

# Primeiro Exercício-Programa

Prof. Luciano Antonio Digiampietri

Prazo máximo para a entrega: 03/12/2023

## 1 Resolvendo um Labirinto

Neste trabalho, você deverá desenvolver uma função, baseada em tentativa e erro (*backtracking*), para solucionar um labirinto.

**Detalhamento:** Para este EP, há apenas uma estrutura de dados que será utilizada, chamada *LABIRINTO*.

**LABIRINTO:** tem por objetivo representar um labirinto. É uma estrutura que possui sete campos: *linhas* do tipo inteiro que corresponde ao número de linhas (largura) do labirinto; *colunas* do tipo inteiro que corresponde ao número de colunas (comprimento) do labirinto; *inicioX* do tipo inteiro que contém sua posição inicial no eixo X; *inicioY* do tipo inteiro que contém sua posição inicial no eixo Y; *fimX* do tipo inteiro que contém a posição fim (destino/saída) no eixo X; *fimY* do tipo inteiro que contém a posição fim (destino/saída) no eixo Y; *labirinto* do tipo ponteiro para ponteiro de caracteres que corresponde a uma matriz de caracteres que representa o labirinto. Nessa matriz, o caractere '#' indica uma parede, o caractere 'I' indica o início e o caractere 'F' indica o fim. O labirinto é cercado por paredes e as posições que não possuem parede, início ou fim são indicadas com espaços em branco.

Exemplo de labirinto com quatro linhas e quatro colunas:

```
####  
#I #  
# F#  
####
```

	Coluna 0	Coluna 1	Coluna 2	Coluna 3
Linha 0	#	#	#	#
Linha 1	#	I		#
Linha 2	#		F	#
Linha 3	#	#	#	#

Exemplo de labirinto com dez linhas e vinte colunas:

```
#####
# # # # # ##
## # # ##
# ## # # # #
# # # # # #
# # ## ## ## #
# # #
## # ## # # ##
#I # # F#
#####
```

O código fornecido para este EP já possui várias funções implementadas para gerar diferentes labirintos e imprimir os labirintos.

**Há uma função que você deverá implementar/completar, podendo, se julgar conveniente, implementar funções adicionais/auxiliares:**

- *bool resolveLabirinto(LABIRINTO lab)*: função que recebe um labirinto como parâmetro e deve resolvê-lo seguindo estritamente uma abordagem de tentativa e erro. Se for possível resolver o labirinto (isto é, a partir do início chegar no final) a função deverá retornar *true*, caso contrário deverá retornar *false*. Adicionalmente, o percurso realizado dentro do labirinto deve ser indicado com o caractere '.' (ponto) no caso da função conseguir resolvê-lo. Este percurso deverá seguir uma abordagem de tentativa e erro conforme descrito a seguir.

**Estratégia:** a partir de cada posição (começando do início) há quatro ações potencialmente possíveis (que devem ser seguidas nesta ordem): ir para baixo (isto é, aumentar 1 no valor da linha, sem mudar de coluna); ir para a esquerda (isto é, diminuir 1 no valor da coluna, sem mudar de linha); ir para a direita (isto é, aumentar 1 no valor da coluna, sem mudar de

linha); ir para cima (isto é, diminuir 1 no valor da linha, sem mudar de coluna). Um movimento possível será viável caso a posição de destino esteja preenchida com um espaço em branco no labirinto, neste caso, deverá ser colocado um '.' (ponto) nessa posição e o algoritmo deverá tentar um novo movimento a partir dessa posição. Se a posição de destino for o fim ('F'), a função deverá retornar *true*. Se não for 'F' nem '.', então o movimento não será válido e o algoritmo deverá tentar outro movimento. Se nenhum movimento for possível a partir da posição atual, o algoritmo deverá desmarcar a posição atual (substituir o '.' por ' ') e voltar (*backtracking*) para tentar a solução por outro caminho.

**Observação:** apesar de, potencialmente, existirem diversas soluções para cada labirinto, só existe uma solução que considera a estratégia apresentada. Sua implementação deve encontrar essa solução.

**Exemplos:** a seguir são apresentadas as soluções para diferentes labirintos.

```
####
#I #
#.F#
####
```

```
#####
#I #
#. #
#..F#
#####
```

```
#####
#F..#
#...#
#..I#
#####
```

```
#####
# # # # # # # #
## # # # # # #
# ## # # # # #
# # # # # # # #
#.....# ## ## #
#.. #. #. #. #
##..#..## #.#..## #
#I.. #..... #...F#
#####
```

```
#####
#I# # # ## # # #
#. # ... # # ## #
#. ##...#..... # #### # #
#. #.....# #.. # ## ## #
#. # #...# #.. ## # #
#. ####.# ## #F# #
#. #.....#### # ## #####
#..... # # ## # #
##...#.....# # #
# .....###.# # ## #
#####
```

## 1.1 Material a Ser Entregue

Um arquivo, denominado *NUSP.c* (sendo NUSP o seu número USP, por exemplo: 123456789.c), contendo seu código, incluindo todas as funções solicitadas e qualquer outra função adicional que ache necessário. Para sua conveniência, *completeERenomeie.c* será fornecido, cabendo a você então completá-lo e renomeá-lo para a submissão.

### Atenção!

1. Não modifique as assinaturas das funções já implementadas e/ou que você deverá completar!
2. Para avaliação, a função solicitada será invocada diretamente. Em especial, qualquer código dentro da função `main()` será ignorado.

## 2 Entrega

A entrega será feita única e exclusivamente via sistema e-Disciplinas, até a data final marcada. Deverá ser postado no sistema um arquivo .c, tendo como nome seu número USP:

`seuNumeroUSP.c` (por exemplo, `12345678.c`)

Não esqueça de preencher o cabeçalho constante do arquivo .c, com seu nome, número USP, turma etc.

A responsabilidade de postagem é exclusivamente sua. Por isso, submeta e certifique-se de que o arquivo submetido é o correto (fazendo seu download, por exemplo). Problemas referentes ao uso do sistema devem ser resolvidos com antecedência.

## 3 Avaliação

A nota atribuída ao EP será baseada nas funcionalidades solicitadas, porém não esqueça de se atentar aos seguintes aspectos:

1. Documentação: se há comentários explicando o que se faz nos passos mais importantes e para que serve o programa (tanto a função quanto o programa em que está inserida);
2. Apresentação visual: se o código está legível, indentado etc;
3. Corretude: se o programa funciona.

Além disso, algumas observações pertinentes ao trabalho, que influenciam em sua nota, são:

- Este exercício-programa deve ser elaborado individualmente;
- Não será tolerado plágio;
- Exercícios com erro de sintaxe (ou seja, erros de compilação), receberão nota ZERO.