MC102 - Algoritmos e Programação de Computadores
Turmas QRSTWY
Instituto de Computação - Unicamp
Professores: *Hélio Pedrini e Zanoni Dias*
Monitores: *Andre Rodrigues Oliveira, Gustavo Rodrigues Galvão, Javier Alvaro Vargas Muñoz e Thierry Pinheiro Moreira*

Lab 15a - Walk-Bot - Part II

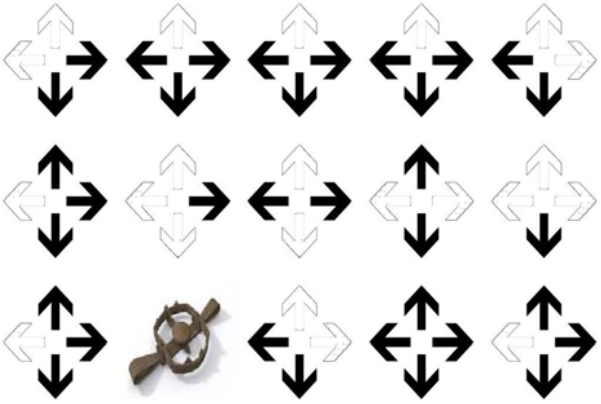
Prazo de entrega: 29/06/2015 às 13h59m59s
Peso: 10

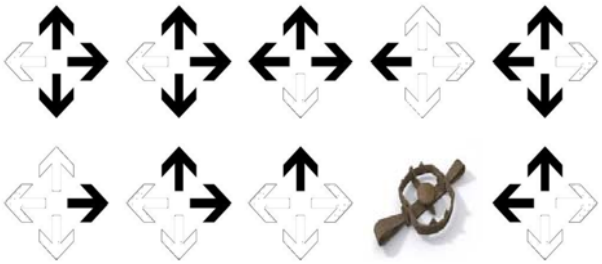
Os *feedbacks* da primeira versão do Walk-Bot não foram muito positivos. Os usuários gostaram da premissa do jogo, porém acharam que havia pouca interação: a única escolha que poderia ser feita era a casa inicial do Walk-Bot. Pensando nisso, a empresa LastLife desenvolveu uma segunda versão do jogo.

Nesta nova versão, o Walk-Bot deve andar por um tabuleiro tal que, em cada casa, há um conjunto de instruções possíveis. Desse modo, cabe ao jogador escolher qual deve ser a instrução executada caso haja mais do que uma. A ideia básica do jogo continua a mesma: o Walk-Bot deve atravessar o tabuleiro, isto é, começar em uma das casas localizadas na coluna mais à esquerda do tabuleiro e terminar em uma das casas localizadas na coluna mais à direita do tabuleiro, na qual deve existir uma instrução, dentre as possíveis, para ele ir à direita. A casa inicial continua sendo escolhida pelo jogador.

O desafio do jogo permanece praticamente o mesmo: o jogador deve realizar as escolhas (tanto da casa inicial quanto nas demais casas) que levem o Walk-Bot a atravessar o tabuleiro com o menor número de passos. Isso porque, a cada passo, o robô perde uma unidade de energia, que é parcialmente carregada a cada mudança de fase (ou tabuleiro). Desse modo, se o caminho de travessia for muito longo, a energia do Walk-Bot acaba e ele morre. Além disso, conforme as fases vão avançando, maiores são os tabuleiros e, portanto, mais energia é necessária para atravessá-los.

Para entender melhor como a nova versão do jogo funciona, considere que o Walk-Bot deve atravessar o tabuleiro (5x5) ilustrado abaixo. As setas preenchidas com a cor preta indicam as instruções (direções) possíveis de serem executadas naquela casa.





Dado que uma casa localizada na linha `i` e na coluna `j` pode ser identificada pelo par `(i,j)` e que a casa mais acima e mais à esquerda do tabuleiro é a casa `(1,1)`, nós podemos traçar os seguintes *caminhos mínimos* para o Walk-Bot no tabuleiro ilustrado acima:

- `(1,1)` → `(1,2)` → `(1,3)` → `(1,4)` → `(1,5)` → `(2,5)` → `(3,5)`
- `(2,1)` → `(2,2)` → `(2,3)` → `(2,4)` → `(3,4)` → `(3,5)`
- `(3,1)` → `(2,1)` → `(2,2)` → `(2,3)` → `(2,4)` → `(3,4)` → `(3,5)`
- `(4,1)` → `(3,1)` → `(2,1)` → `(2,2)` → `(2,3)` → `(2,4)` → `(3,4)` → `(3,5)`

Note que não foi listado nenhum caminho começando da casa `(5,1)`. Isso se deve ao fato de que não é possível atravessar o tabuleiro começando desta casa. Além disso, suponha que a energia inicial do Walk-Bot é 6. Dado que o Walk-Bot perde uma unidade de energia a cada passo dado, o único caminho de travessia possível é o caminho iniciado na casa `(2,1)`, no qual o Walk-Bot dá 5 passos e termina com uma unidade de energia. Nos demais caminhos, sua energia é zerada e, portanto, ele morre.

