

Centro de Ciências Exatas, Ambientais e de Tecnologias

Faculdade de Análise de Sistemas Curso Sistemas de Informação

Projeto Integrado C 1º Trabalho do 1º Semestre de 2019

O propósito deste trabalho é, usando pilhas, construir em Java um programa apropriadamente orientado a objetos que encontre o caminho até a saída de um labirinto. Para tanto, considere que existam arquivos texto que descrevam labirintos, como, por exemplo, um arquivo chamado teste1.txt com o seguinte conteúdo:

5 ####### E # ### ### ### ####

Na primeira linha dele, <u>5</u> indica que o labirinto será descrito por <u>5</u> linhas texto. Veja que as as linhas de texto que descrevem o labirinto propriamente dito.

O número de caracteres de cada linha deverá ser identificado por você quando feita a leitura da primeira linha, lembrando que todas as linhas deverão conter a mesma quantidade de caracteres para a formação do labirinto. Caso haja diferença na quantidade de caracteres de cada linha, você não deverá prosseguir com a verificação do caminho e deverá notificar o usuário que há inconsistência no arquivo.

Nas linhas de texto, o caractere <u>#</u> representa parede, o <u>espaço em branco</u> representa passagem, o caractere <u>E</u> representa a entrada do labirinto e o caractere <u>S</u> representa a saida do labirinto.

Para resolver o problema, seu programa deve realizar uma sequência de passos, a saber:

- Solicitar a digitação do nome do arquivo texto que contém a estrutura do labirinto; suponha que tenha sido ditigado <u>teste1.txt</u>;
- 2. Carregar os caracteres do arquivo texto em uma matriz chamada <u>labirinto</u>. no caso deste exemplo, esta matriz terá <u>5</u> linhas e <u>8</u> colunas. A quantidade de linhas você deverá ler do arquivo. A quantidade de colunas você irá descobrir quando ler a segunda linha do arquivo, e verificar a quantidade de caracteres desta linha. Esta quantidade de caracteres da segunda linha deverá ser assumida como a quantidade de colunas a matriz. Desta forma, garanta que ao ler as demais linhas do arquivo, estas estarão consistentes com o tamanho da segunda linha, e caso não esteja, deverá notificar o usuário que o arquivo está inconsistente. Caso seja possível carregar os dados do arquivo, o labirinto fica assim:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Е							#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

- Instanciar um objeto do tipo <u>Pilha<Coordenada></u> chamada <u>caminho</u> com capacidade para armazenar <u>5</u>x8, ou seja 40, coordenadas;
- 4. Instanciar um objeto do tipo <u>Pilha<Pilha<Coordenada>></u> chamado <u>possibilidades</u> com capacidade para armazenar <u>5</u>x<u>8</u>, ou seja 40, pilha de coordenada;
- 5. Procurar a entrada do labirinto (o caractere E) nas bordas do <u>labirinto</u>, acusando erro, caso não encontre ou instanciando um objeto chamado <u>atual</u> da classe Coordenada para representar a posição onde o caractere E foi encontrado; no caso, (1,0).
- 6. Instanciar um objeto do tipo <u>Pilha<Coordenada></u> chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali as possíveis posições adjacentes a <u>atual</u> (acima, abaixo, à esquerda ou à direita) válidas e contendo um <u>espaço em branco</u> ou o caractere <u>S</u>, indicando, respectivamente, passagem livre ou saída; no caso a pilha gerada conteria:

(1,1)

7. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; <u>atual</u> fica (1,1) e <u>pilha de adjacentes</u> passa a conter:



8. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*						#
2 3 4	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

9. Empilha <u>atual</u> em <u>caminho</u> ; assim, <u>caminh</u>	o fica:
	1,1)
10. Empilha a pilha de adjacentes em possibi	lidades; assim, possibilidades fica:

11. Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali
as possíveis posições adjacentes a <u>atual</u> (acima, abaixo, à esquerda ou à direita) válidas e
contendo um espaço em branco ou o caractere S , indicando, respectivamente, passagem
livre ou saída; no caso a pilha de adjacentes gerada conteria:
(1,2)
12. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u> ; <u>atual</u> fica (1,2) e
pilha de adjacentes passa a conter:
13. Colocar no labirinto na posição indicada por atual um caractere *, indicando que foi dado um
passo naquela direção; assim, <u>labirinto</u> fica:
0 1 2 3 4 5 6 7 0 # # # # # # # # 1 E * * *
(1,2)
15. Empilha a <u>pilha de adjacentes</u> em <u>possibilidades</u> ; assim, <u>possibilidades</u> fica:

