

Ficha 4 – Exercícios

Exercícios de frequências de anos anteriores

Tópicos abordados:

- Comandos Unix
- Programação em PERL (até vectores)

Nota: as perguntas do capítulo 2 (PERL) foram ajustadas para a matéria abordada nas aulas práticas.

Duração prevista: 1 aulas

©2009: {rui, patricio, nuno.costa}@estg.ipleiria.pt

1 Comandos Unix

Para cada uma das alíneas indique a linha de comandos a empregar para obter o resultado pedido:

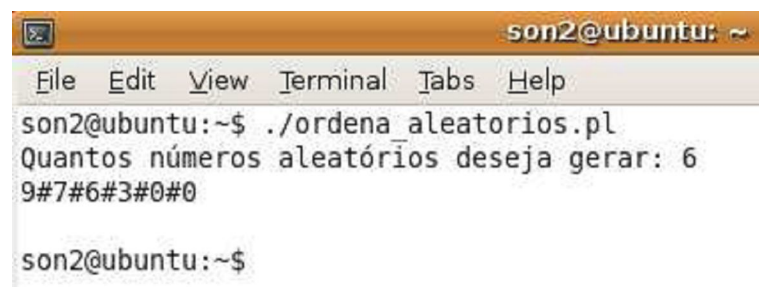
- Indique o caminho completo do comando `date`.
- Mostre todas as ligações (caminho e nome) existentes na sua directoria de trabalho.
- Devolva de forma ordenada o tamanho dos ficheiros da directoria corrente.
- Liste todas as *strings* de permissões dos ficheiros da directoria local que terminem por “x”.
- Devolva a terceira linha a contar do fim (e apenas essa linha) do ficheiro `/etc/passwd`.

- f) Acrescente a permissão de escrita para os “outros” a todos os ficheiros do directório corrente, cujo nome se inicie por A ou B ou C e se termine por xt
- g) Liste de todas as *strings* de permissões (por exemplo: `rw- r-x r-x`) dos ficheiros do directório `/bin`. Não deverão existir *strings* repetidas, nem linhas em branco. Por exemplo, na máquina virtual, o resultado a obter será o seguinte:
- ```
lrwxrwxrwx
-rwsr-xr-x
```
- h) Execute o comando `ps aux`, com a saída padrão enviada para o ficheiro `MAQUINA_SEGUNDOS_psaux.txt`. `MAQUINA` deve corresponder ao nome da máquina corrente e `SEGUNDOS` ao número de segundos decorridos desde 1 Janeiro de 1970.
- i) Mostre a data corrente, num formato idêntico a este: “05\_de\_Maio\_2008---14h11:35.123456789”, sendo que a última parte corresponde a nano segundos (o nome do mês poderá também aparecer em língua Inglesa).
- j) Liste o conteúdo do ficheiro `/boot/config-2.6.20-15-generic`, **não** mostrando as linhas que se iniciem por “#” nem as linhas em branco.
- k) Copie o ficheiro `/boot/config-2.6.20-15-generic` para o directório corrente e alterar as permissões da cópia para que passem a ser: **`rw- --- rw-`**

## 2 PERL

Recorrendo à linguagem PERL, implemente os seguintes programas:

- a) Recorrendo à linguagem PERL, implemente o script `ordena_aleatorios.pl` cujo objectivo é pedir ao utilizador quantos números aleatórios ele pretende que sejam gerados. Os números gerados deverão estar compreendidos entre 0 e 9. No final deverão ser mostrados na consola todos os números gerados, por ordem decrescente e separados por #.



```
son2@ubuntu: ~
File Edit View Terminal Tabs Help
son2@ubuntu:~$./ordena_aleatorios.pl
Quantos números aleatórios deseja gerar: 6
9#7#6#3#0#0
son2@ubuntu:~$
```

- b) Recorrendo à linguagem PERL, implemente o script `multiplica.pl`, cujo propósito é o de multiplicar os números existentes num ficheiro de texto (a ser criado segundo as próximas regras).

O formato do ficheiro de números permite que possam existir linhas com apenas um número e linhas com vários números, sendo que neste último caso, os números encontram-se separados pelo carácter ":".

No final, o script deve mostrar no terminal, quantas linhas e quantos números foram processados bem como o produto dos números. Adicionalmente, e como optimização, se o script detectar a existência do número 0 (zero) deve terminar de imediato escrevendo "[AVISO] 0 detectado" para o terminal.

Como exemplo de ficheiro de números considere o ficheiro `nums.txt`, mostrado de seguida:

```
2
3:1:14.11
2:2:2
```

Sendo que quando executado da seguinte forma: `./multiplica.pl` o script produz a seguinte saída:

```
Linhas processadas: 3
Números processados: 7
Produto: 677.28
```

- c) Recorrendo à linguagem PERL, implemente o script `multiplos_de_N.pl`, cujo propósito é o de identificar os números múltiplos do número inteiro N de entre uma lista de números obtida de um ficheiro, no formato de um número inteiro por linha (utilizar comando `seq` para criar o ficheiro).

O número N deve ser pedido ao utilizador, sendo que deverá ser superior a 1.

Quando o script detecta que um dado número é múltiplo de N, deve mostrar no terminal o número e a indicação que corresponde ao produto de M por N.

Considere o seguinte exemplo como a saída do programa:

```
Introduza um número: 2
2 >> 2x1
4 >> 2x2
6 >> 2x3
```

NOTA: o ficheiro utilizado foi criado com o comando: `seq 6`

- d) Recorrendo **somente** à linguagem PERL, implemente o script “`cut.pl`” cujo propósito é o de ser uma versão simplificada do utilitário “`cut`” do Unix. Deste modo, o script “`cut.pl`” deve suportar a opção “`-f`” e a opção “`-d`”. Estes parâmetros devem ser pedidos ao utilizador, bem como o ficheiro a ser processado (o ficheiro tem que existir em disco).

ATENÇÃO: não é possível o recurso a comandos e utilitários do Unix.