (18 คะแนน) มีเส้นทางเพียงหนึ่งเส้นทางระหว่างแต่ละคู่ของช่องว่าง
4. (20 คะแนน) เส้นทางสั้นสุดจากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$ มีความยาวเป็น $H + W - 2$
5. (46 คะแนน) ไม่มีข้อจำกัดอื่นใด

ในข้อมูลทดสอบใด ๆ หากการเรียกฟังก์ชัน `set_instruction` หรือหากระหว่างการทำงานของโปรแกรมของพูลิบอทไม่เป็นไปตามข้อกำหนดในส่วน "รายละเอียดการเขียนโปรแกรม" คะแนนของคำตอบของคุณสำหรับปัญหาย่อยดังกล่าวจะเป็น 0

ในแต่ละปัญหาย่อย คุณสามารถได้คะแนนบางส่วน โดยการตั้งค่าสีที่เกือบจะถูกต้อง

กล่าวคือ

- คำตอบสำหรับข้อมูลทดสอบจะ **สมบูรณ์** (complete) ถ้าการตั้งค่าสีของช่องว่างเป็นไปตามกฎการแข่งขันหุ่นยนต์
- คำตอบสำหรับข้อมูลทดสอบจะ **ถูกบางส่วน** (partial) ถ้าการตั้งค่าสีเป็นดังต่อไปนี้
  - มีเส้นทางสั้นสุดเส้นทางหนึ่งจากช่อง $(0, 0)$ ไปยังช่อง $(H - 1, W - 1)$ ที่สีของแต่ละช่องในเส้นทางนี้เป็น 1
  - ไม่มีช่องว่างอื่นใดในตารางที่มีสีเป็น 1
  - ช่องว่างบางช่องในตารางมีสีอื่นที่ไม่ใช่ 0 และไม่ใช่ 1

ถ้าคำตอบของคุณสำหรับข้อมูลทดสอบหนึ่งไม่สมบูรณ์และไม่ใช่ถูกบางส่วน คะแนนของคุณสำหรับข้อมูลทดสอบนั้นจะเป็น 0

ในปัญหาย่อย 1-4 คะแนนของคำตอบที่สมบูรณ์จะเป็น 100% และ คะแนนของคำตอบที่ถูกบางส่วนจะเป็น 50% ของคะแนนของปัญหาย่อยนั้น

ในปัญหาย่อยที่ 5 คะแนนของคุณจะขึ้นอยู่กับจำนวนสีที่ใช้ในโปรแกรมของพูลิบอท กล่าวคือ ให้ $Z^\star$ คือค่ามากสุดของ Z จากการเรียกใช้ `set_instruction` ทั้งหมด คะแนนของข้อมูลทดสอบจะถูกคำนวณตามตารางด้านล่างนี้

เงื่อนไข	คะแนน (สมบูรณ์)	คะแนน (ถูกบางส่วน)
$11 \le Z^\star \le 19$	$20 + (19 - Z^\star)$	$12 + (19 - Z^\star)$
$Z^\star = 10$	31	23
$Z^\star = 9$	34	26
$Z^\star = 8$	38	29
$Z^\star = 7$	42	32
$Z^\star \le 6$	46	36

คะแนนของแต่ละปัญหาย่อยคือคะแนนน้อยที่สุดของแต่ละข้อมูลทดสอบในปัญหาย่อยนั้น

เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างอ่านข้อมูลนำเข้าในรูปแบบต่อไปนี้

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

ในที่นี้ ให้ m คืออาร์เรย์ความยาว H ของอาร์เรย์ความยาว W ของจำนวนเต็ม ซึ่งระบุถึงช่องที่ไม่ใช่ช่องขอบของเขาวงกต โดย $m[r][c] = 0$ ถ้าหากช่อง (r, c) เป็นช่องว่าง และ $m[r][c] = 1$ ถ้าหากช่อง (r, c) เป็นช่องสิ่งกีดขวาง

เกรดเดอร์ตัวอย่างจะเรียกใช้ `program_pulibot()` เป็นอย่างแรก ถ้าเกรดเดอร์ตัวอย่างตรวจพบการเรียกใช้งานที่ผิดพลาด เกรดเดอร์ตัวอย่างจะพิมพ์ `Protocol Violation: <MSG>` แล้วหยุดทำงาน โดยที่ `<MSG>` เป็นอย่างใดอย่างหนึ่งต่อไปนี้

- `Invalid array`: $-2 \leq S[i] \leq Z_{MAX}$ ไม่เป็นจริงสำหรับบางค่า i หรือความยาวของ S ไม่ใช่ 5
- `Invalid color`: $0 \leq Z \leq Z_{MAX}$ ไม่เป็นจริง
- `Invalid action`: อักขระ A ไม่ใช่อันใดอันหนึ่งต่อไปนี้ `H`, `W`, `S`, `E`, `N` หรือ `T`
- `Same state array`: `set_instruction` ถูกเรียกด้วยค่า S ที่เหมือนกันอย่างน้อยสองครั้งขึ้นไป

