Universidade Federal da Bahia
Instituto de Computação
MATA54 - Estruturas de Dados e Algoritmos II
Segundo Trabalho Prático
Prof. Flávio Assis

Semestre 2022.2 - 31 de outubro de 2022

Dicionário

1 Descrição Geral do Trabalho

Neste trabalho deve ser implementado um programa que lerá dados sobre um dicionário e permitirá que operações sejam feitas sobre ele. O programa manterá traduções de um conjunto de palavras em um idioma, chamado de *idioma origem*, em outro idioma, chamado de *idioma destino*, e informação sobre a classe morfológica (substantivo, adjetivo ou verbo) das palavras.

O programa deve organizar as palavras em uma árvore PATRICIA, considerando as suas representações binárias. Nas folhas da árvore devem ser armazenadas referências aos locais, em arquivo, onde cada palavra e os dados sobre ela estão armazenados.

ATENÇÃO: A árvore poderá ser mantida em memória principal, mas contendo apenas, além dos nós internos com indicações de bits em que as palavras se diferenciam, as referências aos locais de armazenamento das palavras e seus dados em arquivo. As palavras em si, traduções e indicações de classe morfológica devem ser mantidas **apenas** em arquivo.

2 Formato da Entrada e Saída

A entrada constará de uma sequência de comandos, de acordo com os formatos abaixo:

1. **insere tradução:** este comando conterá inicialmente uma linha contendo a letra 'i'. Em seguida, haverá uma linha contendo uma palavra no idioma origem. Na linha seguinte haverá uma das letras 's', 'a' ou 'v', indicando, respectivamente, que a palavra se trata de um substantivo, adjetivo ou verbo. A linha seguinte conterá um número inteiro, que indica o número de traduções para a palavra no idioma destino. As traduções aparecerão nas linhas seguintes, uma por linha. Considere que o tamanho máximo de uma palavra no idioma origem é de 30 caracteres. Cada tradução pode ser uma palavra ou um conjunto de palavras, mas cada tradução terá, no máximo, 50 caracteres (incluindo espaços). Considere que o número máximo de traduções é 10.

   Caso a palavra já exista no dicionário, o programa deve gerar a sequência de caracteres *'palavra ja existente:'*, seguida de um espaço, seguido da palavra no idioma origem. Caso a palavra não exista, o comando a adicionará no dicionário e gerará, na saída, a sequência de caracteres *'palavra inserida no dicionario:'*, seguida de um espaço, seguido da palavra.

2. **lista palavras no idioma origem:** este comando consistirá de uma linha contendo a letra 'l', seguida de outra linha, contendo ou a letra 'c' ou 'd'.

   Este comando lista as palavras do idioma origem. Se a letra na segunda linha for 'c', as palavras devem ser mostradas na ordem crescente. Se for 'd', as palavras devem ser mostradas na ordem decrescente. As palavras devem ser listadas uma por linha, sem espaço após cada palavra. Caso o dicionário esteja vazio, o comando não gera saída (nem espaço em branco).

3. **lista traduções:** este comando consistirá de uma linha contendo a letra 't', seguida de outra linha, contendo uma palavra do idioma origem.

   Se a palavra não existir no dicionário, o programa deve gerar, na saída, a sequência de caracteres *'palavra inexistente no dicionario:'*, seguida de um espaço, seguido da palavra no idioma origem. Se a palavra existir, o comando gera na saída a sequência de caracteres *'traducoes da palavra:'*, seguida de um espaço, seguido da palavra no idioma origem. Em seguida, o comando apresenta as traduções da palavra, uma por linha.

4. **lista palavras por classe:** este comando consistirá de uma linha contendo a letra 'a', seguida de outra linha contendo a letra 's', 'a' ou 'v', seguida de outra linha contendo a letra 'c' ou 'd'.

   Esse comando deve retornar as palavras do idioma origem que são substantivos, adjetivos ou verbos, a depender de a segunda letra ser, respectivamente, 's', 'a' ou 'v'. As palavras retornadas devem estar em ordem crescente, se a terceira letra for 'c', ou decrescente, se esta letra for 'd'. As palavras devem ser apresentadas uma por linha. Caso não haja palavras, o comando não gera saída.

5. **consulta classe de palavra:** este comando consistirá de uma linha contendo a letra 'c', seguida de outra linha, contendo uma palavra.

