UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

SSC0603 – Estrutura de Dados I – 2025-2 (ED1)

Professor responsável: *Fernando Santos Osório*
Semestre: 2025/2
Horário: Terça 13h/14h (Prova P2)

Wiki: SSC-603-2025(FOsorio)
Web: ***http://www.icmc.usp.br/~fosorio /***
DATA PROVA: 09/12/2025 - PROVA P2-FINAL

NRO. USP: **< Colocar o seu NUSP no programa fonte>** COMO COMENTÁRIO no CÓDIGO

NOME : **< Colocar o seu Nome no programa fonte>** COMO COMENTÁRIO no CÓDIGO

>> COLOCAR SEU NOME E NRO. USP COMO COMENTÁRIO DO PROGRAMA FONTE ENTREGUE!

PROVA PROVA P2 FINAL – SSC0603 – ED1

Q1 (Questão única). Implemente o seguinte programa descrito abaixo de acordo com o especificado. O programa deve ser enviado para o RunCodes da Disciplina SSC0603 (Prova P2 FINAL)
RunCodes SSC0603 - https://runcodes.icmc.usp.br/offerings/view/123 - Regras da Prova no final deste texto.

O programa da prova é a implementação de uma lista de palavras para o "autocompletar", conforme descrito abaixo, baseado no USO DE ÁRVORES BINÁRIAS ORDENADAS E BALANCEADAS (TAD visto em aula ou usando outro TAD qualquer desde que tenha alocação dinâmica de nodos, a árvore seja binária, seja ordenada e seja balanceada).

Descrição: Criar um programa "modular" (com sub-rotinas, conforme descrição mais abaixo), que realize as seguintes tarefas:

(1) Ler um arquivo texto de entrada que possui uma lista de palavras (words list). O arquivo de palavras será denominado de "words10000.txt". O formato do arquivo é o seguinte: sequência de linhas com 1 palavra em cada linha do arquivo (que contém 10.000 palavras). O arquivo termina por uma "marca" especial, a palavra "$END" ('$' como o primeiro caracter da linha final).

Veja um exemplo de partes do arquivo "words10000.txt" abaixo:

```
...
ac
academic
academics
academy
acc
accent
accept
acceptable
acceptance
accepted
...
zum
zus
$END
```

Você pode ler o arquivo usando fscanf ou fgets, como achar melhor. É um arquivo texto, sem acentos e de palavras em inglês (em minúsculas).

(2) Criar uma árvores binária ordenada e balanceada com as palavras lidas do arquivo. A árvore, AVL, RedBlack ou outra árvore balanceada, deve armazenar em seus nodos as palavras.

Lembre-se de incluir a biblioteca de STRINGS do "C" e de usar sempre as funções de manipulação de strings, como a strcpy, strcmp para copiar e comparar as strings ao inserir nos nodos da árvore.

(3) Criar uma função que busque palavras com o mesmo prefixo (caracteres iniciais iguais), por exemplo: `academic - academics - academy` – considerando o **prefixo "aca" e 3 letras**.
A função deve receber o prefixo e o número de letras a considerar, onde usualmente serão indicadas pelo menos 3 letras. Sugere-se o uso da função **strncmp** para achar strings com o prefixo indicado.
A busca deve ser feita de forma OTIMIZADA, ou seja, NÃO DEVE SER UMA BUSCA EXAUSTIVA em toda árvore (busca binária), deve ir direto aos nodos relevantes, considerando que estão ordenados.

(4) Ler do teclado as seguintes informações:

```
    <int> - Inteiro que indica 0 ou 1
            0 se for só para achar as palavras com o prefixo indicado
              na ordem em que se encontram na árvore;
            1 se for para achar as palavras e exibir de modo ordenado,
              e eficiente (o mais eficiente é usualmente criando outra
              árvore binária ordenada e balanceada só com essa palavras.
    <int> - Tamanho do prefixo, usualmente 3 caracters (tamanho 3)
    <prefixo> - Sequência de letras que definem o prefixo procurado
                Por exemplo: "aca"

Exemplo de dados lidos do teclado:
     0
     3
     aca
Ssída na tela: (não ordenada)
     academics
     academy
     academic

Exemplo de dados lidos do teclado:
     1
     3
     aca
Ssída na tela: (ordenada)
     academic
     academics
     academy
```

Em resumo, o programa lê o arquivo, cria uma árvore balanceada inicial, busca palavras que possuam o mesmo prefixo (especificado por entrada pelo teclado), e exibe a lista destas palavras na tela. Se a opção for exibir ordenado, cria uma nova árvore balanceada com as palavras encontradas, ordenando e exibindo elas de modo ordenado na tela.