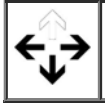
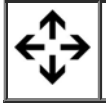
Tenha em mente que este foi apenas um exemplo. Podem existir tabuleiros em que há mais de um caminho que leve o Walk-Bot a atravessar o tabuleiro com vida. Além disso, você pode considerar que o Walk-Bot só consegue sair do tabuleiro pelas casas localizadas na coluna mais à direita. Em outras palavras, as casas localizadas na primeira linha (*i.e.* na linha mais acima) nunca terão uma instrução para o Walk-Bot ir para cima, as casas localizadas na última linha (*i.e.* na linha mais abaixo) nunca terão uma instrução para o Walk-Bot ir para baixo e, finalmente, as casas localizadas na primeira coluna (*i.e.* na coluna mais à esquerda) nunca terão uma instrução para o Walk-Bot ir para esquerda.

Você foi novamente um dos usuários selecionados para testar o jogo e dar um *feedback* para a LastLife. A fim de dar um bom *feedback*, você decidiu escrever um programa que, dado um tabuleiro, faz um relatório de quais são as casas iniciais que levam o Walk-Bot a atravessar o tabuleiro com vida. Com esse relatório, é possível classificar a dificuldade de cada tabuleiro.

Entrada

A entrada é constituída de várias linhas, tal que:

- A primeira linha da entrada contém 3 números inteiros `N`, `M` e `E`, onde:
  - `N` representa o número de linhas do tabuleiro, com $5 \leq N \leq 20$;
  - `M` representa o número de colunas do tabuleiro, com $5 \leq M \leq 20$;
  - `E` representa a quantidade de energia inicial do Walk-Bot, com $5 \leq E \leq 100$.
- As próximas `N` linhas contêm `M` números inteiros cada, pertencentes ao intervalo `[0,15]`. Cada número representa um conjunto de instruções possíveis de serem executadas na casa correspondente, tal como ilustrado nas tabelas abaixo. Note que cada número corresponde à representação decimal de um número binário com 4 bits $B_1B_2B_3B_4$, tal que B_1 indica se é possível ir para cima, B_2 indica se é possível ir para a direita, B_3 indica se é possível ir para baixo e B_4 indica se é possível ir para esquerda. Uma armadilha corresponde ao caso em que não é possível ir para nenhuma direção.

Casa	Binário	Decimal	Casa	Binário	Decimal	Casa	Binário	Decimal	Casa	Binário	Decimal
	0000	0		0100	4		1000	8		1100	12
	0001	1		0101	5		1001	9		1101	13
	0010	2		0110	6		1010	10		1110	14
	0011	3		0111	7		1011	11		1111	15

Saída

A saída deve ser constituída de `N` linhas, de tal modo que a linha `i` deverá ter o formato `"Sim"` caso exista um caminho começando na casa `(i,1)` que leve o Walk-Bot a atravessar o tabuleiro com vida e deverá ter o formato `"Nao"` caso contrário (com `i` variando de 1 a `N`).

Exemplos

#	Entrada	Saída
1	5 5 6 6 7 7 7 3 14 4 5 10 2 14 0 3 15 14 14 14 13 1 11 4 12 8 0 9	Nao Sim Nao Nao Nao
2	5 7 32 2 4 2 4 2 4 2 2 8 2 8 2 8 2 2 8 2 8 2 8 2 2 8 2 8 2 8 2 4 8 4 8 4 8 4	Nao Nao Nao Sim Sim
3	5 10 10 6 7 7 7 7 7 7 7 7 3 14 15 15 15 15 15 15 15 15 11 14 15 15 15 15 15 15 15 15 11 12 13 13 13 13 13 13 13 13 0 12 13 13 13 13 13 13 13 13 13	Nao Nao Nao Nao Sim
4	9 8 100 4 4 4 4 4 4 4 2 2 1 1 1 1 1 1 1 4 4 4 4 4 4 4 2 4 4 4 2 0 1 1 1 4 4 2 4 4 4 4 2 4 4 4 4 4 4 2 2 8 1 1 1 1 1 1 2 4 4 4 4 4 4 8 4 4 4 4 4 4 4 4 0	Nao Nao Nao Sim Nao Nao Nao Nao Nao
5	10 10 22 6 7 7 0 7 0 7 7 7 0	Sim Sim

	14 0 15 0 15 15 15 0 15 11	Sim
	14 0 15 0 15 0 15 0 15 11	Nao
	14 0 15 15 15 0 15 0 15 9	Nao
	14 0 15 0 15 0 15 0 15 15	Nao
	14 0 15 0 15 0 15 0 15 15	Nao
	14 0 15 15 15 0 15 0 15 3	Sim
	14 0 15 0 15 0 15 0 15 11	Sim
	14 0 15 0 15 15 15 0 15 11	Sim
	12 13 13 0 13 0 13 13 13 0	