PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e Informática
Trabalho Prático 4

Curso : *Engenharia de Software*
Disciplina : *Algoritmos e Estruturas de Dados II*
Professora : *Eveline Alonso Veloso*

Regras Básicas:

1. Desenvolva esse trabalho prático a partir do que já foi implementado nos Trabalhos práticos 2 e 3.
2. Estude bastante cada par de entrada/saída.
3. Todos os programas deverão ser desenvolvidos na linguagem de programação Java.
4. Esse trabalho prático poderá ser desenvolvido em grupos de, no máximo, três integrantes.
5. Cópias de trabalho, se existirem, serão encaminhadas ao colegiado de coordenação didática do curso.
6. Fique atento ao *charset* dos arquivos de entrada e saída. Recomenda-se a utilização dos métodos da classe MyIO.java para leitura de dados do teclado.
7. Para cada exercício, vocês devem submeter apenas um arquivo (.java) por grupo. Essa regra será necessária para a submissão de trabalhos no VERDE e no identificador de plágios utilizado na disciplina.
8. A resolução (código) de cada exercício deverá ser submetida ao VERDE.
9. A correção será realizada automaticamente pelo VERDE e validada por meio de apresentações durante as aulas práticas da disciplina.

Base de Dados:

A *National Basketball Association* (em português: Associação Nacional de Basquetebol; abreviação oficial: NBA) é a principal liga de basquetebol profissional da América do Norte. Com 30 franquias como membros (29 nos Estados Unidos e 1 no Canadá), a NBA também é considerada a principal liga de basquete do mundo. É um membro ativo da USA *Basketball* (USAB), que é reconhecida pela FIBA (a Federação Internacional de Basquetebol) como a entidade máxima e organizadora do basquetebol nos Estados Unidos. A NBA é uma das 4 '*major leagues*' de esporte profissional na América do Norte. Os jogadores da NBA são os esportistas mais bem pagos do mundo, por salário médio anual.

A liga foi fundada na cidade de Nova Iorque, em 6 de junho de 1946, como a *Basketball Association of America* (BAA). Adotou o nome de *National Basketball Association* em 1949, quando se

fundiu com a rival *National Basketball League* (NBL). A liga tem diversos escritórios ao redor do mundo, além de vários dos próprios clubes fora da sede principal na *Olympic Tower*, localizada na Quinta Avenida, 645. Os estúdios da NBA *Entertainment* e da NBA TV estão localizados em Secaucus, New Jersey.