หากไม่ได้เกิดปัญหาดังกล่าว เมื่อ `program_pulibot` ทำงานเสร็จสิ้นแล้วเกรดเดอร์ตัวอย่างจะทำงานตามโปรแกรมของพูลิบอทในเขาวงกตตามที่ระบุในข้อมูลนำเข้า

เกรดเดอร์ตัวอย่างจะแสดงข้อมูลสองอย่าง

อย่างแรก เกรดเดอร์ตัวอย่างจะพิมพ์บันทึกการทำงานของพูลิบอทลงในไฟล์ `robot.bin` ในไดเรกทอรีปัจจุบัน ไฟล์นี้จะถูกใช้ในเครื่องมือแสดงผลที่จะพูดถึงในหัวข้อถัดไป

อย่างที่สอง ถ้าโปรแกรมของพูลิบอทไม่ได้จบการทำงานอย่างถูกต้อง เกรดเดอร์ตัวอย่างจะแสดงข้อความระบุความผิดพลาดอย่างใดอย่างหนึ่งต่อไปนี้

- `Unexpected state`: พูลิบอทอ่านค่าอาร์เรย์สถานะที่ไม่ได้มีการระบุไว้ด้วย `set_instruction`
- `Invalid move`: มีการทำงานที่ทำให้พูลิบอทเคลื่อนที่ไปยังช่องที่ไม่ใช่ช่องว่าง
- `Too many steps`: พูลิบอททำงานมากกว่า $500\,000$ ขั้นโดยไม่ได้จบการทำงาน

แต่ถ้าหากไม่มีข้อผิดพลาดข้างต้น ให้ $e[r][c]$ เป็นสถานะของช่อง (r, c) หลังจากที่โปรแกรมของพูลิบอทจบการทำงาน เกรดเดอร์ตัวอย่างจะแสดงผล H บรรทัดในรูปแบบต่อไปนี้

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

เครื่องมือแสดงผล

ชุดไฟล์แนบของโจทย์ข้อนี้มีไฟล์ชื่อ `display.py` เมื่อเรียกใช้โปรแกรมภาษาไพธอนนี้จะแสดงการทำงานของพูลิบอทในเขาวงกตที่ระบุไว้ในข้อมูลนำเข้าของเกรดเดอร์ตัวอย่าง การใช้งานนี้จำเป็นต้องมีไฟล์ `robot.bin` อยู่ในไดเรกทอรีปัจจุบัน

การเรียกใช้โปรแกรมนี้ทำได้โดยเรียกคำสั่ง

```
python3 display.py
```

หน้าจอแสดงผลแบบกราฟฟิกจะแสดงขึ้น โดยมีการใช้งานหลักดังนี้

- คุณสามารถดูสถานะของเขาวงกตทั้งหมด ตำแหน่งปัจจุบันของพูลิบอทแสดงโดยรูปสี่เหลี่ยม

- คุณสามารถไล่ดูขั้นตอนการทำงานของพูลิบอทโดยการคลิกที่ปุ่มลูกศร หรือ กดปุ่มลัด คุณสามารถกระโดดไปยังขั้นตอนการทำงานตามที่ระบุได้ด้วย
- ขั้นตอนการทำงานถัดไปของโปรแกรมของพูลิบอทจะแสดงไว้ด้านล่าง โดยจะระบุอาร์เรย์สถานะและคำสั่งที่จะทำ หลังจากขั้นตอนสุดท้าย โปรแกรมจะแสดงข้อความระบุความผิดพลาดของเกรดเดอร์ หรือไม่ก็แสดง `Terminated` ถ้าโปรแกรมจบการทำงานอย่างถูกต้อง
- สำหรับค่าสีแต่ละค่า คุณสามารถกำหนดสีพื้นหลังที่จะใช้รวมถึงข้อความที่จะแสดง โดยข้อความที่จะแสดงจะเป็นสายอักขระสั้น ๆ ที่แสดงในแต่ละช่องที่มีค่าสีนั้น คุณสามารถกำหนดสีพื้นหลังและข้อความที่จะแสดงด้วยวิธีใดวิธีหนึ่งต่อไปนี้
    - ตั้งค่าดังกล่าวในหน้าจอที่แสดงหลังจากกดปุ่ม `Colors`
    - แก้ไขค่าในไฟล์ `colors.txt`
- หากต้องการโหลดข้อมูลจากไฟล์ `robot.bin` อีกครั้ง ให้กดปุ่ม `Reload` สิ่งนี้ใช้ประโยชน์ได้เมื่อข้อมูลในไฟล์ `robot.bin` มีการเปลี่ยนแปลง