Projeto – Estrutura de Dados 2

O objetivo deste projeto é avaliar tecnicamente o desenvolvimento dos conceitos adquiridos durante as aulas da disciplina Estrutura de Dados 2, além de prepara-los para uma prévia dos problemas que irão encontrar no dia a dia e em processos seletivos de empresas.

O projeto deve ser codificado utilizando a linguagem JAVA de programação. Os arquivos fonte que compõe a solução deve ser entregue para análise e avaliação. Os arquivos que compõe o projeto devem ser enviados via exercício do BlackBoard. Esses arquivos devem ser entregues na extensão .zip, e um arquivo README dentro do zip deve conter instruções de compilação e de execução do mesmo.

Cada aluno entregará o seu projeto, ou seja, o projeto será desenvolvido e entregue via BlackBoard individualmente.

Bom desenvolvimento!

O Projeto

Árvores de decisão são importantes estruturas usadas em vários ramos da Ciência da Computação, tais como, classificação, inteligência artificial (IA), aprendizado de máquina (machine learning), Jogos, etc.

Na árvore de decisão, um simples nó folha representa o conjunto de "solução" ou respostas. Já a raiz e os nós filhos representam o conjunto de "questões".

O usuário do software, chega a solução final através das respostas fornecidas aos questionamentos. Estas respostas são do tipo Sim (Yes) ou Não (No).

A natureza da resposta fornecida a uma determinada questão irá indicar qual caminho deve-se seguir para a próxima questão ou para o nó folha com a solução final.

Para o caso de uma árvore binária, as respostas são tipicamente Sim (Yes) ou Não (No), as quais correspondem a uma das duas subarvores referentes aos nós de questões.

Será tomado como convenção o seguinte critério de percurso para respostas afirmativas, do tipo Sim (Yes):

- Quando o usuário entrar com resposta Sim (Yes), a próxima pergunta corresponderá ao próximo nó da subarvore ESQUERDA.

Outra convenção que será levada em consideração é o critério de percurso para respostas negativa, do tipo Não (No):

- Quando o usuário entrar com resposta Não (No), a próxima pergunta corresponderá ao próximo nó da subarvore DIREITA.

Ao final do percurso de todas as perguntas, o software irá indicar a resposta equivalente ao caminho percorrido.

A única iteração do usuário com o software será através das respostas Sim (Yes) ou Não (No). Sendo assim, nenhum outro tipo de informação inserida pelo usuário deverá ser levado em consideração. Caso haja inserção de dados divergentes as respostas consideradas, de a opção de repetição para a inserção da resposta correta aceitável, ou seja, apenas Sim (Yes) ou Não (No).

Nunca se esqueça, deixar o código seguro quanto a parâmetros de entrada de usuário, é meio caminho andado para evitar Bugs. Além do mais, o software deve ser robusto o suficiente para contornar erros de entrada de informação.

Comece a implementação levando em consideração o seguinte cenário com as perguntas e respostas abaixo:

- O animal come carne?
  - Sim (Yes)
    - O animal tem listras?
      - Sim (Yes)
        - É um TIGRE
      - Não (No)
        - É um LEOPARDO
  - Não (No)
    - O animal tem listras?
      - Sim (Yes)
        - É uma ZEBRA
      - Não (No)
        - É um CAVALO

Um exemplo de como as informações poderão ser exibidas:

```
O animal come carne?
sim
O animal tem listras? (yes)
nao
RESPOSTA: É um LEOPARDO
BUILD SUCCESSFUL (total time: 5 seconds)

O animal come carne?
nao
O animal tem listras? (no)
sim
RESPOSTA: É uma ZEBRA
BUILD SUCCESSFUL (total time: 4 seconds)
```

Ultimas informações sobre a implementação:

1. Não há limites para criação de classes, métodos, variáveis e objetos. Encontre a quantidade de acordo com a sua necessidade.
2. Reutilize os códigos implementados em laboratório.
3. Utilizem Orientação a Objetos (OO) e linguagem JAVA.

Mãos à obra!!! ☺

Dica 1: Reutilizar o código da classe "*NodeTree*".

Dica 2: Crie uma classe para manipular as entradas das respostas do usuário. (Exemplo: método "*capturaResposta()*" e classe "*GameHandle*")

Dica 3: Crie uma classe para manipular o percurso de acordo com a entrada das respostas. (Exemplo: método "*gamePlay()*" e classe "*GameHandle*")

Dica 4: Crie uma classe para implementar a sua árvore de decisão. (Exemplo: classe "*DecisionTree*")

Dica 5: Na classe de implementação da árvore de decisão, crie o método para inserção de nó. (Exemplo: método "*inserir()*")

Dica 6: Crie uma classe principal com o método "*main()*", o qual executará métodos ("*game1()*", "*game2()*", etc) de criação e execução da árvore (criando nós de pergunta e nós de resposta).

Dica 7: Pense no funcionamento como se fosse um jogo de perguntas e respostas. (Exemplo: *Akinator*)

Observação 1:

Parabéns!

"*Inteligencia artificial (IA) é a inteligência similar à humana exibida por mecanismos ou* ***software***" (John McCarthy)

Sim! Pelo que podem ver, vocês estão iniciando o desenvolvimento de uma IA, mais especificamente quando desenvolverem a tomada de decisão das respostas para as perguntas.

Observação 2:

"*Diante de uma dificuldade, substitua o não consigo pelo vou tentar outra vez*" (de um pensador anônimo).