| | |
|---|---|
| UFV Campus Florestal | **Universidade Federal de Viçosa**<br>**Campus Florestal**<br>**CCF212 - Algoritmos e Estruturas de Dados II**<br>**2020/1 PER** |

## 1º Trabalho Prático de CCF212 (Valor: 20pts)

## ÁRVORES DIGITAIS

**Formação do grupo:** O trabalho deverá ser desenvolvido pelos grupos já formados na disciplina.

**Problema: Comparação entre árvores TRIE TST e PATRICIA**

Implementar os TADs Árvore TRIE TST e PATRICIA para armazenar palavras de um texto ou dicionário, para fins de comparação em termos do custo computacional das operações de inserção e pesquisa.

**Tarefas:**

1. Criar uma árvore TRIE (TST) e uma árvore PATRICIA para armazenar as palavras de um texto ou dicionário à escolha do grupo. No caso de se usar o texto, recomenda-se transformar as palavras para minúsculas antes da inserção nas árvores. Recomenda-se ainda usar dicionário e/ou texto em inglês para evitar acentuação.
2. Para a árvore PATRICIA, adaptar os algoritmos fornecidos em sala de aula para permitir o armazenamento de palavras. A solução mais comum é inserir mais um campo de comparação em cada nó, ou seja, além do campo de índice (que avança x posições na palavra) será necessário também ter um campo com o caracter que está sendo comparado naquela posição para se decidir o caminho a seguir (esquerda ou direita). A decisão de se colocar, no nó interno, o menor ou o maior caracter de comparação e se os iguais ficarão à esquerda ou à direita deste nó, deverá levar em conta o melhor uso de memória e a diferença de tamanho entre as palavras.
3. Cada árvore conterá apenas uma ocorrência de cada palavra inserida, não sendo permitida a inserção de palavras iguais nem sendo necessário registrar o número de ocorrências das palavras no texto (quando for o caso).
4. A estrutura da árvore TRIE TST está diretamente relacionada às primeiras chaves inseridas. Portanto, para o armazenamento de chaves de um dicionário, os testes devem considerar **cenários** onde as chaves são inseridas também em ordem aleatória e não apenas em ordem alfabética. Replicar os cenários para PATRICIA, para fins de comparação.

5. Usar **cenários** que permitam avaliar a escalabilidade (falta de) da árvore PATRICIA para palavras com mesmo radical e que se diferem em um mesmo caracter. Para o caso de se usar o texto como fonte de dados, inserir palavras extras que permitam esse teste já que a probabilidade dessas palavras estarem no texto, desta forma, é baixa. As mesmas inserções devem ser feitas na TRIE TST para fins de comparação.
6. Os testes objetivam comparar as estruturas quanto aos seguintes parâmetros e cenários: **uso de memória, tempo de execução e número de comparações para as operações de inserção (Palavra JÁ EXISTE ou NÃO) e pesquisa (COM e SEM sucesso)**.
7. Todos os testes devem ser realizados em uma mesma máquina, cuja configuração deve ser informada no relatório.
8. Elaborar um relatório sintético, contendo: uma introdução, com o objetivo do trabalho e as principais fontes (referências) dos algoritmos utilizados; uma seção de desenvolvimento, com os detalhes da implementação, os testes realizados e a análise dos resultados; e uma seção de considerações finais.
9. Na subseção de implementação, explicar em linhas gerais os algoritmos utilizados e as adaptações realizadas para as operações de inserção e pesquisa, podendo inserir alguns pequenos trechos de código na explicação. Demais detalhes de implementação das estruturas devem ser documentados no próprio código.
10. Os testes devem ser apresentados em tabelas e gráficos (quando aplicável), agrupados segundo cada cenário utilizado e que permitam a comparação entre as estruturas.
11. O menu de opções do programa deverá conter as seguintes opções: 1) escolher árvore, 2) inserir palavra, 3) pesquisar palavra, 4) exibir todas as palavras em ordem alfabética e 5) contar palavras. Para as operações de inserção e pesquisa, exibir uma mensagem com o número de comparações realizadas.

## Entrega:

- O trabalho deverá ser entregue via PVANET, pelo líder do grupo, através de um **único** arquivo compactado, **com o nome do grupo**, contendo:
  - código-fonte do programa em C;
  - arquivos utilizados como entrada (dicionário e/ou textos para testes);
  - arquivo “leiame.txt” com explicações de uso do programa.
- **Data de entrega (PVANet): 26/10/20**
- **Data de apresentação (Google Meet): 27 e 29/10/20**

# Universidade Federal de Viçosa
# Campus Florestal
# CCF212 - Algoritmos e Estruturas de Dados II
# 2020/1 PER

## Comentários Gerais:

- O grupo deverá tomar como base os códigos discutidos em aula, ou similares, desde que estejam coerentes com a fundamentação teórica das estruturas. Os códigos de inserção e consulta podem ser modificados, a critério do grupo, desde que os algoritmos (em essência) sejam mantidos;
- O grupo deverá ser identificado no cabeçalho de TODOS os arquivos do código-fonte;
- O código-fonte DEVERÁ ser devidamente comentado;
- As implementações relativas a cada TAD devem estar em arquivos separados;
- As operações referentes à leitura e carga dos dados devem estar em um arquivo separado;
- Atenção quanto ao uso e inicializações de variáveis no programa principal, que podem comprometer o funcionamento do seu código (encapsular funções sempre que possível);
- Apesar de o trabalho ser em grupo, a nota poderá ser individualmente atribuída, a critério da professora (entrevistas individuais poderão ser realizadas);
- Em caso de plágio entre trabalhos, será atribuída nota zero para todos os envolvidos (dos grupos em questão) e atribuição de conceito **F**. Se houver discussões entre os grupos acerca de soluções para questões específicas dos algoritmos, não há problema, desde que isso esteja devidamente documentado no relatório e no código-fonte (na função correspondente).
- Trabalhos entregues **em atraso** ou que **não sejam apresentados** pelo grupo receberão nota ZERO.
- Durante o desenvolvimento do trabalho, caberá ao grupo propor e construir uma implementação para o problema apresentado. A professora não analisará erros em código-fonte, nem tampouco fornecerá detalhes técnicos da solução a ser construída.