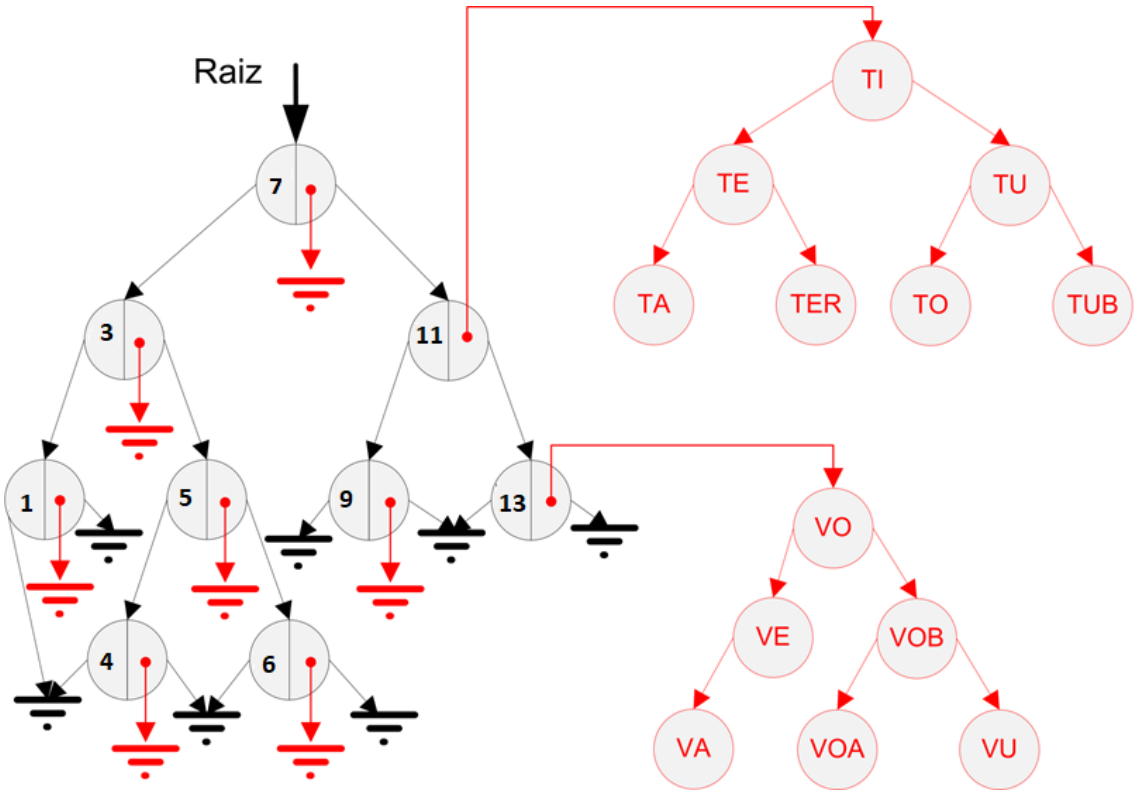
O arquivo players.csv contém um conjunto de dados de jogadores da liga de basquete norte-americana – NBA – extraídos do *site* https://www.kaggle.com/drgilermo/nba-players-stats. Essa base contém registros de jogadores desde 1950, um total de 67 temporadas da NBA. Esse arquivo sofreu algumas adaptações para ser utilizado neste e nos próximos trabalhos práticos da disciplina. Tal arquivo deve ser copiado para a pasta /tmp/.

Exercícios:

Árvores:

1. **Árvore Binária de Busca**
   Crie uma árvore binária de busca. Em seguida, faça a inserção dos registros correspondentes a alguns jogadores conforme a entrada padrão. A chave de pesquisa será o atributo ***nome*** do jogador. Não insira um jogador se sua chave já estiver na árvore. Por fim, pesquise se alguns registros estão cadastrados na árvore, mostrando seus respectivos caminhos de pesquisa.
   A entrada padrão é igual a da questão **Pesquisa sequencial em Java**.
   A saída padrão será composta por várias linhas, uma para cada pesquisa realizada na árvore. Cada linha será composta pelo caminho de pesquisa, ou seja, os ***nomes*** dos jogadores que foram inspecionados durante o processamento da pesquisa; e, no final, pelas palavras SIM ou NAO, indicando se cada um dos nomes pesquisados existe ou não na árvore.
   Além disso, crie um arquivo de *log* na pasta corrente com o nome matrícula_arvoreBinaria.txt com uma única linha contendo: seu número de matrícula, tempo de execução de seu algoritmo (em milissegundos) e número de comparações realizadas. Todas as informações desse arquivo de *log* devem ser separadas por uma tabulação '\t'.

2. **Árvore Binária de Busca de Árvore Binária de Busca**
   Refaça a questão anterior, contudo, considerando a estrutura "árvore binária de busca de árvore binária de busca". Nessa estrutura, temos uma árvore binária de busca na qual cada nó tem uma referência para outra árvore binária de busca. Graficamente, a primeira árvore binária de busca está no plano xy e as árvores binárias de busca referenciadas por seus nós podem ser imaginadas no espaço tridimensional, como ilustrado na figura abaixo:



Temos dois tipos de nós: os nós da primeira árvore binária de busca têm um número inteiro como chave, as referências esquerda e direita (ambas para nós desse primeiro tipo) e uma referência para nós do segundo tipo. O outro tipo de nó tem uma *string* como chave e as referências esquerda e direita (ambas para nós desse segundo tipo). A chave de pesquisa da primeira árvore binária de busca é o atributo ***altura*** mod 17 e a da outra árvore binária de busca é o atributo ***nome*** do jogador.

