Prova Final de Algoritmos e Estruturas de Dados I
08/07/2009

Um dos formatos para representação de imagens em tons de cinza é o PGM. Este formato, simplificadamente, consiste de um arquivo ASCII com o seguinte conteúdo:

- A primeira linha contém as letras "P2"
- A segunda linha contém a largura e a altura de uma matriz, isto é, o número de colunas (*col*) e de linhas (*lin*) da imagem.
- A terceira linha contém o maior valor da matriz.
- as linhas seguintes contém *col* números de $0 - 255$ representando o valor de cada ponto da imagem.

Nas questões que seguem, considere as seguintes estruturas de dados e assinaturas de funções e procedimentos:

```
const MAX=10000;

type
     matriz = array [1..MAX,1..MAX] of integer;

     vetor  = array [1..MAX] of integer;

     imagem = record
                  col, lin, maior: integer;
                  m: matriz;
     end;

     imgcompactada = record
                  tam: integer;
                  v: vetor;
     end;

function calcula_valor_medio (var I: imagem): integer;
(* funcao que retorna o valor medio dos pixels da imagem, isto eh
   a soma de todos os elementos dividido pelo numero de elementos *)

procedure ler (var I: imagem);
(* procedimento que le uma imagem no formato PGM *)

procedure imprime_imagem (var I: imagem);
(* procedimento que imprime uma imagem no formato PGM *)

procedure binariza (var I: imagem; limiar: integer);
(* procedimento que transforma a imagem de tons de cinza para preto e branco
   para isto, os pixels que forem maiores que o limiar devem se tornar brancos
   e os que forem menores ou iguais a este mesmo limiar devem se tornar pretos *)

procedure compacta_imagem (var I: imagem; var C: imgcompactada);
(* procedimento que recebe uma imagem no formato PGM e cria um vetor C
   que eh uma representacao compactada desta                              *)

procedure imprime_img_compactada (var C: imgcompactada);
(* procedure que recebe uma imagem compactada e a imprime no formato PGM *)
```

Questões

1. (25 pontos) Faça um programa principal em *free Pascal* que *use* as definições acima (não é para implementar as funções e procedimentos!) para receber um certo número N de imagens PGM em tons de cinza (onde 0 representa preto e o branco é representado pelo maior valor da imagem) e imprime a imagem binarizada, isto é, em preto e branco (onde 0 representa preto e 1 representa branco. Note que o limiar é obtido do valor médios dos pixels.

2. (25 pontos) Implemente a função *calcula_valor_medio* de maneira que ela faça exatamente o esperado, isto é, o que está escrito no comentário da função, acima.

3. (25 pontos) Implemente o procedimento *binariza* de maneira que ela faça exatamente o esperado, isto é, o que está escrito no comentário do procedimento, acima.

4. (25 pontos) Implemente um procedimento que gere um vetor que represente a matriz binarizada de forma compacta. Para isto, use a seguinte idéia: como a matriz só tem zeros e uns, vamos substituir seqüências de uns pelo número de uns consecutivos. Os elementos vão sendo colocados no vetor, de maneira linear, cada linha seguinte é concatenada à anterior. Veja o exemplo:

   Exemplo:

   - **Imagem binarizada**:

   ```
   P2
   11 10
   1
   1  1  1  1  0  1  1  1  1  1  0
   1  1  0  1  1  1  1  1  1  1  1
   0  0  1  0  0  0  1  1  1  0  0
   1  1  1  1  1  1  1  1  1  1  1
   0  0  0  1  1  1  1  1  0  0  0
   0  0  0  1  1  1  1  1  1  1  1
   1  1  1  0  1  1  1  1  1  1  1
   1  1  1  1  1  1  1  0  1  1  1
   1  1  1  1  1  1  1  1  1  0  0
   0  1  1  1  1  1  1  1  1  1  1
   ```

   - **Imagem compactada**:

   ```
   36
   4 0 5 0 2 0 8 0 0 1 0 0 0 3 0 0 11 0 0 0 5 0 0 0 0 0 0 11 0 14 0 12 0 0 0 10
   ```

   Isto é, a primeira linha da matriz possui 4 uns consecutivos seguido de um zero e outros 5 uns consecutivos, por isto, o vetor contém seus primeiros elementos "4, 0 e 5". Preste atenção antes de escrever o código. Você pode definir, se precisar, funções, procedimentos ou estruturas de dados adicionais.