Exercicis pràctica 2

7 de març de 2019

En aquest document es mostren els exercicis a programar. Per tal d'intentar seguir la "filosofia" dels scripts a Unix, els arguments es passen per línia de comandes i la sortida s'imprimeix per pantalla.

Tots els scripts necessiten arguments per línia de comandes. En cas que no s'especifiquin, en executar, el nombre necessari d'arguments l'script haurà de mostrar un missatge d'error per pantalla i sortir amb un codi d'error (exit 1). En cas que s'especifiquin tots els arguments, podeu suposar que s'han especificat de forma correcta. Si l'script s'executa correctament, caldrà que aquest surti amb un codi d'execució correcta, exit 0.

Per exemple, suposeu un script en què s'han d'especificar dos arguments: el primer argument ha de ser un número i el segon una cadena. Si no s'especifiquen els dos arguments, caldrà donar un missatge d'error per pantalla i sortir de l'script amb el codi d'error. Si s'especifiquen els dos arguments, podeu suposar que el primer argument és un número i el segon una cadena.

Tots els noms de fitxers i directoris a tractar no tindran espais ni caràcters especials i, a més, podeu suposar que tots els fitxers a tractar són fitxers de text pla. En alguns d'aquests exercicis es passa com a paràmetre un directori: l'script ha de funcionar independement del directori que es passi com a paràmetre. També podeu suposar que no hi ha enllaços simbòlics en els fitxers analitzats. En altres paraules, només hi haurà fitxers o directoris.

La puntuació de cada exercici és de 2 punts. Aquests exercicis es poden fer amb les instruccions que hi ha al document de l'enunciat de la pràctica 2. Es poden fer servir totes les opcions disponibles, encara que no estiguin explicades al tutorial. També es permetrà fer servir comandes no especificades en els tutorials. Caldrà però justificar i explicar el funcionament d'aquestes comandes a l'script (fent servir comentaris).

Exercici 1:

Es demana implementar un script exercici1.sh que rep dos arguments, un directori i una extensió. L'script buscarà de forma recursiva tots els fitxers amb l'extensió especificada i els ordenarà segons la seva mida (en bytes) de menor a major. Finalment escriurà per pantalla només el nom dels 10 primers fitxers.

En cas que es faci servir un bucle per implementar l'script, la puntuació màxima del problema serà de 1,5 punts. La puntuació màxima també serà de 1,5 en cas que s'utilitzin fitxers (temporals) per emmagatzemar resultats intermedis. En altres paraules, feu servir canonades (i no pas redirecció) per implementar aquest exercici!

Exemple d'execució

```
./exercici1.sh gutenberg/ txt
gutenberg/copyright.txt
gutenberg/donate-howto.txt
gutenberg/etext90/bill11.txt
gutenberg/etext98/2ws2710.txt
gutenberg/etext90/liber11.txt
gutenberg/etext05/bib0010h/Readme.txt
gutenberg/etext04/kknta10.txt
....
```

Exercici 2:

Fer un script anomenat exercici2.sh que rep tres arguments: un directori, una extensió i una cadena. L'script ha d'analitzar, de forma recursiva, els fitxers dintre del directori amb la extensió corresponent. Per a cada fitxer trobat, cal imprimir el nom complet del fitxer (`path`) i analitzar quants cops hi apareix la cadena dins del fitxer, ja sigui com a paraula individual o com a part d'una altra.

La puntuació màxima serà de 1,5 en cas que s'utilitzin fitxers (temporals) per emmagatzemar resultats intermedis. Es pot fer servir com a màxim un bucle per implementar l'script.

Exemple d'execució

```
./exercici2.sh gutenberg/etext00/ txt science
gutenberg/etext00/00ws110.txt
la cadena science apareix 126 vegades
gutenberg/etext00/1cahe10.txt
la cadena science apareix 59 vegades
gutenberg/etext00/1vkip11.txt
la cadena science apareix 16 vegades
gutenberg/etext00/2cahe10.txt
la cadena science apareix 113 vegades
gutenberg/etext00/2yb4m10.txt
la cadena science apareix 4 vegades
gutenberg/etext00/8rbaa10.txt
...
```

Exercici 3:

Fer un script anomenat exercici3.sh que rep un argument, un directori, que podeu suposar que existeix. L'script ha de calcular de forma recursiva el nombre total fitxers que conté així com el seu pes total, és a dir, la suma de la mida de tots el fitxers en bytes. Cal evitar que se sumin la mida dels directoris i podeu suposar que no hi a cap altre tipus d'element (com enllaços simbòlics).

En cas que es faci servir un bucle per implementar l'script, la puntuació màxima del problema serà de 1,5 punts. La puntuació màxima també serà de 1,5 en cas que s'utilitzin fitxers (temporals) per emmagatzemar resultats intermedis.

Exemple d'execució

```
./exercici3.sh gutenberg/etext00/
47 files
33974281 bytes
```

(continua a la pàgina següent)

Exercici 4:

Es demana fer un script que accepti un argument que serà un usuari del sistema. Per exemple, l'usuari `root`. Mitjançant la comanda `ps aux` haureu de trobar el processos d'aquest usuari i calcular el total de memòria que està ocupada per aquests processos. La memòria ocupada per un procés es pot saber fent servir la columna VSZ, que està associada a la memòria (en bytes) que ocupa el procés a l'espai de memòria virtual, així com la columna RSS, que indica quants bytes està fent servir en aquell moment el procés a memòria física RAM.

En cas que es faci servir un bucle per implementar l'script, la puntuació màxima del problema serà de 1,5 punts. La puntuació màxima també serà de 1,5 en cas que s'utilitzin fitxers (temporals) per emmagatzemar resultats intermedis.

Exemple d'execució

```
./exercici4.sh root
VSZ: 6286016 bytes
RSS: 479416 bytes
```

Exercici 5:

En l'entorn de la informàtica és habitual treballar amb fitxer del tipus CSV (comma-separated values) o TSV (tab-separated values). En aquest cas haureu de crear un script que donat el fitxer alumnes.csv, de tipus CSV, hi permeti buscar un alumne i permeti modificar el fitxer per tal de canviar el grup pertinent del alumne, sigui del grup de problemes o de pràctiques. En concret, el primer paràmetre de l'script serà el niub de l'estudiant, el segon paràmetre serà el paràmetre "problemes" o "practiques" depenent de si es vol modificar el grup de problemes o de pràctiques de l'estudiant, el tercer paràmetre serà el nou grup assignat a l'estudiant. Podeu suposar que el niub especificat és correcte i existeix al fitxer CSV.

En cas que es faci servir un bucle per implementar l'script, la puntuació màxima del problema serà de 1,5 punts. La puntuació màxima també serà de 1,5 en cas que s'utilitzin fitxers (temporals) per emmagatzemar resultats intermedis.

Per separar els elements del fitxer CSV entre sí es pot fer servir l'eina awk. Mitjançant l'opció -F es pot establir un separador diferent que el de per defecte. En el següent cas indiquem que el separador és un punt i coma ";": `awk -F ";" '{print $X}'`, on $X és la columna que volem extreure. A l'hora de canviar l'alumne de grup de pràctiques o de problemes caldrà mantenir el separador ";" com a separador entre columnes, ja que el punt i coma és un caràcter que el sistema utilitza habitualment per separar columnes en un fitxer CSV.

Exemple d'execució

```
./exercici5.sh niub16884276 problemes F00
Grup de problemes modificat.
./exercici5.sh niub16884276 practiques AB0
Grup de pràctiques modificat.
```