Tópicos de Programação

Quarto Exercício-Programa (Primeira Versão)

Entrega: 06/02/2017 até 23:55

Verão 2017 - IME-USP

Solução de um labirinto

Nesse exercício-programa iremos encontrar um caminho para sair de um labirinto.

O labirinto será um tabuleiro quadriculado com m x n casas distribuidas em m linhas e n colunas. As casas livres (caminhos livres) são marcadas por 0 e as casas ocupadas (paredes do labirinto) são marcadas por 1.

Na casa (1, 1), que sempre é livre, existe um robô que deve encontrar a saída do labirinto, que é a posição $(m-1, n-1)$, e pode se mover nas seguintes direções **norte, leste, sul** e **oeste**, uma casa por vez, desde que a casa a ser alcançada seja livre. Você deve descobrir se existe um caminho para a saída do labirinto e, se o caminho existir, imprimir a distância entre a casa (1, 1) e a casa $(m-1, n-1)$.

Exemplos:

Labirinto 1 (3x3):
Distância da casa incial à casa final: 4

```
011
000
110
```

Labirinto 2 (4x4):
Distância da casa inicial à casa final: 6

```
0000
1010
1010
1100
```

Labirinto 3 (6x9): Distância da casa inicial até à casa final: 13

```
011001101
001101001
100001011
010010001
001000101
100110000
```

Tarefas

Para resolver o problema, você deve utilizar uma **fila com a implementação em lista encadeada**.

Você deve criar as seguintes funções:

- **insereFila (int linha, int coluna, Fila *f);** que insere um elemento no fim da fila;
- **No *removeFila (Fila *f);** que retira um elemento do inicio da fila;
- **void imprimeFila (Fila *f);** que imprime a fila atual começando pelo início;
- **int resolveLabirinto (int labirinto[LIN][COL], int m, int n);** que retorna 1 e imprime o valor da distância entre a casa incial e a casa final, caso exista um caminho até a saída, ou retorna 0, caso não exista um caminho.

Você deve implementar uma fila e seu código deve seguir o seguinte modelo:

```c
#include <stdio.h>
#include <stdlib.h>

#define LIN 3
#define COL 3

typedef struct no No;
struct no{
    int linha;
    int coluna;
    No *prox;
};

typedef struct fila Fila;
struct fila{
    No *inicio;
    No *fim;
    int tamanho;
};

void insereFila (int linha, int coluna, Fila *f);
No *removeFila (Fila *f);
void imprimeFila (Fila *f);
int resolveLabirinto (int labirinto[LIN][COL], int m, int n)
    ;
```

Você pode implementar mais funções se achar necessário.

Para facilitar e uniformizar o desenvolvimento, optamos por passar matrizes que representam o labirinto como no exemplo abaixo. Não se esqueçam de

alterar o valor das macros LIN e COL, nas linhas 4 e 5 do modelo, para a dimensão dos matrizes a serem passadas para a função resolveLabirinto.

```
1  /* ... */
2  int main(){
3      int m, n;
4
5      /* Declaracao dos labirintos */
6
7      int labirinto1[3][3] = { {0,1,1}, {0,0,0}, {1,0,0}};
8      /*int labirinto2[4][4] = { {0,0,1,1}, {1,0,0,0},
       {1,0,1,0}, {1,0,0,0}};
9      int labirinto3[5][6] = { {0,0,1,0,1,1}, {1,0,0,0,1,0},
       {1,1,0,1,0,0}, {1,0,0,0,0,1}, {1,1,1,1,0,0}};*/
10     m = 3;
11     n = 3;
12     resolveLabirinto (labirinto1, m, n);
13
14     m = 4;
15     n = 4;
16     /*resolveLabirinto (labirinto2, m, n);*/
17
18     m = 5;
19     n = 6;
20     /*resolveLabirinto (labirinto3, m, n);*/
21
22     return 0;
23 }
```

Bônus: Caminho percorrido

Altere sua função resolveLabirinto() para que, além da distância, ela imprima o caminho percorrido no labirinto.

Exemplo: para o labirinto 1 (3x3):

```
011
000
110
```

Distância da casa incial à casa final: 4
Caminho percorrido:
(0, 0) -> (1, 0) -> (1, 1) -> (1, 2) -> (2, 2)

Informações importantes

O programa entregue deve seguir os itens abaixo:

- O EP deverá ser feito individualmente e plágio não será tolerado;

- O cabeçalho de cada função pedida deve seguir o modelo indicado na sua descrição para fins de correção;
- Você pode adicionar mais bibliotecas caso necessite;
- Outras funções auxiliares podem ser criadas para facilitar o desenvolvimento, desde que as funções pedidas no enunciado estejam presentes e se comportem de acordo com o que foi pedido;
- O EP deve ser entregue no PACA em um .zip contendo o arquivo **ep4.c**;
- Compile o seu EP usando o compilador GCC;
- Utilize as flags de compilação: -Wall -ansi -pedantic -O2
- Bom desenvolvimento e divirta-se!