Lista 5 - CI055 - Algoritmos e Estruturas de Dados I

A lista abaixo deve ser resolvida de maneira individual. As soluções dos exercícios deverão ser discutidas e entregues seguindo as instruções da página da disciplina:

- Alunos da turma do **Prof. Marcos Castilho** devem consultar:
  `http://www.inf.ufpr.br/alexander/ci055/instrucoes-turma-marcos.html`
- Alunos da turma do **Prof. Daniel Weingaertner** devem consultar:
  `http://www.inf.ufpr.br/alexander/ci055/instrucoes-turma-daniel.html`

O prazo para entrega desta lista é: **31 de Maio de 2015 (domingo).**

Enunciados dos exercícios

1. Escreva um programa completo na linguagem Pascal capaz de realizar a intercalação de duas listas de valores reais lidos do teclado. O programa deve conter um procedimento chamado `intercala`. Seus parâmetros são três vetores (e seus tamanhos): `Va`, `Vb` e `Vc`. Todos são vetores de valores *Reais*, sendo que `Va` e `Vb` possuem uma quantidade arbitrária de valores lidos do teclado (o zero é o último valor de cada lista de entrada). O procedimento deve intercalar os valores de `Va` com os de `Vb` e colocar o resultado em `Vc`. A intercalação é uma operação de montagem de uma lista com elementos que foram escolhidos alternadamente de duas outras listas. Por exemplo, suponha que os valores do vetor `Va` recebidos como parâmetro sejam os seguintes:

   ```
   1.0  2.0  3.0  4.0  5.0  6.0  7.0  0 <ENTER>
   ```

   Suponha também que os valores do vetor `Vb` sejam os seguintes:

   ```
   11.0  12.0  13.0  8.0  0 <ENTER>
   ```

   O procedimento `intercala` deve retornar os seguintes valores por meio do vetor `Vc` a quem o ativar:

   ```
   1.0  11.0  2.0  12.0  3.0  13.0  4.0  8.0  5.0  6.0  7.0
   ```

   Atenção: O procedimento `intercala` deve ser projetado para imprimir nenhum valor na saída padrão. Realize a impressão por meio de outro módulo.

2. Escreva um programa completo na linguagem Pascal para ler uma sequência de inteiros positivos e imprimir quantas vezes cada valor aparece na sequência. A sequência contém no máximo 100 inteiros e termina quando o valor 0 é digitado. Veja um exemplo de execução:

   ```
   Entre com uma sequencia de inteiros terminada por zero:
   104  8  9  12  8  73  104  9  8  1001  0  <ENTER>
   A sequencia contem:
   104: 2 ocorrencias
   8: 3 ocorrencias
   9: 2 ocorrencias
   12: 1 ocorrencia
   73: 1 ocorrencia
   1001: 1 ocorrencia
   ```

   Sugestão: escreva a função abstrata `busca( vet, tam, num )` que retorna o índice onde se encontra o valor `num` no vetor `vet` de `tam` elementos. Caso `num` não exista, a função retorna um código de erro (por exemplo, -1).

3. Fazer um programa em Pascal para preencher uma matriz de elementos inteiros com m colunas por n linhas. O valor de cada célula poderá ser um número que vai de 0 (zero) até max, onde max será definido como uma constante inteira do programa em Pascal, assim como m e n também o serão. Na primeira versão do programa, m deverá valer 210, n deverá valer 140 e max deverá ser 29399, assim:

```
   ...
const
  m = 210;
  n = 140;
  max = 29399;
   ...
```

O preenchimento dos elementos da matriz durante a leitura deverá ser feito por linha (não por coluna), utilizando apenas um dos modos relacionados abaixo para a entrega da lista (o aluno poderá tentar os outros por conta própria pois são todos bem simples):

- preenchimento serial $(0, 1, 2, 3, \ldots, 29399)$;
- preenchimento aleatório (função `random` para valores entre 0 e 29399);
- alguma função cíclica, cujos valores nunca ultramassam 29399 (exemplo: 29399 * sen(x), com x variando de 0 a 29399, mas use as funções trunc e abs do Pascal para tansformar a saída da função seno em número inteiro e ainda transformar números negativos em positivos);
- outras possibilidades interessantes de sua própria escolha.

O formato da impressão do programa deverá ser exatamente o mesmo de uma imagem codificada no pradrão PGM (Portable Gray Map), o qual terá $m + 3$ linhas. A primeira linha só deverá possuir dois caracteres: P2. A segunda linha deverá ter dois inteiros: a quantidade de colunas (n) da matriz seguido da quantidade de linhas (m). A terceira linha possui apenas o valor da constante max, a qual representa o valor da máxima intensidade de luz em uma análoga escala de cinza cujos valores variam de 0 a max. Veja um exemplo de impressão se o programa tivesse n = 24 colunas, m = 7 linhas e max = 15.

```
./geraimagem | less
P2
24 7
15
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0  3  3  3  3  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0 15  0
0  3  3  3  0  0  0  7  7  7  0  0  0 11 11 11  0  0  0 15 15 15 15  0
0  3  0  0  0  0  0  7  0  0  0  0  0 11  0  0  0  0  0 15  0  0  0  0
0  3  0  0  0  0  0  7  7  7  7  0  0 11 11 11 11  0  0 15  0  0  0  0
0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

Apenas a título de curiosidade, é possível visualizar a imagem PGM gerada pelo programa usando o utilitário display, assim:

```
./geraimagem | display &
```

4. Fazer um programa em Pascal para ler dois números inteiros que representam, respectivamente, a quantidade m de linhas e n de colunas de uma matriz. Em seguida, devem ser lidos os $m \times n$ elementos da matriz, os quais são números reais, entrados pela ordem das linhas da matriz. Depois da leitura, o programa deve decidir e imprimir se ela é ou não uma *Matriz de Vandermonde*. Se for, imprimir `Sim`. Se não for, imprimir `Nao`.

   Pesquise sobre *Matriz de Vandermonde* em:

   `http://pt.wikipedia.org/wiki/Matriz_de_Vandermonde`

   De forma resumida, uma *Matriz de Vandermonde* é aquela que segue a forma geral abaixo:

$$\begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \ldots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \ldots & \alpha_2^{n-1} \\ 1 & \alpha_3 & \alpha_3^2 & \ldots & \alpha_3^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_m & \alpha_m^2 & \ldots & \alpha_m^{n-1} \end{bmatrix}$$

   Veja um exemplo de execução do programa abaixo:

```
./executa
5 4 <ENTER>
1  2   4   16 <ENTER>
1  3   9   81 <ENTER>
1  4  16  256 <ENTER>
1  5  25  625 <ENTER>
1  6  36 1296 <ENTER>
Nao
```

Veja outro exemplo:

```
./executa
5 4 <ENTER>
1  2   4    8 <ENTER>
1  3   9   27 <ENTER>
1  4  16   64 <ENTER>
1  5  25  125 <ENTER>
1  6  36  216 <ENTER>
Sim
```