Terceiro Exercício Programa

MAC0110 - BMAC Noturno 2011

Professor Roberto M Cesar Jr
Entrega até 30/06/2011

Alagações na USSP

Devido a um forte temporal que caiu em São Paulo, várias regiões da cidade universitária da USSP (Universidade de Sábios Samaritanos Pragmáticos) ficaram alagadas. Representamos o *campus* da USSP por um reticulado, como o da figura abaixo, onde o rótulo 0 representa uma posição **seca** e o rótulo -1 representa uma posição **alagada**

-1	0	0	0	0	-1	-1	0	-1	-1
-1	-1	-1	0	0	0	-1	0	-1	0
0	-1	0	0	-1	0	0	0	0	0
0	0	0	-1	-1	-1	-1	-1	0	0
0	0	0	0	-1	-1	0	0	-1	-1
0	0	-1	0	0	0	0	-1	-1	-1
0	0	-1	-1	-1	0	0	0	-1	-1
-1	0	-1	-1	-1	-1	0	0	0	-1
-1	0	0	0	0	0	-1	-1	0	0
-1	0	0	0	0	0	-1	0	0	0

Dizemos que duas posições deste reticulado são **adjacentes** se são vizinhas entre si. Uma **região R** é definida como um conjunto de posições com mesmo rótulo, tal que é possivel, a partir de uma posição de **R**, atingir qualquer outra posição deste conjunto através de posições vizinhas. Se todas as posições de uma região são rotuláda -1, então é denominada **alagada**.

Objetivo

Faça um programa em C que resolva o seguinte problema: dado um reticulado do tipo acima (uma matriz), marcar todas as regiões alagadas maximais. Especificações a serem seguidas:

1. Dados: dois inteiros m e n e uma matriz $U_{m\times n}$ inteira representando um reticulado com entradas 0 e -1. (Na figura acima temos $m = n = 10$)

2. A partir da matriz U, construir uma nova matriz $M_{m\times n}$, onde estão rotuladas cada **região alagada maximal** com um rótulo (número inteiro) da seguinte forma: se existirem k regiões alagadas maximais, então cada posição de uma mesma região recebe um número entre 1 e k, sendo que todas as posições de uma mesma região devem receber um mesmo número. Por exemplo, o reticulado acima tem 8 regiões alagadas maximais. Neste caso um exemplo de saída seria:

1	0	0	0	0	2	2	0	3	3
1	1	1	0	0	0	2	0	3	0
0	1	0	0	4	0	0	0	0	0
0	0	0	4	4	4	4	4	0	0
0	0	0	0	4	4	0	0	5	5
0	0	6	0	0	0	0	5	5	5
0	0	6	6	6	0	0	0	5	5
7	0	6	6	6	6	0	0	0	5
7	0	0	0	0	0	8	8	0	0
7	0	0	0	0	0	8	0	0	0

3. Imprimir a matriz dada U e matriz rotulada.

4. Use obrigatoriamente as seguintes funções:

   (a) `void cria_lista (int m, int n, int MAT[][MAX_LIN], int LISTA[][3])`

      i. parâmetros:
      m, n: números inteiros
      MAT: matriz inteira $m \times n$
      LISTA: matriz inteira $(m * n) \times 3$

      ii. descrição:
      A partir da matriz MAT essa função deve construir a matriz LISTA, e esta deverá indicar quais são as posições alagadas de MAT. A informação de que uma posição (x, y) de MAT está alagada deve ficar armazenada nas linhas da matriz LISTA (a primeira coluna armazena x, a segunda armazena y e os valores da terceira deverão conter -1). Se a matriz MAT tiver t posições alagadas, então as t primeiras linhas da matris LISTA armazenarão essas informações. As linhas restantes deverão conter -3 por convenção.

   (b) `int vizinho (int x, int y, int z, int w)`

      i. parâmetros:
      x, y, z, w: números inteiros

      ii. descrição:
      Se a posição (x, y) for vizinha da posição (z, w), a função devolve o valor 1, ou devolve 0 caso contrário.

   (c) `void rotula_com_k (int k, int lin, int MAT[][MAX_LIN], int LISTA[][3])`

      i. parâmetros:
      k, lin: números inteiros
      MAT: matriz inteira $m \times n$
      LISTA: matriz inteira $(m * n) \times 3$

      ii. descrição:
      Esta função recebe como entrada a matriz MAT e realiza a rotulação com k da região alagada a partir do valor de (x, y) da linha lin da LISTA. Para isso, ela deve usar a função anterior. O parâmetro lin indica qual é a linha da matriz LISTA que esta sendo usada como ponto de partida para atribuição do valor k. Ou seja, a partir da posição (x, y) indicada por lin e armazenada na matriz LISTA, deve-se percorrer a LISTA e identificar suas posições vizinhas, armazenando o parâmetro k tanto em MAT quanto na

terceira coluna de LISTA. Note que as posições que estão na LISTA e que receberam o valor k na terceira coluna devem ser agora testadas para verificar se também possuem vizinho na LISTA. Ou seja, o processo de marcar o valor k deve continuar enquanto houver alguma posição na LISTA que recebeu o valor k e ainda não teve os seus vizinhos testados. *Sugestão: para indicar que as posições marcadas já foram testadas, à medida que estas vão sendo testadas e marcadas (com algum inteiro k), você pode armazenar nelas algum valor negativo conveniente.*

Observações

1. Para imprimir a matriz, basta imprimir os números (não é necessário imprimir as linhas horizontais e verticais).

2. Adotaremos por convenção que a matriz de entrada sempre terá valores os de linha e coluna igual à 10 ($m = n = 10$). Tais valores deverão ser definidos com `#define` e com os nomes `MAX_LIN` e`MAX_COL`.

