Pontifícia Universidade Católica de Minas Gerais
Curso de Ciência da Computação
Disciplina: Algoritmos e Estruturas de Dados II

Trabalho Prático IV

Regras Básicas

- `extends Trabalho Prático 03`
- Fique atento ao Charset dos arquivos de entrada e saída.

  Observação:

  Nas questões de árvore, utilizamos o mostrar pré.

  Não será necessário implementar a opção de remoção nas TADs abaixo.

O Disney+ é um serviço de streaming de vídeo lançado pela The Walt Disney Company em 12 de novembro de 2019. A plataforma oferece um catálogo extenso de filmes e séries das marcas da Disney, incluindo Marvel, Star Wars, Pixar, National Geographic e 20th Century Studios. Desde seu lançamento, tem se consolidado como uma das principais plataformas do mercado de streaming, competindo diretamente com serviços como Netflix, Amazon Prime Video, HBO Max e Apple TV+.

Um dos grandes diferenciais do Disney+ é a oferta de filmes e séries exclusivos, que não estão disponíveis em outras plataformas. Entre os destaques estão produções originais como The Mandalorian (Star Wars), WandaVision (Marvel), Loki, Encanto e muitas outras. Além disso, a plataforma mantém um extenso catálogo de clássicos da Disney, como O Rei Leão, A Bela e a Fera, Aladdin, entre outros, permitindo que diferentes gerações revisitem conteúdos icônicos da empresa.

Desde o seu lançamento, o Disney+ teve um crescimento impressionante, atingindo mais de 100 milhões de assinantes em menos de dois anos. A plataforma está disponível em diversas partes do mundo, com suporte a múltiplos idiomas e adaptações do catálogo conforme a região. Essa expansão rápida reflete a força das marcas que compõem o serviço e o interesse global pelo conteúdo oferecido.

O arquivo DISNEYPLUS.CSV contém um conjunto de dados extraídos do site Kaggle. Este conjunto de dados contém listagens de todos os filmes e programas de TV disponíveis, juntamente com detalhes

como elenco, diretores, classificações, ano de lançamento, duração, entre outros. Tal arquivo deve ser copiado para a pasta /tmp/. Quando reiniciamos o Linux, ele normalmente apaga os arquivos existentes na pasta /tmp/.

Implemente os itens pedidos a seguir.

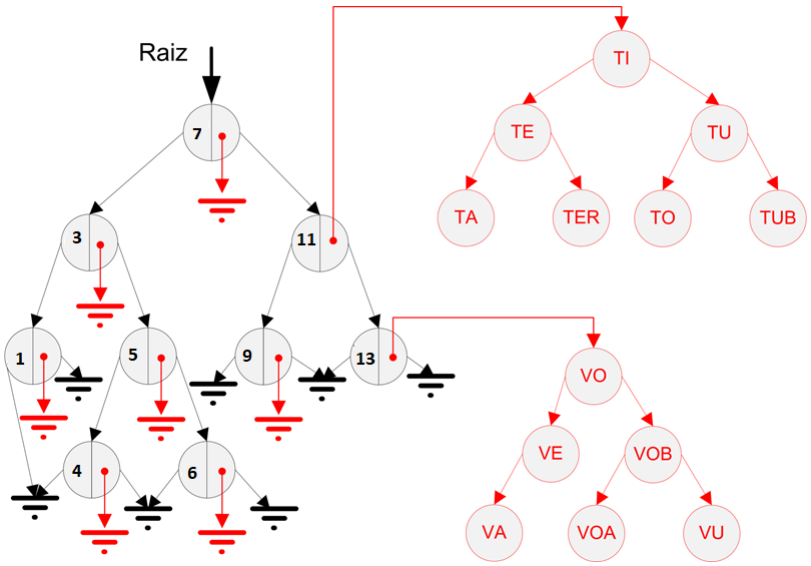
Árvores

Observação: ATENÇÃO para os algoritmos de árvore que já estão implementados no Github!

