Linguagens de Programação
Departamento de Engenharia Eletrônica e de Computação - DEL-Poli/UFRJ
Prof.: Miguel Elias Mitre Campista

== Laboratório 1: Exercícios de Perl e Lua ==

Para esquentar...

Execute no terminal os seguintes programas para verificar se os interpretadores estão funcionando corretamente:

```
perl –e 'print "hello\n";'

lua –e 'print "hello"'
```

Caso tudo ocorra bem, faça os seguintes exercícios:

1. Dado um arquivo de texto onde as linhas ímpares possuem o nome de uma pessoa e as linhas pares o sobrenome correspondente ao nome da linha anterior, escreva um programa em Lua que leia todas as linhas do arquivo, armazene as informações em uma tabela e imprima a tabela resultante na tela. Note que cada elemento da tabela é um par chave/valor igual ao par nome/sobrenome de cada contato. Utilize o for genérico e a função iteradora **`io.lines (nomedoarquivo)`** para ler as linhas do arquivo. Modularize o código.

2. Repita o exercício anterior em Perl.

3. Ordene uma sequência de inteiros utilizando o algoritmo Quicksort. Para isso, escreva um programa em Lua. O pseudocódigo é o seguinte:

```
quickSort (array, indice_inicio, indice_fim)
        i, j ← indice_inicio, indice_fim
        pivo ← elemento do meio do array
        enquanto i <= j faça
                enquanto array [i] < pivo faça
                        incrementa i
                   fim
                enquanto array [j] > pivo faça
                        decrementa j
                   fim
                se i <= j então
                      troca array [i], array [j]
                     incrementa i
                             decrementa j
                fim
        fim
        se j > indice_inicio então
                   quickSort (array, indice_inicio, j)
        fim

              se i > indice_fim então
                    quickSort (array, i, indice_fim)
        fim
fim
```

4. Escreva um programa em Perl para obter as seguintes informações de um arquivo de texto: Número de linhas, número de palavras, número total de caracteres, número de caracteres diferentes de espaço em branco. Utilize a função **`length`** para contar o número de caracteres de uma string e a função **`split`** para inserir uma linha de texto em um array contendo todas as palavras da linha separadas. Modularize o código.

5. Modifique o programa do exercício anterior para detectar padrões de interesse. Utilize detecção de expressão regular.

.	a single character
\s	a whitespace character (space, tab, newline,...)
\S	non-whitespace character
\d	a digit (0-9)
\D	a non-digit
\w	a word character (a-z, A-Z, 0-9, _)
\W	a non-word character
[aeiou]	matches a single character in the given set
[^aeiou]	matches a single character outside the give set
(foo|bar|baz)	matches any of the alternatives specified
^	start of string
$	end of string
*	zero or more of the previous thing
+	one or more of the previous thing
?	zero or one of the previous thing
{3}	matches exactly 3 of the previous thing
{3,6}	matches between 3 and 6 of the previous thing
{3,}	matches 3 or more of the previous thing