**Relatório TP3 AED 2017/2018**

Nome: João Filipe Sequeira Montenegro TP: PL3

Nº Estudante: 2016228672 E-mail/login no Mooshak: FilipeMontenegro

# Estruturas de Dados utilizadas

Para guardar o dicionário base de palavras e o histórico de correções foram utilizadas duas tabelas de dispersão usando *separate chaining* para tratar colisões.

Para guardar palavras geradas através de erros ortográficos e guardar palavras obtidas do dicionário são usadas listas.

# Justificação para as estruturas

Escolhi usar tabelas de dispersão principalmente devido à fácil implementação e devido à complexidade temporal média ser O(1), especialmente se for utilizada uma boa função de dispersão, garantindo que as entradas são espalhadas corretamente pela tabela.

Em alternativa, poderiam ser utilizadas árvores-BK, em que um nó de raiz é uma palavra, e os nós adjacentes são palavras semelhantes. A distância a que cada palavra se encontra da raiz depende de quantos ajustes tem de ser feita à palavra para ser igual à raiz (distância de Levenshtein).

# Fontes

Modelo de erros - <https://norvig.com/spell-correct.html> e <https://medium.com/@willsentance/how-to-write-your-own-spellchecker-and-autocorrect-algorithm-in-under-80-lines-of-code-6d65d21bb7b6>

Função de dispersão - <http://www.cse.yorku.ca/~oz/hash.html>

Lista de palavras portuguesas - <http://natura.di.uminho.pt/download/sources/Dictionaries/wordlists/>

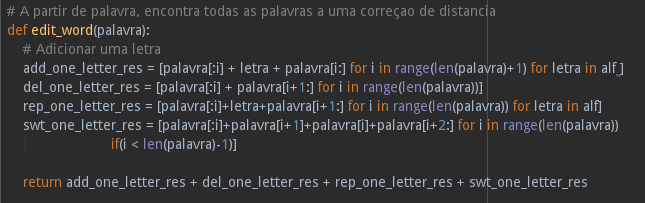
Frequência de palavras - <https://en.wiktionary.org/wiki/Wiktionary:Frequency_lists/Portuguese_wordlist>

# Modelo de erros tipográficos

O modelo é baseado na maneira de corrigir palavras descrita no site incluído nas fontes do enunciado “How to write a spelling corrector” (<https://norvig.com/spell-correct.html>)

Ao encontrar uma palavra errada, é primeiro gerado todas as palavras de distância de erro 1, isto é, todas as palavras que podem ser criadas ao adicionar uma letra em cada posição, ao substituir uma letra por outra em cada posição, ao remover uma letra em cada posição e ao trocar todas as letras adjacentes, duas de cada vez.

Como erros de distância 2 não são suficientes para algumas palavras, todas as palavras geradas voltam a ser editadas da mesma maneira e posteriormente são guardadas todas na mesma lista.



Para criar as palavras são usadas listas de compreensão e *list splicing* em Python.

Se a palavra já tiver sido corrigida antes, é obtida a correção do histórico, perguntando ao utilizador se a pretende usar.

# Exemplo de utilização

O programa recebe as palavras de input através do ficheiro de texto *input.txt*.

Após uns breves segundos para carregar o dicionário de palavras na tabela de dispersão, o programa verifica se determinada palavra existe no dicionário.

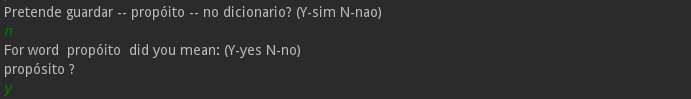
Se não existir, é perguntado ao utilizador se a pretende guardar assim no dicionário.



Se não quiser guardar a palavra, é primeiro verificado se a palavra existe no histórico, e, se existir, é perguntado ao utilizador se pretende usar a correção guardada.



Se não se encontrar no histórico, o programa procede à correção da palavra, gerando todas as palavras possíveis de distância de erro 1 e 2, verificando quais dessas são palavras reais, e ordenando por frequência, de modo a tentar sugerir a palavra correta. As sugestões são sugeridas uma a uma.



Se o utilizador não aceitar no máximo 5 sugestões, é colocado CORREÇÃO-NÃO-ENCONTRADA na frase final.