16.Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali
as possíveis posições adjacentes a <u>atual</u> (acima, abaixo, à esquerda ou à direita) válidas e
contendo um espaço em branco ou o caractere S, indicando, respectivamente, passagem
livre ou saída; no caso a <u>pilha de adjacentes</u> gerada conteria:

(1,3)

17. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; <u>atual</u> fica (1,3) e <u>pilha de adjacentes</u> passa a conter:



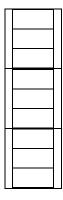
18. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Е	*	*	*				#
2	#	#	#		#	#	#	#
2 3 4	#	#	#		#	#	#	#

19. Empilha atual em caminho; assim, caminho fica:

(1,3)	
(1,2)	
(1,1)	

20. Empilha pilha de adjacentes em possibilidades; assim, possibilidades fica:



21	.Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
	chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali
	as possíveis posições adjacentes a atual (acima, abaixo, à esquerda ou à direita) válidas e
	contendo um espaço em branco ou o caractere S , indicando, respectivamente, passagem
	livre ou saída; no caso a pilha de adjacentes gerada conteria:

(1,4) (2,3)

22. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; <u>atual</u> fica (1,4) e <u>pilha de adjacentes</u> passa a conter:



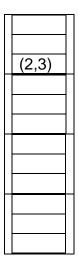
23. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Е	*	*	*	*			#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

24. Empilha atual em caminho; assim, caminho fica:

(1,4)	
(1,3)	
(1,2)	
(1,1)	

25. Empilha a **pilha de adjacentes** em **possibilidades**; assim, **possibilidades** fica:



26	Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
	chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali
	as possíveis posições adjacentes a atual (acima, abaixo, à esquerda ou à direita) válidas e
	contendo um espaço em branco ou o caractere S, indicando, respectivamente, passagem
	livre ou saída; no caso a pilha de adjacentes gerada conteria:

(1,5)

27. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; <u>atual</u> fica (1,5) e <u>pilha de adjacentes</u> passa a conter:



28. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*	*	*		#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

29. Empilha atual em caminho; assim, caminho fica:

(1,5)	
(1,4)	
(1,3)	
(1,2)	
(1,1)	

30. Empilha a <u>pilha de adjacen</u>	tes em possibilidades; assim, possibilidades fica:	
	(2,3)	

31	I.Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
	chamado <u>pilha de adjacentes</u> com capacidade de armazenar 3 coordenadas e empilhar ali
	as possíveis posições adjacentes a <u>atual</u> (acima, abaixo, à esquerda ou à direita) válidas e
	contendo um espaço em branco ou o caractere S , indicando, respectivamente, passagem
	livre ou saída; no caso a pilha de adjacentes gerada conteria:



32. Retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; <u>atual</u> fica (1,6) e <u>pilha de adjacentes</u> passa a conter:



33. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*	*	*	*	#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

34. Empilha atual em caminho; assim, caminho fica:

(1,6)	
(1,5)	
(1,4)	
(1,3)	
(1,2)	
(1,1)	

35. Empilha <u>a pilha de adjacentes</u> em <u>poss</u>	ibilidades; assim, possibilidades fica:
35. Empilha <u>a pilha de adjacentes</u> em <u>poss</u>	ibilidades; assim, possibilidades fica:

36. Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo <a href="Pilha<Coordenada">Pilha<Coordenada chamado pilha de adjacentes com capacidade de armazenar 3 coordenadas e empilhar ali as possíveis posições adjacentes a atual (acima, abaixo, à esquerda ou à direita) válidas e contendo um espaço em branco ou o caractere S, indicando, respectivamente, passagem livre ou saída; no caso a pilha de adjacentes gerada conteria:



37. Agora, o que deveria ser feito seria retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; ocorre que isso será impossível de ser feito, um vez que <u>pilha de adjacentes</u> se encontra vazia; até agora estivemos em Modo Progressivo e progredíamos na construção de um caminho em <u>labirinto</u> marcado por <u>*</u>; por conta da situação, isso vai mudar e o programa entra em um modo de funcionamento que podemos chamar de Modo Regressivo que funciona da seguinte forma:

