Caracteres Comodines

Los comodines o metacaracteres (file globbing) nos van a servir **para tomar acción sobre múltiples archivos/directorios** que contengan cierta cadena de caracteres en su nombre. Es necesario tener en cuenta que los comodines son interpretados por la shell y no por los comandos. Cuando un comando es puesto con comodines, la shell los interpreta a estos y a otros tipos de expansiones, para luego ejecutar el comando. Este proceso es invisible para el usuario.

Para referirnos a uno o más archivos lo podemos hacer de una manera más acotada. La siguiente tabla nos muestra un resumen:

Comodín	Descripción	Ejemplo
*	Coincide con cualquier cadena de texto, incluso con la cadena vacía. De manera predeterminada se excluyen los nombres de archivos que comienzan con "."	`rm *` (borrará todos los archivos en el directorio en el que estemos parados) rm ho* (borrará todo lo que comience con *ho*) `rm *ho` la (borrará todos los archivos que terminen con la cadena de texto *ho*)
?	Coincide con un único carácter	`ls -l /bin/z???` va a coincidir con archivos que comienzan con "z" seguidos exactamente 3 caracteres
[lista o rango]	Coincide con cualquiera de los caracteres encerrados entre corchetes. Los rangos pueden ser alfabéticos o numéricos.	`ls -l /bin/[a-c][ar]??` va a coincidir con los archivos cuyo nombre empieza con "a", "b", o "c", el segundo carácter puede ser una "a" o una "r". Los dos últimos caracteres pueden ser cualquiera.

[!lista o rango]	Coincide con cualquiera de los caracteres no encerrados entre corchetes	**`ls -l /bin/[!a-c][ar]??`** va a coincidir con los archivos cuyo nombre **no** empieza con “a”, “b”, o “c”, el segundo carácter puede ser una “a” o una “r” y con dos caracteres finales cualquiera.
{ cadena1, cadena2... }	Se usa para generar cadenas de texto arbitrarias	**`mkdir dir{1,2,3,4}`** generará los directorios dir1, dir2, dir3 y dir4.

Más ejemplos

```
# ls -l hola*
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola1
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola2
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola4
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola5

# ls -l ?ola1
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola1
```

Busca que un caracter concuerde con alguno de los que está dentro de la lista, siempre y cuando la misma sea equivalente a la posición de donde se lo defina.

```
# ls -l [haxt4]ola[12]
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola1
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola2

# ls -la hola[2-4]
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola2
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola4
# ls -la ho[a-z]a[2-4]
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola2
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola4

[!a-z] El mismo ejemplo que el anterior pero ahora negando todo.
# ls -l hola[!a-z]
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola1
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola2
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola4
-rw-r--r-- 1 root root 0 Jun 12 05:08 hola5

Un ejemplo más:
```

```
# ls -l dir123
1aba9 aei4 3bcf5 3bdf5
# ls ?[abc][!abc][a-z][1-9]
3bdf5
```