Instituto Superior de Engenharia de Lisboa
Licenciatura em Engenharia Informática e de Computadores
Algoritmos e Estruturas de Dados
Semestre de Inverno 2017/18
Terceira série de problemas, Parte 1

Observações:

- Data de entrega: **11 de Dezembro de 2017**.
- **Para os métodos da primeira parte da série terão de ser desenvolvidos e entregues testes unitários.**

1 Exercícios

1. Realize a classe `TreeUtils`, contendo os seguintes métodos estáticos:

   1.1. O método

   ```
   public static <E> boolean contains(Node<E> root, E min, E max, Comparator<E> cmp)
   ```

   que retorna `true` se e só se a árvore binária de pesquisa com raíz `root` contém algum elemento no intervalo $[min, max]$, segundo o comprador `cmp`.

   1.2. O método

   ```
   public static <E extends Comparable<E>> Node<E> lowestCommonAncestor(Node<E> root, E n1, E n2)
   ```

   que dados dois elementos `n1` e `n2` presentes na árvore binária de pesquisa referenciada por `root` retorna o *menor antecessor comum* a dois nós que contenham `n1` e `n2`. O *menor antecessor comum* entre dois nós `v` e `w` **é definido** como o nó mais distante da raíz que é simultaneamente antecessor dos nós `v` e `w`. Note que existem casos em que o *menor antecessor comum* entre dois nós `v` e `w` pode coincidir com o nó `v` ou com o nó `w`.

   1.3. O método

   ```
   public static <E> boolean isBalanced(Node<E> root)
   ```

   que verifica se a árvore binária, referenciada por `root`, é balanceada.

   Para as implementações destes métodos, considere que o tipo `Node<E>` tem 3 campos: um `value` e duas referências, `left` e `right`, para os descendentes respectivos.

2. Realize a classe `AutoCompleteUtils`, com métodos utilitários para a criação de uma aplicação, semelhante à existente nos telemóveis, que sugira palavras, completando automaticamente palavras parcialmente escritas. Assuma que nesta aplicaçao apenas estao disponíveis caracteres de 'a' a 'z'. Com vista à realização desta aplicação, implemente uma *Trie*:

   2.1. O método

   ```
   public static TNode loadWordsFromFile(TNode root, String fileName)
   ```

   que armazena, numa árvore n-ária referenciada por `root` todas as palavras que ocorrem no ficheiro de texto.

   2.2. O método

   ```
   public static TNode longestWithPrefix(TNode root, String prefix)
   ```

   que retorna a referência para o nó da árvore n-ária referenciada por `root` que contenha o prefixo da palavra `prefix` ou `null` caso não exista.

   2.3. O método

   ```
   public static int countPossibleWords(TNode root, String prefix)
   ```

   que retorna o número de palavras que tenham como prefixo a palavra `prefix` e que existam na árvore n-ária referenciada por `root`.

   Para as implementações destes métodos considere que o tipo `TNode` tem 2 campos: um boleano `isWord` e um *array* `children` do tipo `TNode` (com dimensão 'z' - 'a' +1).

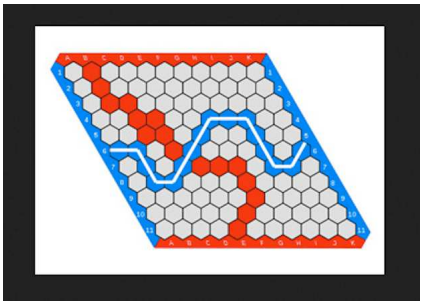
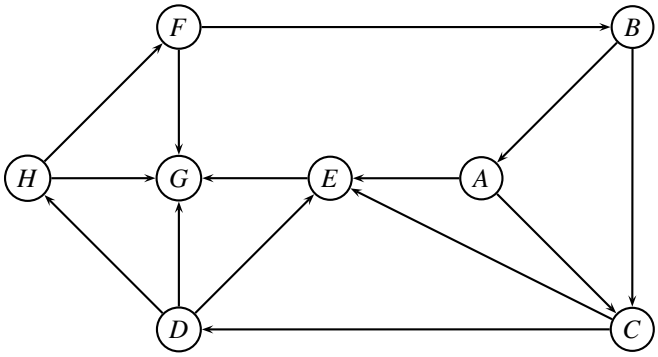


Figure 1: Exemplo do jogo Hex.

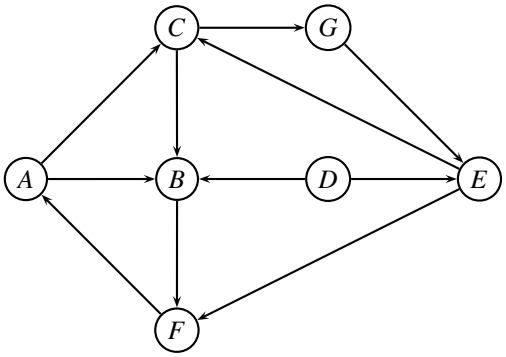
3. Hex é um jogo de tabuleiro jogado numa grelha hexagonal, tradicionalmente como um losango 11x11. Cada jogador tem uma cor, e revezam-se colocando uma pedra da sua cor numa única célula dentro do tabuleiro. O objetivo é formar um caminho conectado de pedras da mesma cor, que liga os lados opostos do tabuleiro marcados com suas cores, antes que o oponente conecte os seus lados de forma semelhante. O primeiro jogador a completar a caminho ganha o jogo. Como exemplo, considere a Figura 1, no qual foi o jogador com cor azul que ganhou.

   Descreva como é que a estrutura conjuntos disjuntos poderá ser utilizada na implementação deste jogo de tabuleiro para detectar que o jogo terminou.

4. Considere o seguinte grafo orientado com 8 vértices e 14 arcos. Qual a sequência de vértices visitados numa travessia em profundidade primeiro (DFS) sobre este grafo, com origem no vértice A ? Justifique. Considere que os vértices são visitados por ordem alfabética e que os vértices adjacentes de um vértice também são visitados por ordem alfabética.



5. Qual a sequência de vértices visitados numa procura em largura primeiro (BFS) sobre o grafo abaixo, com origem no vértice A. Considere que os vértices são visitados por ordem alfabética e que os vértices adjacentes de um vértice também são visitados por ordem alfabética. Justifique.



ISEL, 20 de Novembro de 2017.