Entrecomillado y acentos

```
$ var=”algún valor”
$ echo ”Variable var: $var” # Dobles
$ echo ’Variable var: $var’ # Sencillas
$ echo `ls –a` # Acento grave
```

Redirección

```
$ cat archivo > salida
$ cat archivo >> salida
$ rm no-archivo 2> salida
$ cat < entrada
$ cat archivo > /dev/null
$ ls –al | wc –l
```

Interacción con el usuario

```
echo –n ”Prompt: ”
read
echo ”Has tecleado $REPLY”

echo –n ”Prompt: ”
read respuesta
echo “Has tecleado $respuesta”

PS3=”Elige un ave: ”
select ave in ”gorrión” ”urraca” ”mirlo”
do
  if [ -n ”$ave” ]
    echo ”Has elegido $ave”
    break
  fi
  echo ”Elección incorrecta”
done
```

Definición de funciones

```
function nombre [()] # o nombre ()
{
  local var=”algún valor”
  #ejecución de comandos
  return [entero] # opcional
}
```

Los argumentos de la función se acceden a través de las variables posicionales

Uso de funciones

```
nombre-función arg1 arg2 arg3 … argN
```

Las funciones deben ser definidas antes de su uso

Bibliografía y reconocimientos

Bash Reference Manual, C. Ramey & B. Fox
BASH Quick Reference Card, J. McCreesh

Copyright & Licencia

Tarjeta de programación BASH

(Septiembre 2013)

```
#!/bin/bash
$ chmod ugo+x programa_shell.sh

$bash [opciones] [archivo]
Opciones
-x muestra la ejecución de [archivo]
-v muestra líneas leídas
```

Variables

```
$ var=”algún valor” # declaración
$ echo $var # acceso contenido
$ echo ${var} # otra forma de acceso
$ var= # eliminación de variable
```

Variables posicionales

$0	Nombre del script
$1 - $9	Parámetros 1 al 9
${10}	Parámetros superiores al 9
$#	Número de parámetros
$@	Valor de todos los parámetros
$?	Valor de retorno

Variables preestablecidas

$HOME	Directorio de trabajo inicial
$PWD	Directorio actual
$REPLY	Variable defectiva para read y select
$PATH	Directorios de búsqueda de programas

Operadores aritméticos

```
$ var=$(( 2 + 3 ))
$ expr 2 + 3  # 5 (suma)
$ expr 3 – 2  # 1 (resta)
$ expr 2 \* 3 # 6 (multiplicación)
$ expr 10 / 3 # 3 (cociente)
$ expr 10 % 3 # 1 (resto)
```

Operadores de cadena

Expresión	Significado
${#str}	Longitud de str
${str:pos}	Subcadena de str desde pos
${str:pos:len}	Subcadena de len caracteres desde pos
${str/sub/rep}	Sustituye la primera coincidencia de sub por rep
${str//sub/rep}	Sustituye todos los sub por rep
${str/#sub/rep}	Si sub coincide con el inicio de str, sustituye sub por rep
${str/%sub/rep}	Si sub coincide con el final de str, sustituye sub por rep

La posición en las cadenas empieza en 0

Operadores relacionales

Num	Cadena	Significado
-eq	=	Igual a
	==	Igual a
-ne	!=	No igual a
-lt	\<	Menor que
-le		Menor o igual que
-gt	\>	Mayor que
-ge		Mayor o igual que
	-z	Es vacio
	-n	No es vacio

Operadores con archivos

	Verdad si nodo existe y …
-e nodo	
-f nodo	Es un archivo regular
-r nodo	Tiene permisos de lectura
-w nodo	Tiene permisos de escritura
-x nodo	Tiene permisos de ejecución
-d nodo	Es un directorio
-s nodo	Tiene un tamaño mayor que 0

Estructuras de control

```
if [ condición ]
then
  # Se ejecuta si condición es verdad
elif [ condición1 ]
then
  # Se ejecuta si condición1 es verdad
elif [ condición2 ]
then
  # Se ejecuta si condición2 es verdad
else
  # Si ninguna condición es verdad
fi
```

```
case expresión in
  patrón1 ejecución de comandos1 ;
  patrón2 ejecución de comandos2 ;
esac

while [ verdad ]
do
  # ejecución de comandos
done

until [ falso ]
  # ejecución de comandos
Done

for x in 1 2 3 # o for x in {1..3}
do
  echo ”El valor de x es $x”;
done

LIMITE=5
for ((x=1; x <= $LIMITE; x++))
do
  echo –n ”$x ”;
done

for archivo in *
do
  echo ”$archivo”;
done

Para recuperar los nodos ocultos:
shopt –s dotglob

break # sale del bucle
continue # siguiente iteracción
```