38. Desempilha em atual uma coordenada	do caminho ; atual fica (1,6) e caminho passa a
conter:	
	(1,5)
	(1,4)
	(1,3)
	(1,1)

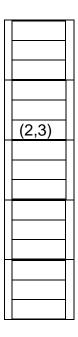
39. Retirar do <u>labirinto</u>, da posição indicada por <u>atual</u> o caractere *, anteriormente ali colocado, substituindo-o por um <u>espaço em branco</u>, indicando que foi dado um passo para trás; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*	*	*		#
2	#	#	#		#	#	#	#
2 3 4	#	#	#		#	#	#	#

40. Desempilha em **possibilidades uma pilha de adjacentes**; **pilha de adjacentes** passa a conter:



e **possibilidades** passa a conter:



41. Para que o programa retornasse ao Modo Progressivo, agora, o que deveria ser feito seria retirar da pilha de adjacentes uma coordenada, armazenando-a em atual; ocorre que isso

será impossível de ser feito, um vez que <u>pilha de adjacentes</u> se encontra vazia; assim sendo, o programa se mantem em Modo Regressivo e continuarmos a partir do passo 38, ou seja:

42. Desempilha em <u>atual</u> uma coordenada do <u>caminho</u>; <u>atual</u> fica (1,5) e <u>caminho</u> passa a conter:

(1,4)	
(1,3)	
(1,2)	
(1,1)	

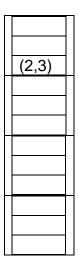
43. Retirar do <u>labirinto</u>, da posição indicada por <u>atual</u> o caractere *, anteriormente ali colocado, substituindo-o por um <u>espaço em branco</u>, indicando que foi dado um passo para trás; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*	*			#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

44. Desempilha em **possibilidades uma pilha de adjacentes**; **pilha de adjacentes** passa a conter:



e **possibilidades** passa a conter:



45. Para que o programa retornasse ao Modo Progressivo, agora, o que deveria ser feito seria retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; ocorre que isso será impossível de ser feito, um vez que <u>pilha de adjacentes</u> se encontra vazia; assim sendo, o programa se mantem em Modo Regressivo e continuarmos a partir do passo 38, ou seja:

46. Desempilha em <u>atual</u> uma coordenada do <u>caminho</u>; <u>atual</u> fica (1,4) e <u>caminho</u> passa a conter:

(1	,3)
(1	,2)
(1	,1)

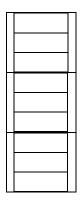
47. Retirar do <u>labirinto</u>, da posição indicada por <u>atual</u> o caractere *, anteriormente ali colocado, substituindo-o por um <u>espaço em branco</u>, indicando que foi dado um passo para trás; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*				#
2	#	#	#		#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

48. Desempilha em **possibilidades uma pilha de adjacentes**; **pilha de adjacentes** passa a conter:



e **possibilidades** passa a conter:



49. Para que o programa retornasse ao Modo Progressivo, agora, o que deveria ser feito seria retirar da <u>pilha de adjacentes</u> uma coordenada, armazenando-a em <u>atual</u>; felizmente, agora isso volta a ser possível de ser feito, um vez que <u>pilha de adjacentes</u> não se encontra vazia; assim sendo, o programa sai do modo Modo Regressivo, volta ao Modo Progressivo e continuarmos a partir do passo 7, ou seja:

50. Retirar da pilha de adjacentes uma coord	rdenada, armazenando-a em <u>atual</u> ; <u>atual</u> fica (2,3) e
pilha de adjacentes passa a conter:	
Г	

51. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere *, indicando que foi dado um passo naquela direção; assim, <u>labirinto</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*				#
2	#	#	#	*	#	#	#	#
3	#	#	#		#	#	#	#
4	#	#	#	S	#	#	#	#

(2,3) (1,3) (1,2) (1,1)
53. Empilha pilha de adjacentes em possibilidades ; assim, possibilidades fica:
54.Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo Pilha <coordenada></coordenada>
chamado pilha de adjacentes com capacidade de armazenar 3 coordenadas e empilhar al
as possíveis posições adjacentes a <u>atual</u> (acima, abaixo, à esquerda ou à direita) válidas e
contendo um espaço em branco ou o caractere S , indicando, respectivamente, passagem
livre ou saída; no caso a <u>pilha de adjacentes</u> gerada conteria:
(3,3)
55. Retirar da pilha de adjacentes uma coordenada, armazenando-a em atual; atual fica (3,3) e
pilha de adjacentes passa a conter:
56. Colocar no <u>labirinto</u> na posição indicada por <u>atual</u> um caractere <u>*</u> , indicando que foi dado um
passo naquela direção; assim, <u>labirinto</u> fica:

