

Trabajo práctico 1: primeros pasos en LINUX

Sistemas Operativos

Grupo 18:

Chort, Julio Alberto julioch_17@outlook.com

Pacheco Pílan, Federico Ignacio fedepacheco2112@gmail.com

Reynoso, Valentín valenreynoso17@gmail.com

Segundo Cuatrimestre

2020

1. Resoluciones

1.1. Ejercicio 1: comandos básicos:

tiempoVivido.sh
<pre>#!/bin/bash echo "Ingrese la fecha de nacimiento (aaaa-mm-dd):" read unaFechaStr unAnio=\$(date -d \$unaFechaStr +%Y) unMesStr=\$(date -d \$unaFechaStr +%b) unDia=\$(date -d \$unaFechaStr +%d) segsHastaFechaNacimiento=\$(date -d "\$unMesStr \$unDia \$unAnio" +%s) segsDesde1970=\$(date +%s) unosSegundosVividos=\$((segsDesde1970-segsHastaFechaNacimiento)) unosMinutosVividos=\$((unosSegundosVividos/60)) unasHorasVividas=\$((unosMinutosVividos/60)) unosDiasVividos=\$((unasHorasVividas/24)) unaEdad=\$((unosDiasVividos/365)) echo "Edad = \$unaEdad" echo "Dias vividos = \$unosDiasVividos" echo "Horas vividas = \$unasHorasVividas" echo "Minutos vividos = \$unosMinutosVividos" echo "segundos vividos = \$unosSegundosVividos"</pre>

1.2. Ejercicio 2: gestión de sistema de archivos y directorios:

testArch.sh
<pre>#!/bin/bash if [[-d \$1]] then echo "\$1 es un directorio." ls \$1 echo "Tamanio1: \$(du -bs \$1)" # b: bytes, s: summarize echo "Tamanio2: \$(du -hs \$1)" # h: human readable else if [[-f \$1]]</pre>

```

then
    echo $(file $1)
    echo "Tamano1: $(du -b $1)"
    echo "Tamano2: $(du -h $1)"
else
    echo "no existente."
fi
fi

```

1.3. Ejercicio 3: manipulación de variables:

entorno.sh
<pre> #!/bin/bash echo "Bienvenido usuario: \$USER" echo "Usted se ha conectado en el dia: \$(date +%d/%m/%Y)" echo "A la hora: \$(date +%lh:%km:%Ss)" echo "En el terminal: \$TERM del host: \$(hostname)" echo -e "Se encuentran conectados en este momento los siguientes usuarios: \n\$(who -m)" echo -e "Su path es: \n\$PATH" </pre>

crear_env.sh
<pre> #!/bin/bash #export nuevoEntorno=\$1 echo "nuevoEntorno=\"\$1\"">>/etc/environment echo "Variable 'nuevoEntorno' creada" echo "Valor de la variable: \$nuevoEntorno" </pre>

1.4. Ejercicio 4: manipulación de variables:

bits.sh
<pre> #!/bin/bash echo "Bienvenido usuario: \$USER" echo "Usted se ha conectado en el dia: \$(date +%d/%m/%Y)" echo "A la hora: \$(date +%lh:%km:%Ss)" echo "En el terminal: \$TERM del host: \$(hostname)" echo -e "Se encuentran conectados en este momento los siguientes usuarios: \n\$(who -m)" </pre>

```
echo -e "Su path es: \n$PATH"
```

1.5. Ejercicio 5: gestión de procesos:

procesos.sh
<pre>#!/bin/bash cd / find . -name *. * > \$1 & ps echo "Se esta generando el listado de archivos que contienen un punto" ps</pre>

1.6. Ejercicio 6: redirecciones:

redirecciones.sh
<pre>#!/bin/bash if [[\$3 = "f"]] then if [[\$2 = "a"]] then ls \$1 sort -d > orden.txt elif [[\$2 = "z"]] then ls \$1 sort -rd > orden.txt fi else if [[\$2 = "a"]] then ls \$1 sort -d elif [[\$2 = "z"]] then ls \$1 sort -rd fi fi</pre>

2. Consideraciones

2.1. De la resolución:

Ejercicio 1:


El script funciona empleando de referencia los segundos transcurridos desde el 1/1/1970, con lo que de este modo contempla años bisiestos y meses de cantidad de días variable (28, 29, 30, 31) y aún así no presenta mucha complejidad adicional. Una consideración adicional es que, debido a la realización de divisiones enteras, puede perderse un poco de precisión en los resultados. De esta suerte, los resultados difieren respecto de, por ejemplo, el caso de prueba suministrado en el documento del trabajo práctico.

