**UNISINOS - UNIVERSIDADE DO VALE DO RIO DOS SINOS**

**Curso:** ***Informática*** **Disciplina:** ***Laboratório 2*** **Horário: 41**
**Prof: Fernando Osório** **E-mail: osorio@exatas.unisinos.br** **Data:19/05/04**

**Nome do Aluno: _____**
**Nro. de Matrícula: _____ - ___**

## TESTE - GRAU B

***3.0 Pontos***

**Questão única:**

Faça um programa de manipulação de **árvores binárias ordenadas** com alocação dinâmica, para armazenar uma coleção de palavras contidas em um arquivo texto, junto com a indicação do número de vezes que cada palavra aparece no texto (gera uma "tabela arborescente" organizada em ordem alfabética e contendo o número ocorrências de cada palavra). Implemente o programa de acordo com a especificação dada a seguir: **ler arquivo - criar árvore - exibir - remover nodos - salvar em disco**

- O programa deve usar uma rotina fornecida pelo professor "**prox_dado**" para obter as palavras, uma por uma, do arquivo texto indicado pelo usuário. Esta rotina retorna apenas as palavras do texto, ignorando caracteres especiais e de pontuação, eliminando os acentos e retornando as palavras convertidas para maiúsculo. As palavras devem ser inseridas de forma ordenada (ordem alfabética) na árvore, com o devido controle do número de vezes que cada palavra ocorre no texto. Exemplo de um arquivo de texto a ser lido: ("arq.txt" – disponível em ***http://inf.unisinos.br/~osorio/txtproc/*** )

**Arq.txt**

```
Program Exemplo;
Var
  a,b:Integer;
  idade:Integer;
  sexo:Char
  Nome:String;
Begin
  writeln('Nome: ');
  readln(nome);
  writeln('Idade: ');
  readln(idade);
  writeln('Sexo: ');
  readln(sexo);
  if (sexo='M') or (sexo='m')
  then writeln('Hello Mr.',nome,' ',idade)
  else writeln('Hello Mr.',nome,' ',idade);
  readln;
end.
```

- Após terminar de criar a estrutura de dados em memória, **exibir na tela os dados** armazenados na árvore em modo **VED (pré-fixado) com identação** de modo que possamos verificar como foram inseridos os nodos (como estão distribuídos na árvore). A cada nível mais abaixo da árvore, os dados devem ser identados de mais 1 espaço em branco. Vide exemplo do programa executável e de como este exibe os dados de modo identado, disponível em ***http://inf.unisinos.br/~osorio/txtproc/***

- Após exibir a árvore, **remover** todos os **nodos** da árvore que possuam palavras que ocorrem uma única vez no texto. Os nodos devem ser removidos da árvore de modo correto, liberando a área de memória corretamente. [Se você não conseguir fazer isto, siga adiante]

- **Salvar em disco** as palavras, com o seu respectivo número de ocorrências (arquivo "**refs.txt**"), em ordem alfabética. Para o exemplo descrito anteriormente (arq.txt) os dados a serem gerados e gravados no arquivo devem ficar como é apresentado no exemplo a seguir: (tabela da esquerda – sem remoção, tabela da direita – com remoção)

**Refs.txt (sem remoção)**

```
A - 1
B - 1
BEGIN - 1
CHAR - 1
ELSE - 1
END - 1
EXEMPLO - 1
HELLO - 2
IDADE - 5
IF - 1
INTEGER - 2
M - 2
MR - 2
NOME - 5
OR - 1
PROGRAM - 1
READLN - 4
SEXO - 5
STRING - 1
THEN - 1
VAR - 1
WRITELN – 5
```

**Refs.txt (com remoção)**

```
HELLO - 2
IDADE - 5
INTEGER - 2
M - 2
MR - 2
NOME - 5
READLN - 4
SEXO - 5
WRITELN - 5
```

- A rotina ***prox_dado***, se comportará da seguinte maneira: cada vez que for chamada ela irá retornar uma nova palavra do texto, desprezando certos elementos sem importância. A rotina prox_dado possui dois parâmetros: Arq e Dado; sendo que esta rotina retorna um valor booleano. Veja a definição da rotina dada abaixo:

***Function Prox_Dado (var Arq:Text; var Dado: String): Boolean;***

***Arq***: Variável do tipo arquivo Texto. O arquivo é passado como parâmetro e já deve estar aberto antes de ser chamada a rotina, de forma que ele possa ser lido dentro dela.

***Dado***: Esta variável retorna a palavra que foi lida do arquivo texto. Alguns caracteres do arquivo original serão ignorados propositalmente, os acentos são eliminados e as palavras são convertidas sempre para maiúsculo (para não afetar a ordenação).

***Valor Retornado:*** TRUE = Indica que foi lida uma palavra, onde a palavra lida está armazenada na variável Dado. FALSE = Indica que não foi possível ler uma nova palavra do arquivo pois este arquivo terminou (End-Of-File).

**PONTO EXTRA**: O aluno que implementar o programa acima usando uma árvore binária ordenada do tipo **BALANCEADA** ganhará um ponto extra (0.5 pts sobre a nota da prova).

***Bom trabalho!***

## Atenção:

- O programa acima deve usar rotinas de manipulação de estruturas de dados tipo árvores com alocação dinâmica similares as rotinas vistas em aula (programa modular). Evite usar variáveis globais no programa!
- O programa que faz a leitura do arquivo texto, fornecendo as palavras, uma por uma, pode ser encontrado na Web, junto com arquivos exemplo: http://www.inf.unisinos.br/~osorio/txtproc
- Lembre-se de colocar o seu nome e número de matrícula como comentários nas primeiras linhas do programa fonte no arquivo teste-gb.pas. Envie por e-mail ao professor (e-mail indicado no cabeçalho da prova) com subject "Teste GB". Durante a prova está proibido o envio de mensagens.