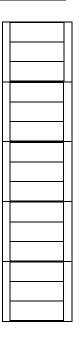
52. Empilha <u>atual</u> em <u>caminho</u>; assim, <u>caminho</u> fica:

	0	1	2	3	4	5	6	7
0	#	#	#	#	#	#	#	#
1	Ε	*	*	*				#
2	#	#	#	*	#	#	#	#
2	#	#	#	*	#	#	#	#
4	#	#	#	S	#	#	#	#

57. Empilha atual em caminho; assim, caminho fica:

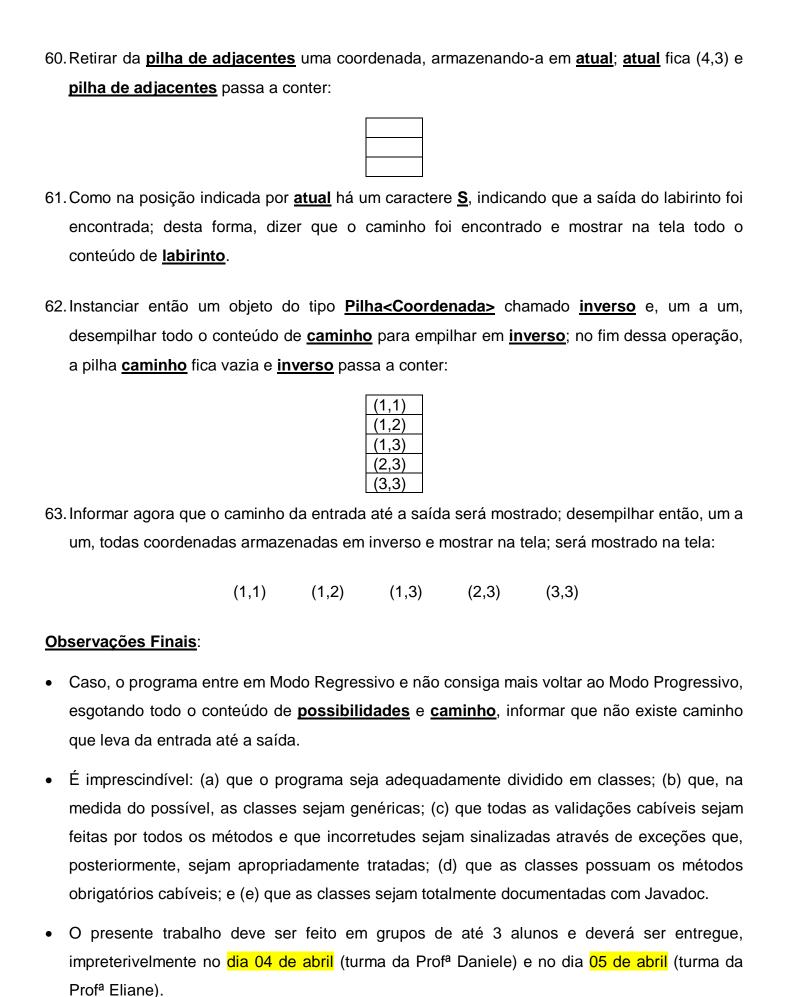
(3,3)	
(2,3)	
(1,3)	
(1,2)	
(1,1)	

58. Empilha pilha de adjacentes em possibilidades; assim, possibilidades fica:



59. Continuar a partir do passo 6, ou seja, instanciar um objeto do tipo <a href="Pilha<Coordenada">Pilha<Coordenada chamado pilha de adjacentes com capacidade de armazenar 3 coordenadas e empilhar ali as possíveis posições adjacentes a atual (acima, abaixo, à esquerda ou à direita) válidas e contendo um espaço em branco ou o caractere S, indicando, respectivamente, passagem livre ou saída; no caso a pilha de adjacentes gerada conteria:





Bom trabalho! Profs André Carvalho, Daniele Frosoni, Campinas, 28/Fev/2019
Eliane Azevedo e Patrícia Nogueira