



Primeiro Dicionário do Samuel

Samuel de apenas 8 anos tem um sonho - ele deseja criar o seu próprio dicionário. Isto não é uma tarefa fácil para ele, pois conhece poucas palavras. Bem, ao invés de pensar nas palavras que sabe, ele teve uma ideia brilhante. A partir do seu livro de histórias favorito, ele quer criar um dicionário com todas as palavras distintas que existem no livro, além disso Samuel quer que as palavras no dicionário estejam em ordem alfabética decrescente, ou seja, das palavras começadas com letra Z para palavras começadas com a letra A. É claro, isso é uma tarefa que toma bastante tempo e, por conta disso, ele decidiu que deveria pedir ajuda de um programador. Você foi convidado a escrever um programa que liste todas as diferentes palavras que existem no livro preferido do Samuel, as palavras do livro estarão armazenadas em arquivo texto. Considere que uma palavra é definida como uma sequência de letras, maiúsculas ou minúsculas, palavras com apenas uma letra também deverão ser consideradas. Ademais o seu programa deverá ser "CaSe InSeNsItIvE", por exemplo, palavras como "Apple", "apple" ou "AP-PLE" deverão ser consideradas como mesma palavra. Para garantir que não tenhamos palavras repetidas no dicionário do Samuel, para cada palavra lida no arquivo texto deve feita uma busca no dicionário, caso a palavra já conste no dicionário a palavra lida deve ser descartada, caso contrário a palavra deve ser inserida no dicionário de forma a manter a ordem alfabética decrescente.

Importante:

- Para armazenar o dicionário de palavras use um vetor de strings. Considere que no arquivo texto não teremos mais de 100 palavras diferentes, ou seja, você pode declarar um vetor de strings com 100 posições.
- A busca no dicionário deve ser implementada utilizando uma **função recursiva (sem loops)** que faz **busca binária** no vetor de palavras (vetor de strings).
- Para inserir uma nova palavra no vetor de String use uma **função recursiva (sem loops)**.
- **Importante**, não é para inserir todas as palavras no vetor e depois ordenar, e sim a cada palavra nova, esta deverá ser inserida no vetor de palavras que continuará ordenado.



Entrada do programa:

O arquivo de entrada contém várias linhas de texto, cada uma delas com várias palavras separadas por um espaço em branco (), o fim da entrada é determinado por um ponto (".") em uma linha com somente esse caractere. Exemplos de arquivo de entrada.

```
texT
```

```
Adventures in Disneyland
```

```
Two blondes were going to Disneyland when they came to a fork in the  
road The sign read disneyland LEFT
```

```
So they went home
```

```
.
```



Saída do programa:

Você deve gerar um arquivo de saída (dicionario.txt) contendo uma lista de diferentes palavras que aparecem no texto, uma palavra por linha. Todas as palavras devem ser escritas com letras minúsculas, em ordem alfabética decrescente. No final da lista você deve informar quantas palavras diferentes existem no texto. Saída do programa:

```
when
were
went
two
to
they
the
text
so
sign
road
read
left
in
home
going
fork
disneyland
came
blondes
adventures
a
total de palavras diferentes no dicionario: 22
```



Observações importantes:

O trabalho será avaliado de acordo com os seguintes critérios de avaliação:

- Funcionamento do programa;
- O programa deve estar na linguagem C.
- O quão fiel é o programa quanto à descrição do enunciado;
- Indentação, comentários e legibilidade do código;
- Clareza na nomenclatura de variáveis e funções;
- Não é permitido o uso de variáveis globais (static).
- A atividade deve ser feita individualmente ou em dupla e deve-se, necessariamente, ser entregue pelo Moodle, não serão aceitos trabalhos por e-mail.