Programação Imperativa 2021/2022 (CC1003), DCC/FCUP

Folha 11

11.1 Escreva um programa lê um dado número de linhas da entrada padrão e escreve tais linhas num ficheiro de texto. Exemplo:

```
Número dei linhas: 4
linha 1
linha 2
linha 3
linha 4
```

11.2 Escreva um programa que lê o ficheiro de texto gerado no exercício anterior e que guarda as linhas do ficheiro num vetor.

11.3 Escreva uma função `int numero_linhas(char *nome_ficheiro)` que retorna o número de linhas dum ficheiro.

11.4 Escreva um programa para contar palavras dum ficheiro. Considere que as palavras são sequências de "carateres normais" (e.g., letras, algarismo ou sinais de pontuação) separados por "carateres brancos" (espaços, tabulação ou mudança de linha, ou seja, `' '`, `'\t'`, ou `'\n'`).

11.5 Escreva um programa que lê o conteúdo de dois ficheiros de texto, "ficheiro1.txt" e "ficheiro2.txt", e escreve um novo ficheiro (de nome "uniao.txt") obtido unindo os conteúdos dos dois ficheiros originais.

11.6 Escreva um programa capaz de receber do utilizador um dado número de palavras (a introdução de palavras termina com uma palavra vazia), armazenando-as num ficheiro de texto (uma em cada linha), de nome "palavras.txt". Para além disso, desenvolva também as seguintes funções:

1. Escreva uma função `void maior(char *nome_ficheiro)` que imprime na saída padrão a palavra com mais carateres existente no ficheiro.
2. Escreva uma função `void par_impar(char *nome_ficheiro)` que escreve dois novos ficheiros. O primeiro ficheiro (de nome "par.txt") vai conter todas as palavras do ficheiro original com tamanho par. O segundo ficheiro (de nome "impar.txt") vai conter todas as palavras do ficheiro original com tamanho impar.
3. Escreva uma função `void ad_palavra(char *nome_ficheiro, char *palavra, int pos)` que adiciona a palavra `palavra` na linha número `pos`. Por exemplo, se for pretendido inserir `palavra` na linha 3, todas as palavras da linha 3 ou em linhas superiores devem avançar uma linha.
4. Escreva uma função `void rem_palavra(char *nome_ficheiro, char *palavra)` que procura a palavra `palavra` no ficheiro e remove a linha correspondente se a palavra existir no ficheiro.

5. Escreva uma função `void sem_repetidos(char *nome_ficheiro)` que escreve um novo ficheiro (de nome "sem_repetidos.txt") que contenha todas as palavras existentes no ficheiro original sem repetições.

6. Escreva uma função `void palindromos(char *nome_ficheiro)` que escreve um novo ficheiro (de nome "palindromos.txt") que contenha todas as palavras existentes no ficheiro original que sejam palíndromos (ou seja, cuja leitura é igual começando do início para o fim ou em sentido inverso).

7. Escreva uma função `void cifrar(char *nome_ficheiro, int n)` que escreve um novo ficheiro (de nome "cifrado.txt") que contenha uma versão cifrada de todas as palavras existentes no ficheiro original, usando a cifra ROT13 descrita na aula 7. A cifra deve ser aplicada a letras minúsculas e maiúsculas. Eventuais outros tipos de carateres devem ficar inalterados.

8. Escreva uma função `void ordenar(char *nome_ficheiro)` que ordena as palavras no ficheiro por ordem alfabética.

Sugestão: pode utilizar um vetor de apontadores para armazenar as palavras.