Universidade do Estado de Santa Catarina (UDESC)
Curso: Tecnologia em Análise e Desenvolvimento de Sistemas
Disciplina: Estrutura de dados (EDA0001)
Prof. Rui Jorge Tramontin Jr.

Trabalho 2: Menor caminho em um Labirinto usando BFS

O objetivo deste trabalho é implementar o algoritmo BFS (*Breadth-first Search*) ou ***Busca em Largura*** para realizar o menor caminho entre dois pontos de um labirinto. O labirinto é modelado na forma de uma matriz *M*x*N*, onde os caminhos são representados por 1 e as paredes por 0. O algoritmo se inicia em uma célula (linha *i*, coluna *j*) e explora todas células vizinhas (no mesmo nível), antes de se mover para as células no próximo nível de profundidade. Para isso, a implementação desse algoritmo utiliza uma **fila**.

Os dados de entrada são:

- Matriz *M*x*N* representando o labirinto;
- Célula inicial A (linha *ai*, coluna *aj*);
- Célula final B (linha *bi*, coluna *bj*).

O algoritmo começa em A e faz a busca em largura até encontrar B. Uma aplicação do algoritmo BFS é que ele permite encontrar o **menor caminho entre A e B**, em termos de número de células percorridas. Nesse sentido, após a aplicação da busca, a saída deve ser a **quantidade** de células visitadas entre A e B.

Implementação

O labirinto é representando como uma matriz de inteiros, na qual o valor 1 representa um caminho e o valor 0 representa uma parede. Uma célula adjadente deve estar acima, abaixo, à direita ou à esquerda e deve ter o valor 1. Por exemplo, considere o labirinto 7x6 a seguir e as seguintes células A e B:

Célula Inicial A (2,1)

Célula Final B (7,6)

	1	2	3	4	5	6
1	1	1	1	1	1	0
2	1	0	0	0	1	0
3	1	1	1	0	1	1
4	1	0	1	0	0	1
5	1	0	1	1	1	1
6	1	1	1	0	0	1
7	0	0	0	0	0	1

O resultado da busca para este exemplo é 10, pois representa o menor caminho entre A e B.

A matriz pode ser facilmente implementada utilizando a biblioteca de matrizes dinâmicas feita como exercício em aula (Lista 2). Repare que, para o usuário, os índices são numerados a partir de 1, enquanto que na linguagem C, os índices começam em 0.

As seguintes estruturas são necessárias para o algorirmo:

1. Matriz: de inteiros, alocada dinamicamante, conforme entrada do usuário;
2. Matriz de *status* das células: também alocada dinamicamente; indica se cada célula foi visitada (1), ou se ainda não foi (0);
3. *struct* com informações sobre uma célula: coordenadas *x* e *y*, e *distância* até a célula inicial (todos os campos inteiros);
4. Fila genérica: a bilioteca de filas deve se usada para armazenar as *structs* definidas no item anterior.

O algoritmo é implementado conforme o seguinte pseudocódigo:

```
 1. ENTRADA DE DADOS:
     - Matriz de inteiros (suas dimensões e seu conteúdo);
     - Coordenadas da célula inicial do percurso A(ai,aj);
     - Coordenadas da célula final do percurso B(bi,bj);
 2. INICIALIZA MATRIZ DE STATUS (MS): todas as posições com zero;
 3. INICIALIZA FILA F;
 4. MS[ai,aj] <- 1;          // Marca A como visitada,
 5. INSERE (ai,aj,0) em F;   // e insere na Fila (distância 0).
 6. ACHOU <- FALSO           // Variável booleana
 7. ENQUANTO F NÃO ESTIVER VAZIA E NÃO ACHOU FAÇA
 8.    REMOVE A CÉLULA C(x,y,dist) DE F;
 9.    SE C(x,y) = B(bi,bj) ENTÃO    // Se for o vértice final...
10.       ACHOU <- VERDADEIRO;       // Para interromper o laço.
11.    SENÃO
12.       PARA CADA CÉLULA (i,j) ADJACENTE A C FAÇA
13.          SE MS[i,j] = 0 ENTÃO         // Se ainda não foi visitada,
14.             MS[i,j] <- 1;                // - marca como visitada;
15.             INSERE (i,j,C.dist+1) EM F;  // - e insere na fila.
16.          FIMSE
17.       FIMPARA
18.    FIMSE
19. FIMENQUANTO
20. SE ACHOU ENTÃO
21.    MOSTRA C.dist NA TELA;
22. SENÃO
23.    ESCREVA "Não há caminho possível entre A(ai,aj) e B(bi,bj)!";
24. FIMSE
25. DESALOCA FILA F;
```

Entrada

A primeira linha da entrada contém os inteiros *M* e *N*, que representam as dimensões da matriz. Em seguida, são lidas cada uma das *M* linhas da matriz. Finalmente, temos como entrada o vértices A (inicial) e B (final).

Saída

A distância entre A e B (em termos de quantidade de células percorridas).

Exemplo

Exemplo de entrada	Exemplo de saída
7 6 1 1 1 1 1 0 1 0 0 0 1 0 1 1 1 0 1 1 1 0 1 0 0 1 1 0 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 1 2 1 7 6	10

Critérios de avaliação

- Execução correta e alinhamento com o que foi solicitado neste enunciado;
- Uso apropriado das funções dos *tipos abstratos de dados* (matriz e fila). Respeite o encapsulamento!

Informações importantes

- **Equipe:** 1 ou 2 alunos.
- **Entrega**: no Moodle, até o dia **12/03**.