Un ejemplo de uso del script es

```
Ingrese la fecha de nacimiento (aaaa-mm-dd):
1970-1-1
Edad = 50
Dias vividos = 18536
Horas vividas = 444865
Minutos vividos = 26691946
segundos vividos = 1601516810
```

que comparado con el resultado recuperado de *Wolfram Alpha* (www.wolframalpha.com/)

time from 1/1/1970 to 1/10/2020 14:33

 Extended Keyboard  Upload

Assuming day/month/year | Use month/day/year instead

Input interpretation:

Thursday, January 1, 1970 to 2:33 pm UTC-03 | Thursday, October 1, 2020

Result:

50 years 9 months 14 hours 33 minutes

Time span:

50 years 9 months 0.6063 days

2648 weeks 0.6063 days

18537 days

50.75 years

444879 hours

2.669×10^7 minutes

se observa una precisión bastante razonable.

Por las pruebas que se hicieron, el script funciona correctamente hasta una fecha de nacimiento mínima del 14/12/1901.

Ejercicio 2:

_____ El comando *du*, según *man du*, informa el tamaño en disco de cada archivo, lo cual podría diferir del tamaño “real” (en bytes) de cada uno (<https://superuser.com/questions/66825/what-is-the-difference-between-size-and-size-on-disk>)

. Se hace esta aclaración puesto que no se especifica de qué tamaño se está hablando en la consigna del ejercicio. Por otra parte, “tamano1” especifica el tamaño en disco en bytes, mientras que “tamano2” lo hace con la unidad más “conveniente” (kilobytes, megabytes, etc.).

Ejercicio 3:

En el proceso de investigación se encontró que usar la palabra clave *export* es la manera más sencilla de crear una variable de entorno, mas no parece funcionar correctamente. Es por esto que se optó editar el archivo *environment* en el directorio */etc*, lo cual arriba al resultado esperado. Nota: luego de ejecutado el script, si se requiere acceder al valor a la nueva variable de entorno el usuario debe loguearse nuevamente para que los cambios surtan efecto.

Ejercicio 4: -

Ejercicio 5:

_____ En un principio se intentó usar el comando *ls -R / *.**, pero no se obtenía un buen resultado: la expresión regular no funcionaba cuando se la combinaba con el listado recursivo. Luego de investigar se encontró el comando *find*, para el cual luego de revisar *man find*, se halló que había que usarlo como *find -name \[patrón]*. La salida de este comando se escribe en el archivo con el operador *>*.

Ejercicio 6:

En un principio se recurrió solamente al comando *ls*, pero:

- En algunos casos el ordenamiento se hacía de manera separada. Por ejemplo, si se hace `./redirecciones.sh /bin/` se presentan en el tope dos archivos, `nc.traditional` y `sh.distrib`, y luego el resto: `znew`, `zmore`, ... , `bunzip2`, `bash`.
- En otros casos los archivos no se ordenaban correctamente. Por ejemplo, si se hace `./redirecciones.sh /lib` se obtiene: `cpp`, ..., `udev`, `xtables`, `libacl.so.1.1.0`, ..., `libc.so.5.7`, `klibc-DyrU_JVs9IOlno7lTBp6hHR5H7w.so`, `ld-2.11.2.so`, ... , `libutil-2.11.2.so`.

Es por esto que se empleó un *pipe* junto al comando `sort`, obteniendo los resultados correctos.

2.2. Fuentes consultadas:

Misceláneo:

- <https://tecadmin.net/tutorial/bash-scripting/>

Ejercicio 1:

- <https://stackoverflow.com/questions/1092631/get-current-time-in-seconds-since-the-epoch-on-linux-bash>
- <https://stackoverflow.com/questions/12722095/how-do-i-use-floating-point-division-in-bash>

Ejercicio 2:

- <https://stackoverflow.com/questions/4277665/how-do-i-compare-two-string-variables-in-an-if-statement-in-bash>
- <https://stackoverflow.com/questions/59838/how-can-i-check-if-a-directory-exists-in-a-bash-shell-script>
- <https://stackoverflow.com/questions/40082346/how-to-check-if-a-file-exists-in-a-shell-script>
- <https://stackoverflow.com/questions/16661982/check-folder-size-in-bash>

Ejercicio 3:

- <https://stackoverflow.com/questions/19306771/how-can-i-get-the-current-users-username-in-bash>
- <https://askubuntu.com/questions/58814/how-do-i-add-environment-variables>

- <https://stackoverflow.com/questions/18929149/print-double-quotes-in-shell-programming>

Ejercicio 4: -

Ejercicio 5:

- <https://stackoverflow.com/questions/6844785/how-to-use-regex-with-find-command>
- <https://stackoverflow.com/questions/186015/whats-the-best-way-to-find-a-string-regex-match-in-files-recursively-unix>
- <https://superuser.com/questions/246061/recursive-ls-with-conditions>

Ejercicio 6: -

- <https://stackoverflow.com/questions/878249/unixs-ls-sort-by-name>