Universidade Federal de Santa Maria
Centro de Tecnologia
Departamento de Computação Aplicada
Curso de Engenharia da Computação
ELC1061 – Estrutura de Dados
Prof. Bruno Crestani Calegaro

Trabalho 3 – Árvore de diretório

Objetivos

Criar um programa na linguagem de programação C capaz de manipular **diretórios e arquivos** com o uso de **árvores**.

Descrição

Implemente uma **árvore de diretório** para simular uma linha de comando. O programa deve ler um arquivo in.txt contendo uma lista de pastas e arquivos e montar uma árvore para armazenar essas informações. Ou seja, dado um nó A da árvore, listas todas as subpastas e arquivos desse nó. Deve ser possível navegar pelos diretórios através da árvore. O arquivo deve seguir a seguinte sintaxe:

```
Arquivos e Programas/Firefox
Arquivos e Programas/Chrome
Arquivos e Programas/Opera
Meus Documentos/apresentacao.ppt
Meus Documentos/relatorio.doc
Meus Documentos/fontes
Meus Documentos/fontes/main.c
Meus Documentos/fontes/main.h
Meus Documentos/imagens
Meus Downloads/7zip.exe
Meus Downloads/t2.rar
```

Note que '/' separa os valores e arquivos sempre contém uma extensão (.doc, .c, etc).

O programa deve oferecer uma interface do tipo linha de comando para o usuário executar operações como:

- **cd** : entrar em uma pasta
- **search** : busca uma pasta ou arquivo
- **rm** : remover uma pasta
- **list** : lista os componentes dentro da pasta em questão
- **mkdir** : cria uma nova pasta
- **clear** : limpa o contéudo da tela
- **help** : exibe a relação completa dos comandos
- **exit** : fechar o programa

Universidade Federal de Santa Maria
Centro de Tecnologia
Departamento de Computação Aplicada
Curso de Engenharia da Computação
ELC1061 – Estrutura de Dados
Prof. Bruno Crestani Calegaro

Especificação

O programa deve impreterivelmente cumprir o seguintes quesitos:

- Ler um arquivo **in.txt**
  - O arquivo deve conter a lista de pastas e arquivos
- Executar os comandos:
  - *cd <diretório>*
    - entra no diretório especificado se ele existir
    - se ele não existe,
      - então imprimi as possíveis alternativas, Ex: diretório = "Me" deve informar que existe um diretório "Meus Documentos" e "Meus Downloads"
      - senão existe alternativas então imprimi "Diretório não encontrado"
  - *search <arg>*
    - busca um arquivo ou pasta pelo seu nome "arg" e informa a sua localização
  - *rm <diretório>*
    - remove um pasta e seus arquivos, deve fazer uma liberação recursiva
  - *list*
    - lista todos os componentes dentro da pasta atual
  - *mkdir <arg>*
    - cria uma pasta com o nome "arg" na pasta atual
  - *clear*
    - limpa o conteúdo da tela imprimindo diversas novas linhas com **printf** ou usando a chamada de sistema "**clear**" ou "**cls**"
  - *help*
    - comando personalizado que deverá explicar quais comandos o programa possui, modo de uso e sua finalidade
  - *exit*
    - encerra o programa mas primeiro deve liberar o espaço alocado
- **Não enviar um programa com apenas um arquivo de código fonte.**

Datas de entrega e apresentação

Os alunos deverão entregar o código fonte até a data limite. A apresentação será em sala de aula em computador pessoal do próprio aluno ou ao oferecido pelo professor. Para a entrega do código fonte o aluno deverá enviar um e-mail para calegaro@inf.ufsm.br com o título "**T3 – ED**" e anexar um arquivo compactado, com o nome do aluno, contendo somente os arquivos da especificação.

- Data de Entrega e Apresentação: 29/07

Avaliação

- **Documentação**: Um bom código fonte contém uma série de comentários para ajudar na compreensão do arquivo. Detalhes específicos merecem uma especial atenção, mas não comente por exemplo o que faz i++.
- **Pontualidade**: Trabalhos atrasados valeram menos nota, tendo sua nota reduzida em até 3 pontos, 0,5 por dia de atraso.
- **Clareza**: envolve a facilidade de compreensão bem como nome de variáveis, funções e os próprios algoritmos. Evite códigos complexos e desnecessários, adote a solução mais simples possível.
- **Funcionalidade**: o programa deve satisfazer todos os requisitos expressos na especificação. Se o programa não compilar ou não atender nenhum requisito recebera nota 0 (zero).