1. **Árvore Binária em Java**: Crie uma Árvore Binária, fazendo inserções de registros conforme a entrada padrão. A chave de pesquisa é o atributo **name**. Não insira um elemento se sua chave estiver na árvore. Em seguida, pesquise se alguns registros estão cadastrados na Árvore, mostrando seus respectivos caminhos de pesquisa. A entrada padrão é igual a da questão de "Pesquisa Sequencial". A saída padrão é composta por várias linhas, uma para cada pesquisa. Cada linha é composta pelo caminho ou sequência de ponteiros (`raiz, esq ou dir`) utilizados na pesquisa e, no final, pelas palavras SIM ou NAO. Além disso, crie um arquivo de log na pasta corrente com o nome matrícula_arvoreBinaria.txt com uma única linha contendo sua matrícula, tempo de execução do seu algoritmo e número de comparações. Todas as informações do arquivo de log devem ser separadas por uma tabulação '\t'.

2. **Árvore Binária de Árvore Binárias em Java**: Refaça a questão anterior, contudo, considerando a estrutura de árvore de árvore. Nessa estrutura, temos uma árvore binária tradicional na qual cada nó tem um ponteiro para outra árvore binária. Graficamente, a primeira árvore está no plano xy e a árvore de seus nós pode ser imaginada no espaço tridimensional. Temos dois tipos de nós. O primeiro tem um número inteiro como chave, os ponteiros esq e dir (ambos para nós do primeiro tipo) e um ponteiro para nós do segundo tipo. O outro nó tem uma String como chave e os ponteiros esq e dir (ambos para nós do segundo tipo). A chave de pesquisa da primeira árvore é o atributo **captureRate mod 15** e, da outra, é o atributo **name**. Conforme a figura abaixo.

   Destaca-se que nossa pesquisa faz um "mostrar" na primeira árvore e um "mostrar" na segunda. Faremos um "mostrar" na primeira árvore porque ela é organizada pelo **releaseYear mod 15**, permitindo que o valor desejado esteja na segunda árvore de qualquer um de seus nós. Faremos o "mostrar" na segunda porque ela é organizada pelo atributo **name**. Antes de inserir qualquer elemento, crie a primeira árvore, inserindo todos seus nós e respeitando a ordem **7, 3, 11, 1, 5, 9, 13, 0, 2, 4, 6, 8, 10, 12 e 14**. O arquivo de log será matrícula_arvoreArvore.txt.



3. **Árvore AVL em C**: Refaça a primeira questão deste trabalho com Árvore AVL em C. O nome do arquivo de log será matrícula_avl.txt.

4. **Árvore Alvinegra em Java**: Refaça a primeira questão deste trabalho com Árvore Alvinegra. O nome do arquivo de log será matrícula_avinegra.txt.

5. **Tabela *Hash* Direta com Reserva**: Refaça a primeira questão deste trabalho com Tabela *Hash* Direta com Reserva. A função de transformação será **(ASCII name) mod tamTab** onde tamTab (tamanho da tabela) é 21. A área de reserva tem tamanho 9, fazendo com que o tamanho total da tabela seja igual a 30. A saída padrão será a posição de cada elemento procurado na tabela (na *hash* ou na área de reserva). Se o elemento procurado não estiver na tabela, escreva a palavra NÃO. Além disso, o nome do arquivo de log será matrícula_hashReserva.txt.

6. **Tabela *Hash* Direta com Rehash**: Refaça a questão anterior com Tabela *Hash* Direta com *Rehash*. A primeira função de transformação será **(ASCII name) mod tamTab** onde tamTab (tamanho da tabela) é 21 e a outra, **(ASCII name + 1) mod tamTab**. O nome do arquivo de log será matrícula_hashRehash.txt.

7. **Tabela *Hash* Indireta com Lista Simples em C**: Refaça a questão anterior com Tabela *Hash* Indireta com Lista Simples. A função de transformação será **(ASCII name) mod tamTab** onde tamTab (tamanho da tabela) é 21. O nome do arquivo de log será matrícula_hashIndireta.txt.