Universidade do Vale do Itajaí
Disciplina: Sistemas Operacionais
Avaliação M3 – Sistemas de Arquivos

Instruções

1. Esta avaliação deve ser feita em dupla ou em trio.

2. Data de entrega: 01/07/2025 até as 19:00. Não pode entregar em atraso.

3. Esta avaliação tem por objetivo consolidar o aprendizado sobre conceitos de escalonamento em sistemas operacionais.

4. A implementação deverá ser desenvolvida utilizando a linguagem C/C++ devido a natureza do código fonte de referência.

5. O sistema deve ser entregue funcionando corretamente. Sistemas não compilando e executando não serão aceitos. É de responsabilidade do(s) aluno(s) apresentar a execução funcionando corretamente.

6. Deve ser disponibilizado os códigos da implementação em repositório do(s) aluno(s) (github, bitbucket, etc...), deve ser fornecido o link e o repositório dever ser público.

7. O relatório, em formato PDF, deve seguir o formato de artigo científico ou seguindo as normas da ABNT e as orientações para produção de trabalhos acadêmicos da Univali contendo:

    - Identificação do autor e do trabalho.
    - Enunciado dos projetos.
    - Explicação e contexto da aplicação para compreensão do problema tratado pela solução.
    - Resultados obtidos com as simulações.
    - Códigos importantes da implementação.
    - Resultados obtidos com a implementação (tabelas, gráficos e etc).
    - Análise e discussão sobre os resultados finais.

8. Deve ser disponibilizado os códigos da implementação juntamente com o relatório (salvo o caso da disponibilidade em repositório aberto do aluno, que deve ser fornecido o link). O repositório deve estar aberto do momento da entrega em diante, sendo que o professor não responsabiliza caso o projeto não esteja disponível para consulta no momento da correção, sendo do(s) aluno(s) essa responsabilidade de manter disponível. Cópias entre alunos implicará em nota zero para todos os envolvidos.

9. O trabalho deverá ser apresentado por meio de um vídeo gravado onde todos os alunos participantes do trabalho deverão apresentar e explicar a estrutura do código e funcionamento/lógica implementada do código. Na gravação deverá ser mostrado o código e o mesmo explicado (a lógica de funcionamento) bem como a compilação e execução do código. É de responsabilidade do(s) aluno(s) explicar os conceitos, comandos, bibliotecas usadas. É de responsabilidade do(s) aluno(s) fazer a solução funcionar. Trabalhos não apresentados terão como nota máxima 5,0, além dos descontos aplicados no restante da avaliação da implementação.

10. O vídeo de apresentação deverá ser postado junto do código e relatório e é de responsabilidade dos alunos verificar se o link do vídeo está funcional (drive, youtube, onedrive), se consegue ser acessado e também se a qualidade visual e de áudio está adequada para verificação e avaliação do professor. Se necessário, o professor poderá solicitar explicações adicionais presenciais.

Descrição do projeto a ser desenvolvido

Implementação de um Sistema de Arquivos Simples Utilizando Árvore B

Com o crescente volume de dados e a necessidade de estruturas eficientes para armazenamento e recuperação de informações, o estudo de sistemas de arquivos se torna essencial no campo da computação. Os sistemas de arquivos são responsáveis por organizar, armazenar e gerenciar o acesso aos dados em dispositivos de armazenamento, sendo componentes fundamentais de qualquer sistema operacional moderno.

Este trabalho tem como objetivo projetar e implementar um sistema de arquivos simplificado utilizando a estrutura de dados árvore B, conhecida por sua eficiência em operações de busca, inserção e remoção. A proposta envolve a construção de um sistema capaz de simular a criação, exclusão e organização hierárquica de arquivos do tipo .txt e diretórios, permitindo operações básicas de navegação e listagem.

A escolha da árvore B como estrutura base se deve à sua ampla utilização em sistemas reais de banco de dados e arquivos, dada sua capacidade de manter os dados ordenados e garantir desempenho balanceado mesmo com grandes volumes de inserções e exclusões. Neste projeto, a árvore B é utilizada para organizar os elementos (arquivos e pastas) de forma eficiente dentro de cada diretório.

