Terceira Prova de Algoritmos e Estruturas de Dados I
26/02/2020

O que será avaliado? Especialmente nesta prova, a modularidade: uso de funções e procedimentos, passagem de parâmetros e uso de variáveis locais. Mas também contam: a clareza, a lógica, a criatividade, a sintaxe, o uso correto dos comandos, a correta declaração dos tipos, os nomes das variáveis, a indentação e o uso equilibrado de comentários no código. Evidentemente seu programa deve funcionar também!
Boa prova!!

1. Questão única (valor 100 pontos) Imagens no formato PGM ocupam bastante espaço. Escreva um programa *Pascal* para compactar imagens binárias no formato PGM. Imagens binárias são constituídas somentes por 0 ou 1 e mais nenhum outro valor. As regras de compactação são as seguintes:

   Cada sequência de 1's ou 0' consecutivos é chamado de segmento.

   O programa deve imprimir inicialmente o número de segmentos presentes na imagem; Na linha seguinte deve imprimir a quantidade de números em cada segmento de acordo com a seguinte regra:

   Devem ser impressos pares de números $[q_1, q_0]$ onde q_1 indica a quantidade de 1's e q_0 indica a quantidade de 0's. Por exemplo se o primeiro segmento tiver oito números 1 e o segundo segmento tiver três números 0, então deve ser impresso 8 3. Porém se o primeiro segmento tiver oito números 0, então deve ser impresso 0 8. A mesma regra se aplica o último par de números: o último número obrigatoriamente representa o segmento com a quantidade de zeros. Se a matriz binária terminar em um segmento composto por 5 números 1, então deve ser impresso 5 0.

   Para a resolução desta questão, você deve utilizar o tipo `imagem_pgm` definido abaixo para armazenar as informações da imagem.

```
type imagem_pgm = record
  pixel : array[1..MAX_LINHAS ,1..MAX_COLUNAS] of  integer;
  num_linhas, num_colunas, maximo : integer;
end;
```

   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 (quantidade de 1's e 0's em sequência):

```
24
4 1 5 1 2 1 8 2 1 3 3 2 11 3 5 6 11 1 14 1 12 3 10 0
```