ATIVIDADE PRÁTICA 1

Especificações:

- Esse trabalho pode ser realizado em duplas **ou** individual;
- O valor total do trabalho representará 10% da nota do semestre;
- O trabalho deve ser entregue em formato digital no Moodle (data da entrega: 14/11/24)
- Arquivos .c compactados.
- Não serão aceitos, em hipótese alguma, trabalhos enviados via e-mail.
- Cópias de trabalhos serão penalizadas com a perda total do valor do trabalho.
- O trabalho enviado deve estar claramente identificado, com nome do aluno.

Sistema de Registro e Busca de Pacientes em um Hospital

Problema

Um hospital local deseja informatizar o seu sistema de registro de pacientes. Com o aumento no número de pacientes atendidos diariamente, o hospital enfrenta dificuldades em gerenciar as informações e realizar buscas rápidas e eficientes. Atualmente, os dados são registrados manualmente, o que dificulta o acesso e a organização das informações, gerando atrasos no atendimento.

Para resolver esse problema, o hospital solicitou o desenvolvimento de um sistema capaz de gerenciar os dados de pacientes utilizando uma Árvore Binária de Pesquisa (ABP). O sistema também deve permitir a persistência dos dados em arquivos e possibilitar a organização da árvore por diferentes critérios, como ID ou nome.

Objetivo

Desenvolver um sistema em C que utilize uma Árvore Binária de Pesquisa (ABP) para gerenciar as informações dos pacientes, permitindo a inserção, busca, remoção, listagem e gravação de dados em arquivos. O sistema deve permitir a escolha do critério de ordenação da árvore (utilizando ID ou nome do paciente).

Requisitos do Projeto

O sistema deve ser capaz de:

1. Carregar as informações dos pacientes de um arquivo de texto no início da execução.

2. Cadastrar novos pacientes na árvore, onde cada paciente terá um número de identificação único (ID), nome, idade e condição médica.

3. Escolher o critério de ordenação da árvore: por ID ou nome.

4. Buscar informações de um paciente específico utilizando o número de ID ou nome como chave de busca.

5. Remover um paciente da árvore utilizando o número de identificação (ID) ou nome.

6. Listar todos os pacientes em ordem crescente de acordo com o critério escolhido (ID ou nome). A função deve organizar a árvore de acordo com o critério escolhido.

7. Salvar as alterações em um arquivo de texto ao final da execução.

8. Exibir o total de pacientes cadastrados no sistema.

9. Desalocar memória dinâmica antes de finalizar o programa.

Estrutura de Dados

Cada nó da árvore deve representar um paciente, conforme *struct* a seguir:

```
typedef struct paciente{
    int ID;
    char nome[20]
    int idade;
    char condicao_medica[30]
}TPaciente
```

A árvore deve ser uma Árvore Binária de Pesquisa (ABP), na qual:

- A chave do nó será o ID do paciente ou o nome do paciente, dependendo do critério escolhido pelo usuário.

Persistência de Dados

- Leitura dos dados: O programa deve carregar as informações dos pacientes a partir de um arquivo de texto ao iniciar. O arquivo de texto deve conter uma linha para cada paciente, com os dados separados por um delimitador (por exemplo, espaço ou vírgula).

- Gravação dos dados: O programa deve salvar todos os pacientes cadastrados no arquivo de texto ao final da execução, preservando a integridade dos dados.

Funcionalidades do Sistema

1. Carregar Dados de Pacientes:

Ler as informações dos pacientes de um arquivo de texto ao iniciar, preenchendo a árvore binária de pesquisa de acordo com o critério escolhido (ID ou nome).

2. Cadastro de Pacientes:

Inserir um novo paciente na árvore, validando o critério de chave escolhido. O programa deve verificar se um ID ou nome já existe antes de inserir. Não são permitidos ID ou nomes iguais.

3. Escolha do Critério de Ordenação:

Permitir que o usuário escolha entre ordenar a árvore por ID ou por nome. A escolha deve ser feita no início da execução.

4. Busca de Paciente:

O programa deve permitir a busca por ID ou nome, dependendo do critério de ordenação escolhido.

5. Remoção de Paciente:

O programa deve permitir a remoção de um paciente da árvore com base no ID ou nome.

6. Listagem de Pacientes:

Listar todos os pacientes em ordem crescente de acordo com o critério escolhido, utilizando um percurso in-order.

7. Salvar Alterações em Arquivo:

Antes de encerrar, o programa deve salvar as informações dos pacientes em um arquivo de texto, com os dados atualizados.

8. Contagem Total de Pacientes:

O programa deve exibir o total de pacientes cadastrados na árvore.

9. Desalocar memória dinâmica

Toda memória alocada dinamicamente deve ser liberada.

Critérios de Avaliação

- Implementação correta da Árvore Binária de Pesquisa com a capacidade de escolher o critério de ordenação.

- Inserção, busca e remoção de pacientes com base no ID ou nome, dependendo do critério escolhido.

- Leitura e gravação de dados em arquivo de texto.

- Listagem correta dos pacientes em ordem crescente de acordo com o critério de ordenação.

- Robustez do programa, com verificação de erros, como IDs ou nomes duplicados.

- Eficiência do código, evitando repetição desnecessária de lógica e otimizando o uso de memória.

Diretrizes do Projeto

- Implemente o código em C, utilizando ponteiros para manipular os nós da árvore e gerenciar a memória dinamicamente.

- Comente seu código explicando as funções principais e suas operações.

- Teste o programa com os arquivos .txt disponibilizados no moodle.

- Não esqueça de liberar toda memória alocada dinamicamente antes do programa finalizar.

Entrega

- Código-fonte comentado e devidamente formatado.