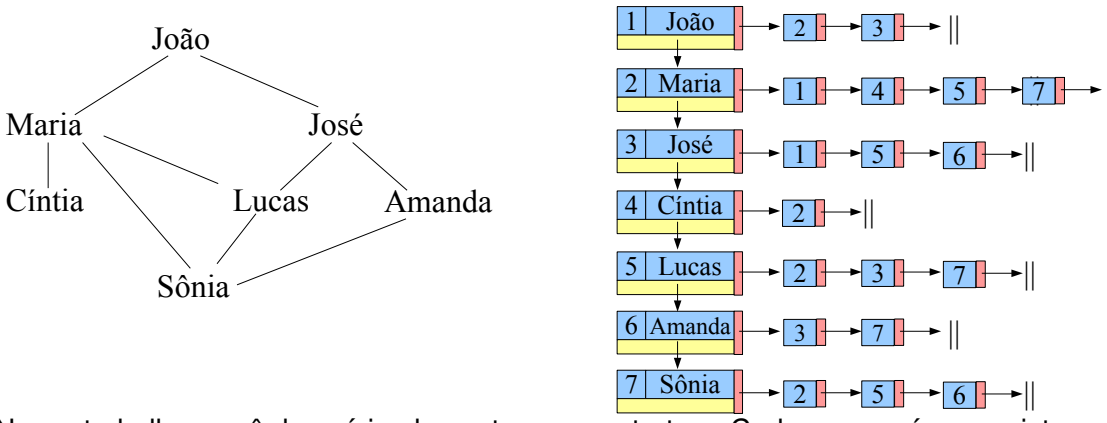
Trabalho sobre TAD's

Você está em um projeto para implementar uma nova rede social. A sua primeira tarefa é implementar um programa para controlar as relações de amizade dentro da rede. De forma geral, para cada pessoa cadastrada você tem que armazenar a sua lista de amigos. Uma forma de visualizar os relacionamentos é através de uma estrutura chamada grafo. Um grafo é composto por vários nodos que são conectados por arestas indicando algum tipo de relação entre esses nodos. Um grafo pode ser implementado através de listas encadeadas: basicamente, implementa-se uma lista onde cada registro contem os dados de um nodo. Além disso, cada registro desta lista contém uma lista encadeada indicando quais os nodos que estão ligados a ele. Isso pode ser observado no seguinte exemplo: João é amigo da Maria e do José; Maria é amiga da Cíntia, Sônia e Lucas; e assim por diante. A figura abaixo mostra o grafo e a estrutura de dados correspondente. Note que cada pessoa tem um identificador único dentro da rede social (atributo ID), e esse atributo é utilizado para construir as listas de amigos:



Nesse trabalho, você deverá implementar essa estrutura. Cada pessoa é um registro contendo os campos nome e cidade além do identificador único. Considere que os nomes e as cidades são compostos por uma única palavra cada (Maria, BH; Hugo, Contagem; etc) e que não existem homônimos na rede social. A rede deve ter as seguintes operações:

- Cadastra(pessoa): cadastra uma nova pessoa na rede social. O identificador único dessa pessoa deve ser gerado automaticamente.
- Remove(Nome): remove a pessoa identificada na rede social. Note que todas as relações de amizade dessa pessoa devem ser removidas.
- Amigos(Nome1, Nome2): Cria uma relação de amizade entre duas pessoas já cadastradas.
- Briga(Nome1, Nome2): Remove uma relação de amizade existente
- Frequencia(Cidade): Imprime o número de pessoas cadastradas de uma determinada cidade
- ImprimeDados(Nome): imprime os dados de uma pessoa bem como o nome e a cidade de todos os seus amigos.
- ImprimeTudo(): imprime os dados de todas as pessoas cadastradas bem como as listas de amigos (com nome e cidade de cada um).

A implementação da estrutura deverá ser feita utilizando alocação dinâmica de memória (referências). Use células sentinela para facilitar a manipulação. Você deve fazer testes de consistência se essas operações podem ser aplicadas e deve imprimir mensagens de sucesso ou falha (por exemplo, Remove(Carlos) deve dar uma mensagem de erro pois não há ninguém cadastrado com esse nome). Além disso, procure escrever funções e procedimentos auxiliares que facilitem a implementação das operações acima, evitando a repetição desnecessária de código. Por exemplo, funções de manipulação de listas, pesquisa de id por nome, etc...

O seu programa deverá ler os dados de entrada a partir de um arquivo (Faz parte do trabalho descobrir como manipular arquivos e Strings em Java). Esse arquivo é basicamente uma lista de comandos (um por linha) em formato texto. O formato a ser usado é exemplificado abaixo:

`CADASTRA nome cidade`	(ex. CADASTRA Cris Ipatinga)
`REMOVE nome`	(ex. REMOVE Cris)
`AMIGOS nome1 nome2`	(ex. AMIGOS Cris Maria)
`BRIGA nome1 nome2`	(ex. BRIGA Cris Maria)
`FREQUENCIA cidade`	(ex. FREQUENCIA Contagem)
`IMPRIME_DADOS nome`	(ex. IMPRIME_DADOS Maria)
`IMPRIME_TUDO`	(ex. IMPRIME_TUDO)

Implemente uma função Distância(Nome1, Nome2) que imprime a "distância" entre dois amigos, ou seja, o menor número de arestas necessárias para ir de uma pessoa a outra. No exemplo acima, a distância entre a Maria e a Cíntia é 1, entre o João e a Sônia é 2 e entre a Cíntia e a Amanda é 3. Para simplificar, considere que a distância máxima que deve ser considerada é 5 (acima disso, deve ser impressa uma mensagem que eles não são amigos próximos). Uma sugestão para implementar essa função é utilizar uma Fila auxiliar. É necessário implementar a função, testá-la e explicar detalhadamente como ela foi implementada (bem como sua ordem de complexidade).

OBSERVAÇÕES:

- Data de entrega: 25/08/2021
- Clareza, identação, comentários e boas práticas OO também serão avaliados.
- Você NÃO deve utilizar estruturas de dados já implementadas na linguagem tais como ArrayList, LinkedList ou similares. Ou seja, você deverá implementar todas as estruturas utilizadas.
- Penalização por atraso: (2*d – 1) pontos, onde d é o número de dias de atraso.

BOM TRABALHO !

BRUNO ROLDÃO: 5,5
trabalho: 90%
- desfaz amizade de quem não é amigo
apresentação: 0
-

EDER: 4,5
trabalho: 70%
- não compilava.
apresentação: 0
-

EGMON: 5
trabalho:
- Não implementou a manipulação de arquivos
apresentação:
- Não fez

GABRIELA: 6,8
trabalho:
- Não implementou a manipulação de arquivos
apresentação: 90%
- problema ao realizar amizade lida do arquivos

KAYKE: 7
trabalho:
apresentação: 100%
-

MARCO TÚLIO: 9
trabalho: 100%
apresentação: 100%
-

MATHEUS: 6
trabalho: 100%
apresentação: 0
-

TIAGO: 5,5
trabalho:
- Após ser apagado, usuário continua na lista de amizade
apresentação: 0
-

WASHINGTON: 5
trabalho:
- Não desfez amizade após excluir
- Não desfez amizade após briga
apresentação: 0
-