O sistema desenvolvido inclui uma interface simples para simular comandos de manipulação de arquivos, e gera uma imagem textual (fs.img) representando a estrutura resultante do sistema de arquivos criado. Através desta abordagem, é possível compreender os fundamentos teóricos das estruturas de dados aplicados a um contexto prático de sistemas operacionais, reforçando a importância da abstração e da organização lógica no armazenamento de dados.

Descrição Geral

Nesse trabalho, vocês deverão desenvolver um sistema de arquivos virtual utilizando uma estrutura de dados baseada em árvore B, que permita:

- Inserção e exclusão de arquivos .txt
- Criação e remoção de pastas
- Armazenamento hierárquico de diretórios e arquivos

Como requisitos funcionais deste projeto, o sistema deverá ser capaz:

- RF1: Inserção de Arquivos:
  - Inserir arquivos .txt com conteúdo textual (limitado a 1MB).
  - Nome do arquivo deve ser único dentro de um diretório.
- RF2: Exclusão de Arquivos:
  - Permitir a remoção de arquivos .txt por nome.
- RF3: Criação de Pastas:
  - Criar diretórios, podendo conter subdiretórios e arquivos.
- RF4: Remoção de Pastas:
  - Remover diretórios (somente se vazios, inicialmente).
- RF5: Navegação:
  - Suporte à navegação hierárquica entre diretórios (ex: cd, ls).
- RF6: Visualização:
  - Listar arquivos e subpastas dentro de um diretório (ls).

Como estrutura de dados principal, vocês deverão usar a árvore B para representar a estrutura de diretórios e arquivos em cada diretório.

- Cada diretório conterá uma árvore B armazenando os seus filhos: subdiretórios ou arquivos.
- A estrutura dos nós da árvore conterá:
  - Nome do item (arquivo ou pasta)
  - Tipo (arquivo ou pasta)
  - Ponteiro para conteúdo (se arquivo) ou para outra árvore B (se diretório)

Como estrutura base, vocês deverão usar:

```c
//Tipos de dados (estruturas)
typedef enum { FILE_TYPE, DIRECTORY_TYPE } NodeType;

typedef struct File {
    char* name;
    char* content;
    size_t size;
} File;

typedef struct Directory Directory;

typedef struct TreeNode {
    char* name;
    NodeType type;
    union {
        File* file;
        Directory* directory;
    } data;
} TreeNode;

struct Directory {
    BTree* tree; // Árvore B com TreeNode*
};

typedef struct BTree BTree;
```

```c
//Manipulação da Árvore B
BTree* btree_create(int t); // t: grau mínimo da árvore B
void btree_insert(BTree* tree, TreeNode* node);
void btree_delete(BTree* tree, const char* name);
TreeNode* btree_search(BTree* tree, const char* name);
void btree_traverse(BTree* tree); // Exibe itens do diretório

//Funções de Arquivos
TreeNode* create_txt_file(const char* name, const char* content);
void delete_txt_file(BTree* tree, const char* name);

//Funções de Diretórios
TreeNode* create_directory(const char* name);
void delete_directory(BTree* tree, const char* name);

//Sistema de Navegação
Directory* get_root_directory();
void change_directory(Directory** current, const char* path);
void list_directory_contents(Directory* dir);
```

No repositório da disciplina, em especial na pasta Simple File System, você encontra os arquivos bases para a implementação do trabalho. Vocês deverão implementar as funções:

- delete_txt_file
- delete_directory
- change_directory
- list_directory_contents
- btree_search
- btree_insert
- btree_delete
- btree_traverse

Vocês devem manter as estruturas disponibilizadas no desenvolvimento e qualquer alteração nos códigos fornecidos, salvo o que você deve implementar, deverão ser autorizadas pelo professor.

Critérios de Avaliação

- **Corretude funcional:** As operações atendem aos requisitos?
- **Uso apropriado da Árvore B:** Implementação correta e eficiente.
- **Modularização:** Código organizado e funções bem definidas.
- **Interface textual:** Permite navegar e manipular arquivos via terminal.
- **Documentação:** relatório descritivo.
- **Exposição e Apresentação:** vídeo de apresentação atendendo as especificações solicitadas nas instruções do trabalho.