Corretor Ortográfico

Nesta tarefa você implementará um corretor ortográfico simples. Seu programa receberá como entrada uma palavra e informará se a palavra está correta, se ela está escrita incorretamente ou se ela não existe. Após o retorno, o usuário terá a opção de aceitar a correção ou indicar que a palavra original está correta e o dicionário deve ser atualizado com a palavra nova.

Naturalmente, isso deve ser feito de maneira eficiente, uma vez que potencialmente milhões de palavras devem ser verificadas em um longo texto. Esta é uma situação onde Tabelas Hash são úteis e você deverá implementar uma para solucionar o problema.

Entrada

A entrada é composta por comandos, um por linha. Quatro comandos são aceitos:

palavra	onde estar-se-á fazendo uma consulta à palavra indicada retorna "ok *palavra*" se a palavra constar no dicionário retorna a palavra correta se a palavra for reconhecida como incorreta retorna "not found" se a palavra não for reconhecida
+	onde a última palavra informada é adicionada ao dicionário, se não for conhecida retorna "ok *palavra*" se a palavra for adicionada com sucesso retorna "fail *palavra*" caso contrário (inclusive se a palavra já existe)
-	onde a última palavra informada é removida do dicionário retorna "ok *palavra*" se a palavra for removida com sucesso retorna "fail *palavra*" caso contrário (inclusive se a palavra não existe)
*	onde indica-se que o programa pode ser encerrado

Algoritmo de reconhecimento

O algoritmo para reconhecer uma palavra testa por variações na palavra fornecida e verifica se pelo menos uma destas variações corresponde a uma palavra correta. Todas palavras candidatas resultantes devem ser listadas, em ordem alfabética. As seguintes variações devem ser testadas:

1. Uma letra a mais: a palavra fornecida possui uma letra qualquer a mais comparada à original. Exemplo: "carrro"
2. Uma letra a menos: a palavra fornecida possui uma letra qualquer a menos em qualquer posição. Exemplo: "caroça"

3. Letras trocadas: a palavra fornecida possui duas letras vizinhas em posições invertidas. Exemplo: "computdaor"
4. Uma letra errada: a palavra correta pode ser construída trocando uma letra da palavra original por outra. Exemplo: "computafor"

Requisitos da Tabela Hash

A Tabela Hash deve ser capaz de suportar vários milhares de palavras e não utilizar muita memória. O vetor inicial deve conter 50 posições. Você deve especificar totalmente o restante da tabela hash, podendo utilizar qualquer combinação de função hash, fator de carga, tratamento de colisões etc. Apenas as bibliotecas padrões podem ser utilizadas (exceto a STL, que não deve ser utilizada).

Seu programa deve ser implementado em C ou C++, em arquivo único. Deve-se utilizar a entrada e saída padrões. Ele deve ser compilável com GNU Compiler Collection (GCC) . A saída do programa deve respeitar **estritamente** o especificado.

Suposições

Todas palavras estarão codificadas em ASCII e terão menos de 100 caracteres. Palavras são compostas por letras e hífens e sempre iniciam por uma letra. As letras podem ser maiúsculas ou minúsculas. Todas saídas devem ser em letras minúsculas.

Avaliação

Sua tabela hash será testada em um amplo conjunto de testes e avaliada segundo o tempo total de execução e a quantidade de memória utilizada (uso máximo ao longo da execução). O tempo e memória utilizados devem ser menores que um valor máximo aceitável, ambos decorrentes de uma implementação minimamente correta. Adicionalmente, todas implementações da turma serão rankeadas. Implementações acima de dois desvios padrões da média de tempo serão penalizadas, enquanto as abaixo de dois desvios padrões receberão pontos extras.

Este trabalho tem peso de 20% da nota do bimestre.

Envio

O nome do seu arquivo fonte deve seguir o padrão **NomeSobrenome.[c|cpp]**. Em adição ao código fonte, também deve ser enviado um arquivo NomeSobrenome.txt contendo uma ***lista de bugs conhecidos*** do programa. Bugs descobertos que não estão nesta lista terão desconto maior. O envio de ambos será feito pelo AVA da disciplina.

Código de Honra

O trabalho é individual e deve ser implementado na sua totalidade, sem uso de bibliotecas prontas (fora as bibliotecas padrões - STL **não** deve ser utilizada) ou código de outros (colegas ou não). O trabalho enviado deve representar um esforço honesto em resolver o problema - isto é, não é algo "pela metade", que não implementa funcionalidades essenciais. Violações a esta conduta serão penalizadas e o violador não só terá nota nula neste trabalho, mas também não terá direito a enviar os próximos trabalhos, ficando portanto sem parte da nota da disciplina. Plágios estão sujeitos a sanções administrativas pelo colegiado do curso.

Data de entrega

21 de novembro de 2017.
Cada dia de atraso reduzirá em 10% a nota obtida.