PRA2

Guions en bash

Sistemes Operatius

Curs 2020-21

13/GEN/2021

Ton Llucià Senserrich
47125160T

Eduard Sales Jové
49539818A

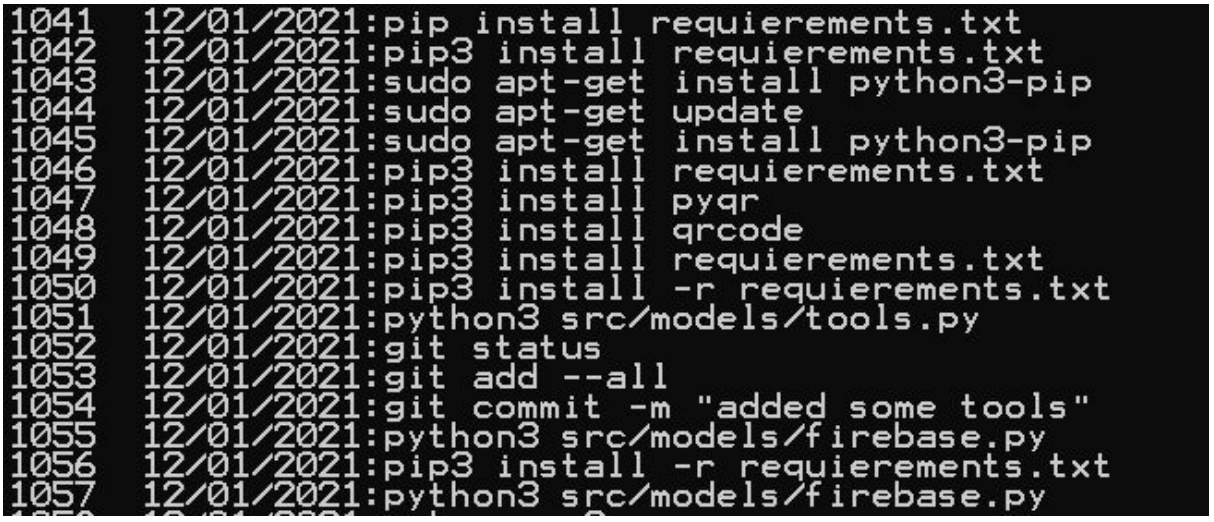
3.1. Comanda history:

History és una comanda útil en la interacció amb l'intèrpret de comandes.

1.- Cerqueu informació sobre aquesta comanda.

Expliqueu què fa:
La comanda history mostra per pantalla, de forma ordenada i numerada, totes les comandes executades anteriorment.
També ens permet fer referència a comandes executades anteriorment.

Doneu exemples d'execució.



Associada a la comanda history hi ha la variable HISTTIMEFORMAT:

2.- Cerqueu informació sobre aquesta variable.

De quin tipus és?
HISTTIMEFORMAT és de tipus string, ja que simplement conte el format que ha d'adoptar la data de la sortida.

Per a què serveix?
De la mateixa forma que la comanda history mostra de forma numerada les comandes executades anteriorment, amb **HISTTIMEFORMAT** mostrarà també quan es van executar, mostrant tant l'any, dia, mes i l'hora, en aquest ordre.

3.- Modifiqueu un dels fitxers d'arrancada del shell per establir-la a tots els vostres intèrprets de comandes.

Quin fitxer heu modificat?

Podríem modificar diversos fitxers, depenen de si volem que es produeixi només en el nostre usuari o en varis, entre altres opcions:

1) ~/.bashrc → scrip definit per l'usuari
2) /etc/bashrc → script global
3) ~/.bash_profile → scrip definit per l'usuari
4) ~/.bash_login → scrip definit per l'usuari
5) ~/.profile → scrip definit per l'usuari
6) /etc/profile → script global
7) /etc/profile.d/* → directori on resideixen diversos scripts globals.

Amb quin format de data-hora heu establert que surti la data de l'històric?

Hem establert el format dia/mes/any(%d/%m/%Y).

Mostreu la comanda usada:

Echo 'export HISTTIMEFORMAT="%d/%m/%Y: "' >> file

on file representa un dels fitxers anteriorment mencionats.

4.- Aprofiteu aquesta modificació per establir el directori actual de treball (.) al vostre PATH.

Com ho heu fet?

S'ha de modificar la variable d'entorn PATH per tal d'afegir-hi el path del directori actual.

Indiqueu la comanda i el fitxer modificat.

