1)El shell scripting es el proceso de escribir scripts para el interprete de comandos del sistema operativo, llamado shell. Un script es un archivo de texto que contiene una serie de comandos que el sistema puede ejecutar secuencialmente.  
Las tareas de los scripts están orientadas a automatizar tareas repetitivas o administrativas.  
Los scripts no deben compilarse, ya que son interpretados, no compilados. Se ejecutan línea por línea por el interprete sin necesidad de generar un archivo binario previo

2)  
Echo: imprime texto  
ej  
 echo “Hola Mundo”  
Read: lee una línea desde entrada estándar en la variable var  
 read var  
a) Los comentarios se indican con #

b) Para declarar variables se utiliza un nombre y la asignación.  
-Bash soporta strings y arrays  
-Son case sensitive  
-Puede contener mayúsculas pero no puedo empezar con un numero

Ej:  
NOMBRE=“pepe” //sin espacios alrededor del =  
  
Para acceder usamos el signo $  
echo $NOMBRE imprime pepe en la salida estándar de video

3) a) Para asignar al archivo creado los permisos necesarios para poder ejecutarse se escribe

chmod +x mostrar.sh

b) ./mostrar ejecuta el script

c) resultado

d) La sustitución de comandos con permite utilizar la salida de un comando como si fuese una cadena de texto normal  
-Permite guardarlo en variables o utilizarlos directamente  
Existen dos maneras:

- con las ‘whoami’ dadas vueltas  
-con $(whoami)

e) script

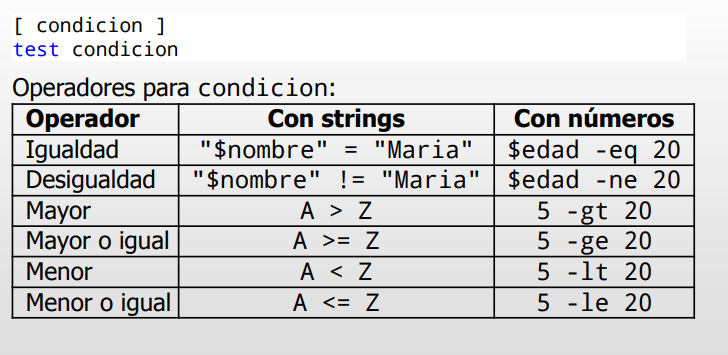
4) Parametrizacion:  
Para acceder a los parámetros enviados al script al momento de su invocación se acceden con $ desde $1. En $0 contiene la invocación al script

-$# contiene la cantidad de argumentos  
-$\* contiene la lista de todos los argumentos  
-$? contiene en todo momento el valor de retorno del ultimo comando  
-$HOME contiene el directorio personal del usuario actual

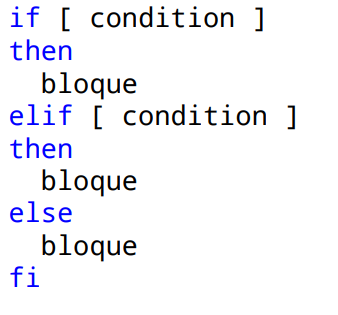
5) EXIT  
El coman do exit se utiliza para terminar un script. Puede devolver un valor entre 0 y 255.  
-El valor 0 indica que se ejecuto de forma exitosa  
-Un valor distinto indica un error o lo que querramos  
-Con $? consultamos el exist status

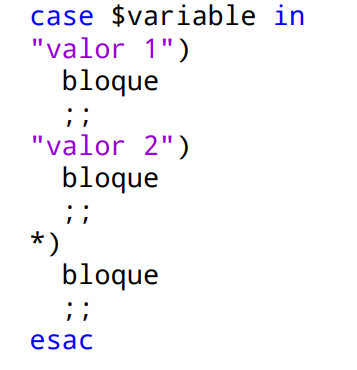
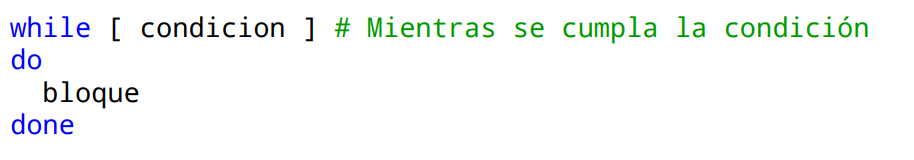
6) El comando expr en bash permite hacer operaciones aritméticas /lógicas y de strings  
expr arg1 opearador arg2 y devuelve el resultado como texto

