Parámetros

$0 => nombre del script
$1, $2, $3,… => $1 primer parámetro pasado después del nombre del script,
$2 segundo parámetro, $3 tercer parámetro,…
$ => lista con todos los parámetros enviados*
$# => número de parámetros enviados.

Estructura if

```
if [ condición ]
then
        #Instrucciones a ejecutar si se cumple la condición
fi

if [ condición ]
then
        #Instrucciones a ejecutar si se cumple la condición
else
        #Instrucciones a ejecutar si no se cumple la condición
fi
```

Estructura while

```
while [ condición ]
do
        #Instrucciones a ejecutar mientras se cumpla la condición
done
```

Condiciones para el if y while

Comparadores cadenas	***Otros***
== igual	*-d comprueba si es un directorio*
!= distinto	*-f comprueba si es un fichero normal*
Comparadores numéricos	*-e comprueba si es un fichero*
-eq igual	*-s comprueba si el fichero no está vacío*
-ne distinto	*-z Comprueba si una cadena de texto está vacía*
-gt mayor que	*-n comprueba si una cadena de texto no está vacía*
-ge mayor o igual	
-lt menor que	
-le menor o igual	

Estructuras for

```
for variable in  lista
do
        #instrucciones a ejecutar
done

for  (( inst1; condición; inst2 ))
do
        #instrucciones a ejecutar
done
```

break => permite salir del bucle
continue => permite pasar a la siguiente iteración del bucle
exit => permite terminar la ejecución de un programa.

Estructura case

```
case $valor in
        valor1)
                #órdenes a ejecutar si el valor coincide con valor1;;
        valor2)
                #órdenes a ejecutar si el valor coincide con varlo2;;
        *)
                #órdenes a ejecutar si el valor no coincide con ninguno de los valores
                anteriores;;
esac
```

Operaciones aritméticas

Ejemplo con let:
```
        Let "suma=$valor1+5"
```

Ejemplo con expr:
```
        suma=`expr $valor1 + 5`
```

Ejemplo con $[]:
```
        suma =$[$valor1+5]
```

+	*Suma*
-	*Resta*
/	*División*
***	*Multiplicación*
%	*Resto división entera*

Operaciones con cadenas

long = `expr length "Hola"` => obtiene la longitud de la cadena Hola
cad=`expr substr "Hola" 1 2` => obtiene una subcadena del texto Hola de dos caracteres a partir del primer carácter.