34. Exercícios

DIM0321

2015.1

1. Escreva um programa para agrupar anagramas de um dicionário numa linha. O programa deve conter 3 etapas:

   (a) `sign`

   (b) `sort`

   (c) `squash`

   Cada etapa lê os dados da etapa anterior e cria um novo arquivo com o resultado dela escrito.

   Fora das funções de manipulação de arquivos, pode-se usar as funções da biblioteca padrão como:

   ```
   void qsort(void *base, size_t nmemb, size_t size,
              int (*compar)(const void *, const void *));
   int strcmp(const char *s1, const char *s2);
   char *strcpy(char *dest, const char *src);
   size_t strlen(const char *s);
   ```

   Pode ainda supor que nenhuma palavra contem mais do que 100 letras e não tem mais do que 1024 palavras. Esses valores já são definidos para você no arquivo `anagrams.c`

   ```
   #define CHARMAX 100
   #define WORDMAX 1024
   ```

   O objetivo é escrever as 3 funções:

   ```
   int sign(const char *fname_in, const char* fname_out);
   int sort(const char *fname_in, const char* fname_out);
   int squash(const char *fname_in, const char* fname_out);
   ```

1. `sign` vai calcular a assinatura de cada palavra é colocá-la antes da palavra original num novo arquivo.

2. `sort` vai ordenar esse arquivo com as assinaturas a partir das assinaturas. Assim todas as palavras de mesma assinatura vão ficar perto uma da outra.

3. `squash` vai criar grupos de palavras de mesma assinatura a partir da etapa anterior. Um grupo corresponde então aos anagramas e cada linha do arquivo final armazena um grupo de anagramas.

   Atenção! Cada etapa cria um novo arquivo a partir do arquivo gerado na etapa anterior. Não esqueça de abrir e fechar corretamente os arquivos usados.

O dicionário original é dado no arquivo `words.txt`. Um dicionário mais simples (para testes) é dado no arquivo `words2.txt`.

Esse dicionário contem as palavras:

```
debar
sabre
bears
baser
bread
barde
bared
bares
```

A etapa 1 cria um arquivo cujo conteúdo deve ser:

```
abder debar
abers sabre
abers bears
abers baser
abder bread
abder barde
abder bared
abers bares
```

A partir deste arquivo, a etapa 2 cria um arquivo com :

```
abder debar
abder bread
abder barde
abder bared
abers sabre
abers bears
abers baser
abers bares
```

A última etapa vai criar um último arquivo:

```
debar bread barde bared
sabre bears baser bares
```

1. Escreva um programa que:
   - abre um arquivo texto existente
   - conta o número de vogais existentes neste arquivo
   - imprime a quantidade de cada vogal na tela
2. Escreva um programa que mescla o conteúdo de dois arquivos texto em um arquivo texto resultante. Para gerar o arquivo resultante, devemos alternar o uso das palavras dos arquivos de origem:
   - Cada par de palavras do arquivo resultante deve ser composto de:

  - uma palavra de um arquivo de origem
  - uma palavra do outro arquivo de origem
- Se um dos arquivos de origem contiver menos palavras que o outro arquivo, esse outro arquivo deve ser descarregado diretamente no arquivo resultante.