o con variables  
expr $a + $b

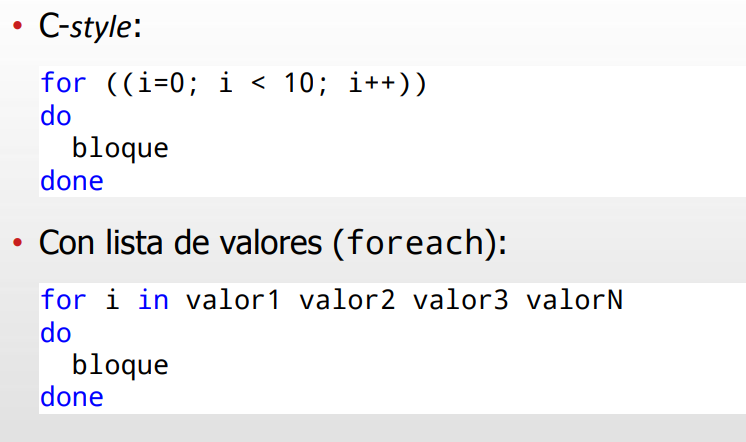
si se usa dentro de $() lo uusamos como valor  
  
7)  


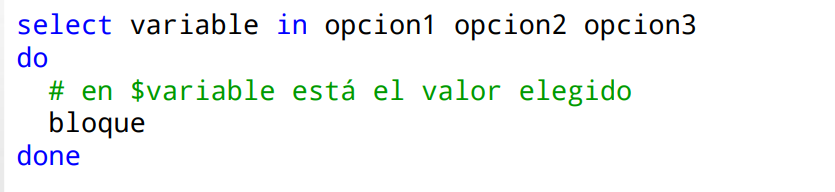
Con archivos podría ser  
-e existe archivo o directorio  
-f es un archivo  
-d si es un directorio  
-r / w / x si tiene permisos

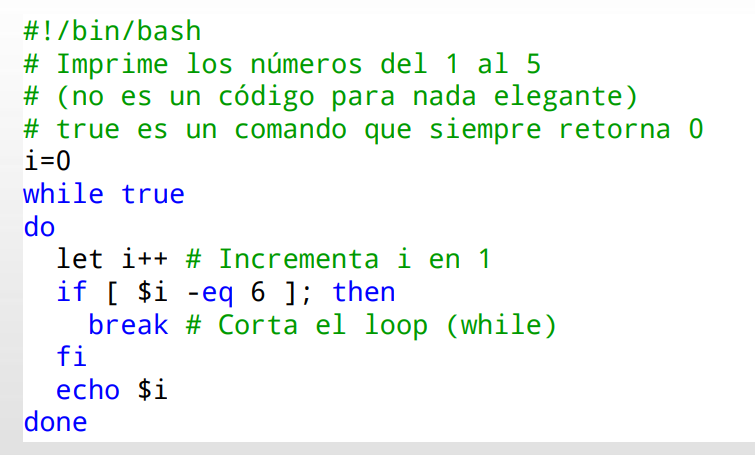
8)  
if  


case  
  
while  


for



select  


9)  
Break [n]: Corta la ejecución de n niveles de loops (el n es opcional)  
Continue [n]: salta a las siguientes n iteraciones del enesimo loop.  
Ej:  


10) Existen strings y arreglos  
No es fuertemente tipado ya que es débilmente tipado y dinámicamente tipado. Se pueden mezclar tipos sin que de error, el tipo de la variable se interpreta como la usemos.

Arrays:  
Creacion:  
 arreglo=() // se crea vacio  
 arreglob=(1 2 3 4 5 21) // arreglo inicializado  
Asignacion:  
 arreglob[2]= spam

Acceso  
 echo ${arreglob[2]} // imprimiría spam // si o si las llaves

Acceso a todos los valores

echo ${arreglo[\*]} // o @

Tamaño del arreglo  
${#arreglo[\*]} // o @

Borrado de un elemento (reduce el tamaño del arreglo pero no elimina la posición, la deja vacia (BORRADO LOGICO)

unset arreglo[2]

arreglo2=”$arreglo{@}”

11) Las funciones permiten modularizar el comportamiento de los scripts

Definicion  
function nombre { bloque}

nombre() { bloque}

Con la sentencia return se retorna un valor entre 0 y 255

El valor de retorno se evalua mediante $?

Se le pasan argumentos como (nombre hola chau)

Reciben argumentos en las variables $1 (hola) , $2 (chau)