Destaca-se que nossa pesquisa faz uma busca na primeira árvore binária de busca e, depois, procura na segunda. Faremos uma busca na primeira árvore binária de busca porque ela é organizada pelo atributo ***altura*** mod 17, permitindo assim que o valor desejado esteja na árvore binária de busca referenciada por qualquer um de seus nós. Faremos uma pesquisa na segunda árvore binária de busca porque ela é organizada pelo atributo ***nome*** do jogador.

O caminho de pesquisa apresentado na saída padrão deve exibir as chaves dos nós da primeira árvore binária de busca que foram inspecionados, seguidas dos ***nomes*** dos jogadores que foram inspecionados, na segunda árvore binária de busca, durante o processamento da pesquisa.

O nome do arquivo de *log* será matrícula_arvoreArvore.txt.

3. **Árvore AVL**

   Refaça a questão **Árvore Binária de Busca** com Árvore AVL.

   O nome do arquivo de *log* será matrícula_AVL.txt.

4. ***Tree sort***

Tree sort é um algoritmo de ordenação. Nesse algoritmo, constrói-se uma árvore binária de busca com os elementos a serem classificados e, em seguida, percorre-se a árvore (utilizando caminhamento em ordem) para que seus elementos sejam impressos em ordem de classificação.
Efetue a ordenação de todos os jogadores presentes no arquivo /tmp/players.csv pelo atributo ***nome***, usando o algoritmo *Tree sort*. Observe o formato da saída padrão.
Além disso, crie um arquivo de *log* na pasta corrente com o nome matrícula_treeSort.txt com uma única linha contendo: seu número de matrícula, tempo de execução de seu algoritmo (em milissegundos) e número de comparações realizadas para inserção de todos os jogadores na árvore. Todas as informações desse arquivo de *log* devem ser separadas por uma tabulação '\t'.

Tabelas *hash*:

5. **Tabela *hash* com endereçamento aberto e *rehashing***
   Crie uma tabela *hash* com endereçamento aberto e *rehashing*.
   Em seguida, faça a inserção dos registros correspondentes a alguns jogadores conforme a entrada padrão.
   A função de transformação que deve ser aplicada é: **(altura mod tamanho da tabela *hash* + k * (altura mod 23)) mod tamanho da tabela *hash***, onde o tamanho da tabela *hash* é 51 e *k* indica o número de tentativas de se inserir o registro na tabela, começando com o valor 0.
   Por fim, pesquise se alguns jogadores estão cadastrados na tabela *hash*, mostrando suas respectivas posições nessa tabela.
   A entrada padrão é igual a da questão **Pesquisa sequencial**.
   A saída padrão será composta por várias linhas, uma para cada pesquisa realizada na tabela *hash*. Cada linha deve apresentar a posição de cada jogador procurado na tabela *hash*. Se o elemento desejado não estiver na tabela, escreva a palavra NAO.
   Além disso, crie um arquivo de *log* na pasta corrente com o nome matrícula_hashRehashing.txt com uma única linha contendo: seu número de matrícula, tempo de execução de seu algoritmo (em milissegundos) e número de comparações realizadas. Todas as informações desse arquivo de *log* devem ser separadas por uma tabulação '\t'.

6. **Tabela *hash* com endereçamento em separado**
   Crie uma tabela *hash* com endereçamento em separado.
   Em seguida, faça a inserção dos registros correspondentes a alguns jogadores conforme a entrada padrão.
   A função de transformação que deve ser aplicada é: **(altura mod tamanho da tabela *hash*)**, onde o tamanho da tabela *hash* é 37.
   Por fim, pesquise se alguns jogadores estão cadastrados na tabela *hash*, mostrando suas respectivas posições nessa tabela.
   A entrada padrão é igual a da questão **Pesquisa sequencial**.
   A saída padrão será composta por várias linhas, uma para cada pesquisa realizada na tabela *hash*. Cada linha deve apresentar a posição de cada jogador

procurado na tabela *hash*. Se o elemento desejado não estiver na tabela, escreva a palavra NAO.
Além disso, crie um arquivo de *log* na pasta corrente com o nome matrícula_hashSeparado.txt com uma única linha contendo: seu número de matrícula, tempo de execução de seu algoritmo (em milissegundos) e número de comparações realizadas. Todas as informações desse arquivo de *log* devem ser separadas por uma tabulação '\t'.