   Se a palavra não existir no dicionário, o programa deve gerar, na saída, a sequência de caracteres *'palavra inexistente no dicionario:'*, seguida de um espaço, seguido da palavra no idioma origem. Se a palavra existir, o comando gera na saída a sequência de caracteres *'classe da palavra:'*, seguida de um espaço, seguido da palavra no idioma origem, seguida de dois pontos (*':'*), seguidos de um espaço, seguido de uma das seguintes sequências de caracteres, a depender da classe da palavra: *'substantivo'*, *'adjetivo'* ou *'verbo'*.

6. **remove palavra:** este comando consiste de uma linha contendo a letra 'r', seguida de outra linha, contendo uma palavra.

   Se a palavra não existir no dicionário, o programa deve gerar, na saída, a sequência de caracteres *'palavra inexistente no dicionario:'*, seguida de um espaço, seguido da palavra no idioma origem. Se a palavra existir, o comando remove a palavra e suas traduções do dicionário e gera na saída a sequência de caracteres *'palavra removida:'*, seguida de um espaço, seguido da palavra no idioma origem,

7. **imprime árvore:** este comando consiste apenas de uma linha, contendo a letra 'p'. O formato de impressão da página está descrito na seção 4.

8. **término da sequência de comandos:** a sequência de comandos será terminada por uma linha com a letra 'e'.

3 Observações sobre a entrada e saída

- O programa deve ler da entrada padrão e escrever os resultados na saída padrão.
- A entrada somente conterá caracteres minúsculos, sem acento e sem cedilha.
- Não pode haver espaço entre linhas na saída. A saída deve apresentar os caracteres em letras minúsculas e sem acento.
- O programa deve manter as atualizações em arquivo. A correção levará em consideração que o estado dos dados é persistente. Com isto, um teste pode ser feito, por exemplo, inserindo-se dados, terminando a execução do programa e fazendo uma consulta a esses dados em nova invocação do programa. Neste caso os dados devem ainda estar no arquivo.

- Lembre-se de que é assumido que a memória principal é insuficiente para armazenar todos os dados. Portanto, por exemplo, uma implementação que mantém a estrutura do arquivo em memória principal e a salva por completo no arquivo será considerada inaceitável. Para esse trabalho, será aceito que a árvore PATRICIA seja mantida na memória principal, como indicado acima. A árvore em si não precisa ser armazenada em arquivo. Ela pode ser novamente construída a cada invocação do programa.
- Os arquivos utilizados devem ser armazenados em formato binário.
- Todo arquivo criado pelo programa para armazenar dados deve conter extensão *'.dat'*.
- O programa não deve gerar nenhum caractere a mais na saída, além dos indicados acima. Em particular, o programa não deve conter menus.

4 Impressão da árvore PATRICIA

Os valores referentes a cada nó da árvore devem ser apresentados, um por linha, seguindo um percurso em *pré-ordem* da árvore.

Para cada nó interno, os dados devem ser apresentados no seguinte formato: a sequência de caracteres *'bit:'*, seguida de um espaço, seguido do valor armazenado no nó, seguido de um espaço, seguido pela sequência de caracteres *'fesq:'*, seguida por um espaço, seguido pelo valor armazenado no filho à esquerda, seguido por um espaço, seguido pela sequência de caracteres *'fdir:'*, seguida por um espaço, seguido pelo valor armazenado no filho à direita. Quando um filho for um nó folha, o valor a ser apresentado, referente àquele nó, será a palavra referenciada pelo nó.

Quando um nó for folha, deve-se gerar na saída apenas a palavra correspondente ao nó (ou seja, a palavra para a qual há uma referência no nó folha).

Após a impressão dos nós da árvore, o programa deve gerar na saída as palavras armazenadas no dicionário, uma por linha, em ordem alfabética crescente. Cada linha deve conter a palavra, seguida de um espaço, seguida de sua representação binária, separando-se as representações dos caracteres por um espaço.

5 Observações Adicionais

- Trabalho individual.
- Data de entrega: **27/11/2022 - prazo inadiável! Somente serão aceitos trabalhos entregues até essa data, até 23:59h!**
- Linguagens de programação permitidas: C, C++, Java ou Python.
  - **Importante:** Para as linguagens C, C++ e Java, somente trabalhos feitos utilizando-se os seguintes compiladores serão aceitos:
    - C: gcc ou djgpp
    - C++: g++ ou djgpp
    - Java: compilador java do JDK (mais recente)

  No caso de Python, deve ser indicada a versão utilizada.

  Não serão compilados trabalhos em outros compiladores! Erros ocasionados por uso de diferentes compiladores serão considerados erros do trabalho!
- O aluno deverá submeter seu trabalho através do *moodle*.