Echo 'export PATH="$(pwd):$PATH"' >> file.

3.2. Un primer script:

Del següent script:

```
#!/dev/bash

if ( $# -ne 3 )
then
       echo "$0 suma els dos nombres passats com a parametres"
       echo "Ús: $0 <nombre1> <nombre2>"
       exit 1
end fi

echo "$1 + $2 = `expr $1 - $2`"
```

1.- Copieu-lo al vostre entorn de treball, anomeneu-lo prac2_2.sh. Aquest guió té error/s de sintaxi, error/s de funcionament i/o error/s de compliment de requeriments.

Cerqueu-los, quins són? Esmeneu-los fins que pugueu executar correctament l'script.

❌ #!/dev/bash

✔ #!/bin/bash

EXP: Perquè la direcció de bash en el nostre sistema ubuntu es aquesta.

❌ if($# -ne 3)

✔ if($# -ne 2)

EXP: El nostre programa sumarà dos nombres, per lo tant no li podem passar més de 2 valors, ja que no els tindrà en compte. En ficar $# -ne 2 sol entrarà dins l'if en cas que ens passin un nombre de números incorrecte.

❌ if ($# -ne 2)

✔ if [$# -ne 2]

EXP: L'if va amb [ ] si el fiquéssim en parèntesis no funcionaria.

❌ end fi

✔ fi

EXP: La comanda if acaba amb fi i en cas de ficar l'end donarà un error.

❌ echo "$1 + $2 = 'expr $1 - $2'"

✔ echo "$1 + $2 = 'expr $1 + $2'"

EXP: estem printant que el primer valor se suma amb el segon, però en veritat s'estan restant, per això li canviem el signe negatiu a positiu, per a que se sumin.

2.- A part de la identificació i correcció dels errors.

Indiqueu els passos que heu hagut de fer per a poder executar-lo (els passos més importants i fonamentals per a poder executar-lo).

Per tal de poder-ho executar-ho en unix, ja que estàvem, treballant en Windows, hem hagut de canviar el format dels salts de línia utilitzant la comanda dos2unix, també hem hagut d'arreglar alguns errors de la sintaxi esmentats en l'apartat anterior, per tal de poder-lo executar.

3.-

Doneu un llistat de les variables que usa, indicant-ne el tipus al qual pertanyen i quina és la seva funcionalitat.

Utilitza les variables d'entorn:

$# que és de tipus enter i conte el nombre d'arguments adquirits pel script.

$0, que és de tipus string i conte el nom del scipt a executar.

$1, $2 que és de tipus enter, en aquest cas, i conte el valor del 1r i del 2n paràmetre respectivament.

4.-

Expliqueu raonadament què fa.

Obté 2 valors enters per paràmetre, comprova que siguin només dos (en les línies 11-16), en cas que no sigui així retorna error, i si tot és correcte els suma i escriu la sortida (línia 17).

5.-

Afegiu 2 solucions alternatives a la darrera comanda del script mantenint la mateixa funcionalitat. És a dir, emprant altres mètodes de fer operacions aritmètiques. Entregueu aquesta versió del script.

```
echo "$1 + $2 = $(($1 + $2))"
let sum=$1+$2
echo "$1 + $2 = $sum"
```

3.3. Nombre de processos dels nivells d'arrancada Sistema Linux:

A les línies 26 i 28, llistarem els fitxers dins del directori obtingut a partir de la carpeta pertanyent al nivell d'arrencada passat per paràmetre, que coincideixin en el tipus de fitxer que estem buscant, en el cas de la línia 26 busquem que comencin amb 'S' i en el cas de la línia 28, amb 'K'.
Seguidament mitjançant una pipe redirigirem la sortida a la comanda wc -l que contarà les línies de la sortida i així obtenim el nombre de scripts d'inici i d'aturada.

3.4. Càlcul binari:

Dins de la funció binary fem un bucle començant amb n igual al valor que volem transformar a binari mentre sigui més gran que zero, i en lloc de restar o sumar, com es faria en un bucle típic, fem l'operació: n >>= 1 per tal de fer un shift cap a la dreta, assignem el resultat a n, i així poder calcular els bits que aniran a l'esquerra del resultat, per tal de calcular aquests valors fem una operació AND entre la variable n i el nombre 1 i concatenem el resultat amb la variable bits, i ho assignem a aquesta mateixa, repetirem aquest procés fins que el nombre n sigui menor que 0 i això haurà posat el nombre binari resultant a la variable global bits.