Algumas dicas...

STRINGS
Lembre-se que você está usando TEXTOs, onde é importante usar a biblioteca “string.h”
Lembre-se que strings são copiadas com a “strcpy” (não pode atribuir direto!)
Lembre-se que a comparação de strings deve ser feita com a função “strcmp” e esta função retorna: 0 se as strings são exatamente iguais; e um valor < 0 ou > 0 se uma string “antecede” ou “sucede“ a outra na ordenação. É como com a ordem numérica, mas considerando strings.
https://www.man7.org/linux/man-pages/man3/strcmp.3.html (menor, igual ou maior na comparação)
Lembre-se que o strncopy compara somente as “n” primeiras letras, pode ser muito útil!

Busque usar recursos de DEBUG que ajudam ver o que pode estar certo ou errado no programa.

BOA PROVA!!!

REGRAS EM RELAÇÃO REALIZAÇÃO DESTA PROVA

1. A PROVA É **INDIVIDUAL com consulta Papel (Livros, papel impresso ou escrito) e formato Digital (Internet, Wiki, Pendrive), porém SEM CONSULTAR ou SE COMUNICAR COM HUMANOS ou QUALQUER OUTRA FORMA DE VIDA TERRESTRE ou EXTRA-TERRESTRE!**
   Não podem usar de formas de comunicação com pessoas externas, além do professor, referente a prova, seja por celular (manter desligado/guardado), por e-mail, por whatsapp, por mensagens ou fóruns (“ao vivo”), com colegas, etc.
   NÃO É PERMITIDO O EMPRÉSTIMO DE MATERIAL (Cadernos, Anotações, Livros, PenDrive, etc).

2. **A PROVA TEM QUE SER FEITA NO LABORATÓRIO 8.113 / 8.115 USANDO OS COMPUTADORES PCs DO LABORATÓRIO (SEM VPN, SEM ACESSO REMOTO!)**
   O ENDEREÇO DE IP USADO (Endereço de rede) PARA O ENVIO DA PROVA TEM QUE SER DAS MÁQUINAS DO LABORATÓRIO 8.113/8.115
   >> As máquinas do Lab. 8.113 e 8.115 estão sendo monitoradas << Sniff, sniff 😉

3. A PROVA DEVE SER **ENTREGUE via RunCodes.icmc (pode submeter várias vezes)**
   ENVIAR O ARQUIVO ZIP do MAKEFILE – O envio de “.c” é “aceito”, mas tem *desconto* na nota.
   Entrega é via https://runcodes.icmc.usp.br/ Disciplina SSC0603 – Prova P2 Final

4. **EM CASO DE PROBLEMAS no RunCodes.icmc pode ser entregue por E-Mail** para:
   fosorio @ icmc.usp.br com **cópia (Cc:)** para fosorio @ gmail.com
   Com assunto/subject: **Prova-P2 <Seu_Nome> <Nro_USP>**
   ANEXANDO O PROGRAMA FONTE (Zip do Makefile ou se não conseguir, o “.C” e o Project “.CBP”)
   NÃO ANEXAR EXECUTÁVEIS! JAMAIS ENVIE EXECUTÁVEIS POR E-MAIL
   NÃO ANEXAR OBJ, BIN e EXE!!! Seu e-mail não será recebido se for anexado bin, obj ou exe!

5. A PROVA USUALMENTE DEVE SER REALIZADA USANDO O CODEBLOCKS DO LABORATÓRIO, e depois enviada (testada com os casos de teste) do RunCodes (USUAL).

6. **AO ENTREGAR A PROVA NO RUNCODES ou POR E-MAIL O(A) ALUNO(A) CONCORDA COM ESTAS REGRAS E SE COMPROMETE A FAZER A PROVA INDIVIDUALMENTE!**

FIM