Universidad de Sonora

División de Ciencias Exactas y Naturales

Departamento de Física

Física Computacional I

Actividad 4 ..... **Intro a la programación de Shell scripts**
Estudiante ..... Jessica Isamar Uriarte García
Docente ..... Carlos Lizárraga Celaya
Fecha ..... 5 de marzo de 2018

Resumen

Los sistemas operativos UNIX (como Linux y macOS), se apoyan con un intérprete de comandos llamado Shell, que juega el papel de intermediario entre el usuario y el sistema operativo.Hay toda una familia de intérpretes de comandos para los sistemas operativos. Entre ellos están: C Shell (/bin/csh), Bourne Shell (/bin/sh), Korn Shell (/bin/ksh/), Bourne Again Shell (/bin/bash), y otros. En el caso de la mayoría de las variantes de Linux y macOS, por default viene el intérprete /bin/bash. En esta actividad, exploraremos las distintas formas de interactuar y hacer programas (scripts) con el Bourne Shell (/bin/sh) y el Bourne Again Shell /bin/bash.

Índice

1. Introducción

El código de un Shell script es diseñada para correr en una terminal shell de Unix, que interpreta una linea de comandos. Éste tipo de programa no es muy popular entre administradores de sistemas, sobre todo cuando es comparado con el tiempo de ejecución del programa imperativo C. Las operaciones del shell script varían, unos manipulan o bajan archivos online, ejecutan programas, etc.

2. *Shell scripting*

El primer shell script que se exploró en esta actividad fue `script1.sh`

```
#!/bin/bash

# Archivo "script1.sh"
# Script para bajar automáticamente los datos de sondeos atmosféricos de la
# Universidad de Wyoming (http://weather.uwyo.edu/upperair/sounding.html)
# para un rango de tiempo. Sustituir el valor de la estación de interés:
# Estacion de Tucson, Arizona.

STATION=72274

# Despues de editar este archivo, ejecuta el comando: chmod 755 script1.sh
# Para ejecutar el script: ./script1.sh

# Definimos el separador de valores de las variables, en este caso es ":"
IFS=":"

# Por ejemplo nos interesan bajar los datos del año 2017
LISTYs="2017"

# Lista de meses por días
LISTM31="01:03:05:07:08:10:12"
LISTM30="04:06:09:11"
LISTM28="02"

for j in $LISTYs ; do

# Meses 31 dias
for i in $LISTM31 ; do
/usr/local/bin/wget "http://weather.uwyo.edu/cgi-bin/sounding?region=naconf&TYPE=TEXT
# Reposa 5 segundos
        /bin/sleep 5
done
# Meses 30 dias
for i in $LISTM30 ; do
/usr/local/bin/wget "http://weather.uwyo.edu/cgi-bin/sounding?region=naconf&TYPE=TEXT
# Reposa 5 segundos
        /bin/sleep 5
done
# Feb. 28 dias
for i in $LISTM28 ; do
/usr/local/bin/wget "http://weather.uwyo.edu/cgi-bin/sounding?region=naconf&TYPE=TEXT
# Reposa 5 segundos
        /bin/sleep 5
done
done
```

La primera linea le pide a Unix que ejecute el archivo del shell Bourne. Los siguientes cinco renglones que empiezan con un # son comentarios que pueden ser vistos en el editor de textos emacs pero no afectan la ejecución del programa. Crea la variable `STATION=72274`, la estación de Tucson, y los comentarios debajo de la estación nos recuerda ejecutar el comando `chmod 755 script1.sh`

y después el script `.\script1.sh`. Empieza a bajar los datos del año 2017 cada 12 horas y los ordena por mes, tomando en cuenta que Febrero solo tuvo 28 días. Wget permite la descarga de estos datos desde el servidor web (Página de la Universidad de Wyoming). Los función de los cíclos `for` era guardar cierta cantidad de columnas en un renglon de la listas de muestras. Se crean 12 archivos, uno por cada mes donde está la información tomada dos veces al día.

3. Comandos

chmod

Con `chmod 755` cambié los permisos de acceso del directorio `script1.sh` para poderlo modificar.

less

Vemos la información del archivo y nos permite explorarlo.

cat

El comando `cat` nos muestra el contenido del archivo y lo utilizamos para con*cat*enar los 12 archivos y guardarlos en un archivo txt.

grep

`grep` ermite realizar filtraciones y en este ejemplo solo seleccionamos y guardamos los renglones importantes. Toda esta información se guardó en un archivo separado por comas llamado df2017.csv.

4. Conclusión

Se pidió crear un script que automatizara la concatetanción y filtración de los datos. La única modificación es el nombre del csv para que no escribiera sobre el primer archivo csv creado.

```
#!/bin/bash

##########################################
# ------------------------------------ #
#           Intro a Shell scripting       #
# ------------------------------------ #
#
# __________Jessica Isamar Uriarte Garcia #
# ____________________21 de Febrero 2018 #
##########################################

# Archivo "filtro.sch"
# Script para automatizar las acciones de los comandos cat, grep/egrep.
# Guardar los resultados en un archivo df2017_2.csv

# Estando ya en la carpeta donde se encuentran los sondeos:

# Concatenamos la colección desondeos y lo almacenamos en un solo archivo sondeos.txt
cat sounding* > sondeos.txt

# Filtramos los renglones que nos interesa conservar del archivo sondeos.txt
egrep -v 'PRES|hPa' sondeos.txt | egrep '72274|Showalter|LIFT|SWEAT|K|Totals|CAPE|CIN
```

5. Bibliografía

1.
2.

6. Apéndice

- ¿Qué fue lo que más te llamó la atención en esta actividad?

  Crear mi propio script.

- ¿Qué consideras que aprendiste?

  Crear mi un script.

- ¿Cuáles fueron las cosas que más se te dificultaron?

  Me tardé en caer en cuenta que lo que escriba en el script lo lee y ejecuta la terminal Unix y no un lenguaje de programación. Me tardé en seguir la última instrucción.

- ¿Cómo se podría mejorar en esta actividad?

  Me pareció bien.

- ¿En general, cómo te sentiste al realizar en esta actividad?

  Siento que con un script puedo ahorrar tiempo y ayuda a realizar cierta orden de comandos que guarden, filtren o modifiquen lo que yo ocupe de una base de datos 'cruda'.