1 Objectivos

Utilização da linguagem Java e dos tipos abstratos de dados leccionados e implementados pelos alunos, para resolver um problema do tipo "Sopa de Letras". O aluno deve providenciar juntamente com os ficheiros pedidos a implementação de todos os tipos de dados que usar na resolução do trabalho.

2 O Trabalho

2.1 Descrição

Pretende-se que o seu programa seja capaz de produzir soluções para um jogo de Boggle. Este jogo aparece em alguns jornais americanos e rivaliza com as habituais palavras cruzadas ou as mais antigas sopa de letras. No Boggle há uma grelha de 4x4 letras, sendo o objectivo do jogo encontrar o maior número de palavras possível e quanto mais letras tiverem as palavras encontradas maior a pontuação. As palavras são formadas usando uma qualquer letra da grelha e escolhendo para formar as palavras qualquer letra adjacente. A única restrição é que não é possível repetir a mesma letra(posição), numa mesma palavra. A figura abaixo apresenta um exemplo real dum boogle do Miami Herald, as palavras mouse, monkey e mule por exemplo são possíveis de encontrar usando as regras do jogo.

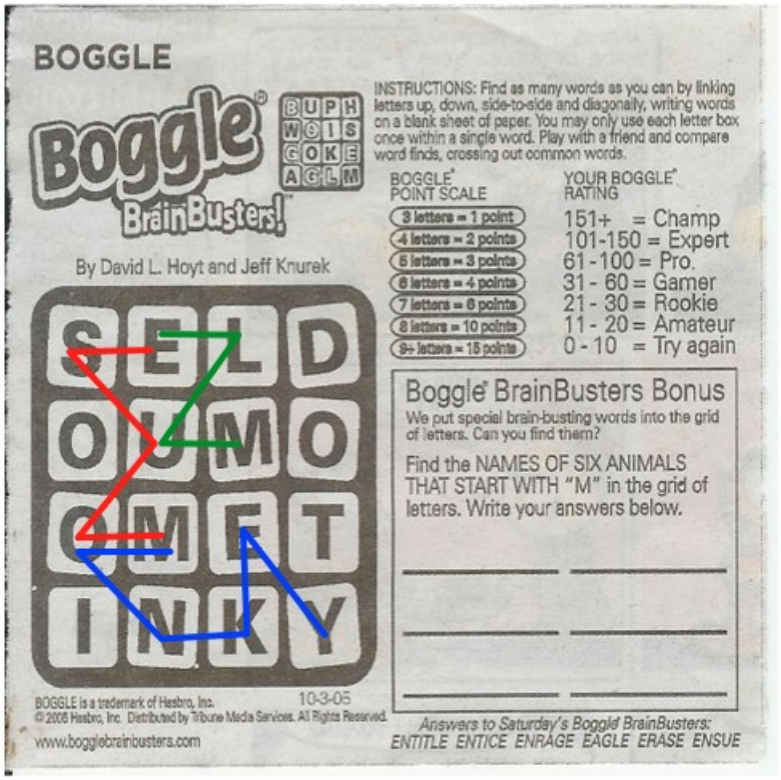

Figure 1: Exemplo dum boggle

2.2 Realização

São fornecidos para a realização do trabalho dois ficheiros:

- um, com um boggle, isto é uma matriz de caracteres.
- outro, uma lista de todas as palavras no Inglês, num ficheiro de texto, uma palavra por linha. Os boggles e o dicionário são Inglês, dado que os caracteres característicos do português, tipo ç, ã, õ, etc, costumam trazer problemas adicionais que em nada se traduzem com conhecimentos de Estruturas de Dados e Algoritmos. Não se preocupe com as palavras do dicionário, se está lá a palavra, pode usá-la.

o output do trabalho, deverá ser uma lista com as palavras encontradas. Deverá providenciar o output da lista, devendo ser listadas as soluções na forma:
mouse (M:(2,1))–>(O:(2,0))–>(U:(1,1))–>(S:(0,0))–>(E:(0,1)). Para a realização do trabalho deverá implementar a classe *Position*, que agrupa no mesmo objecto o conceito de linha e coluna. Para esta classe deverá definir um construtor, o *toString* e o *equals*. Não se esqueça deste último dado que o *contains* para as listas usa o *equals*. A estratégia a seguir é simples, vão-se formando palavras usando uma posição do boggle e as suas posições adjacentes. Se e palavra encontrada é sufixo de uma palavra do dicionário, pode continuar guardando na lista de posições a nova posição. As posições adjacentes são N,S W,E, NE, SE, SW e NW. Tenha somente em atenção que na construção duma palavra não pode repetir posições. Se a palavra formada não é sufixo mas uma palavra do dicionário então guarda a palavra e a lista das posições na lista que contem os resultados.

2.3 Entrega

O trabalho deverá ser entregue até ao dia 20 de Janeiro de 2016, sendo realizada a submissão pelo moodle, nos moldes habituais. Deve submeter os seguintes ficheiros:

- Uma implementação de Listas(que vai usar no trabalho)
- Uma implementação de Tabelas de Hash com acesso fechado à sua escolha. Estas implementações deverão seguir as sugestões dadas nas aulas práticas(uma classe abstracta *HashTable* e uma subclasse com o hasinhg fechado de acesso que escolher). A tabela de hash servirá para guardar as palavras do dicionário e os seus sufixos
- A classe *Boggle*. Esta classe deverá ter os métodos que permitem resolver o problema proposto, tipo *ler_boogle(...)* e um método *solve()* que retorna a lista com as soluções encontradas
- Todas as classe que considerar necessárias para a construção da solução, incluindo a classe *Position*
- Um relatório adequado à apresentação do trabalho

estes ficheiros deverão ser "zipados" e submetidos num único ficheiro com o seu número de aluno. O trabalho é individual.