3. Neste programa você **deve** fazer a leitura de dados de um arquivo chamado *matriz.txt* e imprimir a tabela marcada em um arquivo com o nome *marcada.txt*. Existe um exemplo de entrada e saída utilizando arquivos externos no PACA. Segerimos que você faça o programa inicialmente sem usar este tipo de entrada (a menos que você já saiba), e só quando estiver tudo correto, implementar esta parte.

4. Você pode utilizar mais outras funções além das que foram mencionadas, mas alertamos que apenas estas são suficientes para um boa implementação. Lembre-se que uma das principais características de bom algoritmo é a simplificação do problema e a economia na utilização de recursos.

Bônus (opcional)

Se o seu programa também imprimir uma ilustração da matriz alagada utilizando caracters bonitinhos, o seu EP3 receberá uma nota adicional. Esta ilustração deverá ser impressa em um arquivo com o nome *ilustracao.txt*. Use a imaginação!

Instruções

Sobre a Elaboração

O EP pode ser elaborado por equipes de dois alunos, desde que as seguintes regras sejam respeitadas:

- Os alunos devem trabalhar sempre juntos, buscando a cooperação.
- Caso exista em um grupo um aluno com maior facilidade, este deve explicar as decisões tomadas, e o seu par deve participar e se esforçar para entender o desenvolvimento do programa: denominamos isso de *programação em pares*, uma excelente prática que vocês devem se esforçar para adotar.
- Mesmo a digitação do EP deve ser feita em grupo, enquanto um digita, o outro deve acompanha.
- Recomendamos fortemente que o exercício seja desenvolvido conforme a descrição nos ítens acima, mas ele também pode ser feito individualmente.

Sobre a Avaliação

- É sua responsabilidade manter o código do seu EP em sigilo, ou seja, apenas você e seu par podem ter acesso ao código.
- No caso de EPs feitos em dupla, a mesma nota será atribuída aos dois alunos do grupo.
- **Não serão toleradas cópias!** Exercícios copiados (com eventuais disfarces) levarão à reprovação da disciplina e ao encaminhamento do caso para a Comissão de Graduação.
- Exercícios atrasados não serão aceitos.
- Exercícios com erro de sintaxe (ou seja, erros de compilação) receberão nota ZERO.
- É muito importante que seu programa tenha comentários e esteja bem indentado, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa. Isto afetará a sua nota.
- Caso o programa apresente resultados "estranhos" (inesperados) para eventuais dados de entrada "incorretos", haverá desconto de nota.
- É importante que o exercício programa siga as instruções do enunciado e faça tudo da maneira que ele pede. Caso isso não aconteça, haverá desconto de nota.
- Uma regra básica é a seguinte: do ponto de vista do monitor responsável pela correção dos trabalhos, quanto mais convenientemente apresentado estiver o seu programa, melhor avaliado ele será.

Sobre a entrega

- O prazo de entrega é até o dia 30/06/2011.
- Caso feito em dupla: **Ambos devem submeter a versão final do código! Aquele que não submeter ficará com nota zero!**
- Entregar apenas um arquivo com o nome *alagamento.c* (Nome, letra minúscula e extensão *.c* não são opcionais).
- Para a entrega, utilize o Paca. Você pode entregar várias versões de um mesmo EP até o término do prazo, mas somente a última versão que permanecerá armazenada pelo sistema.
- Não serão aceitas submissões por e-mail ou atrasadas. Não deixe para a última hora, pois o sistema pode ficar congestionado, e você corre o risco de não conseguir enviar.
- Guarde uma cópia do seu EP pelo menos até o final do semestre.
- No início do arquivo, acrescente o seguinte cabeçalho:

```
/*****************************************************************/
/** MAC0110 BMAC IME Noturno 2011                               **/
/** Prof Roberto Cesar                                          **/
/**                                                             **/
/** Terceiro Exercicio Programa -- alagamento.c                  **/
/**                                                             **/
```

```
/** <nome do(a) aluno(a)> <numero USP>                                **/
/** <nome do(a) aluno(a)> <numero USP>                                **/
/**                                                                   **/
/** Informacoes sobre Desenvolvimento:                                **/
/**  <Ambiente (CodeBlocks, Gedit, Dev-C++, ...)>                     **/
/**  <Sistema Operacional (Windows, Linux, ...)>                      **/
/**                                                                   **/
/***********************************************************************/
```