MAC–IME–USP

Carlos Eduardo Ferreira
Sala 108C Tel.: 3091 6079
e-mail cef@ime.usp.br
monitor Leonardo B. Facci
e-mail lbfacci@ime.usp.br

MAC 323 - Estruturas de Dados

Primeiro semestre de 2007

Distância entre palavras – **Entrega: 2 de julho de 2007**

Neste exercício será dado um dicionário de palavras e desejamos, para cada par de palavras dado, encontrar, se existir, um caminho de custo mínimo formado de palavras "vizinhas" que leve uma palavra na outra. Dizemos que duas palavras do dicionário são vizinhas se uma puder ser transformada na outra através de uma das operações abaixo:

- remoção de uma letra: o custo da operação é dado pelo custo da letra removida. Exemplo: as palavras `alta` e `ala` são vizinhas e o custo desta operação é o custo da letra `t`;
- inserção de uma letra: o custo da operação é dado pelo custo da letra inserida. Exemplo: as palavras `cor` e `coar` são vizinhas, e o custo da operação é o custo da letra `a` (observe que esta operação é a inversa da operação acima);
- substituição de uma letra: o custo da operação é dado pelo módulo da diferença dos custos das letras envolvidas na troca. Exemplo: as palavras `rato` e `pato` são vizinhas e o custo da operação é dado pelo módulo da diferença do custo da letra `p` e da letra `r`.

O custo de uma letra é sua posição no alfabeto (a letra "a" custa 1, "b" custa 2 e assim por diante).

Considere agora o exemplo abaixo em que é mostrado um dicionário e um caminho de palavras vizinhas e seu custo:

Exemplo: O dicionário é formado pelas palavras: `gato, pato, lata, aresta, alta, ata, preta, ato, pata, preto, rato, prata`. As palavras `gato` e `preto` estão ligadas por um caminho de palavras do dicionário:

```
gato -> pato -> pata -> prata -> preta -> preto
```

O custo do caminho é 59. Já entre as palavras `ata` e `aresta` não existe caminho possível neste dicionário.

Para resolver eficientemente este problema, vocês deverão implementar as seguintes estruturas:

1. uma tabela de hash para armazenar as palavras do dicionário e fazer buscas eficientemente;
2. as vizinhanças entre as palavras deverão ser armazenadas em listas de adjacências;
3. o algoritmo de busca de um caminho de custo mínimo deverá ser implementado com o uso de uma fila de prioridade (heap).

Você pode considerar que todas as palavras do dicionário são formadas apenas por letras minúsculas e a maior palavra possível tem 27 letras ("inconstitucionalissimamente"). O dicionário pode ter até 5000 palavras.

A entrada do seu programa será dada por um inteiro n seguido de n palavras (uma por linha). Em seguida, é dado um inteiro m e, nas linhas seguintes, m pares de palavras, um por linha, para os quais se deseja achar um caminho mínimo como descrito acima. Na saída você deve responder se tal caminho não existe, ou mostrar o caminho e seu custo. Depois de cada consulta seu programa deverá imprimir uma linha em branco.

Note que a construção das listas de adjacências pode ser bastante demorada. Pense em como verificar eficientemente, usando sua tabela de hash, se uma palavra tem ou não vizinhos no dicionário.

Exemplo de entrada

```
12
gato
pato
lata
aresta
alta
ata
preta
ato
pata
preto
rato
prata
2
gato preto
ata aresta
```

Exemplo de saída

```
O menor caminho entre gato e preto tem custo 59
gato -> pato -> pata -> prata -> preta -> preto

Não existe caminho entre ata e aresta
```