Bash Expansion

Pattern Matching (a/k/a Wildcards or File Globbing)

`*`	Matches any string including no string
`?`	Matches any single character
`[...]`	Matches any characters in bracket, ‘-’ for range, ‘^’ for not matching

Show all the files in the current directory that we’re working with:

```
$ ls -a
.  ..  a4  a78  a8jjj  abc  akc  b6U6ii  .cucug  defz  foo.txt  goo.txt
```

Match all non-hidden files using a ‘*’:

```
$ echo *
a4  a78  a8jjj  abc  akc  b6U6ii  defz  foo.txt  goo.txt
```

Make the meta-characters no longer meta-characters:

```
$ echo \* '?' "[]"
* ? []
```

Match any file starting with the letter ‘a’:

```
$ echo a*
a4 a78 a8jjj abc akc
```

Match any file ending with the string “.txt”:

```
$ echo *.txt
foo.txt goo.txt
```

Match any 3 character file starting with ‘a’, ending with ‘c’, and any one character in the middle:

```
$ echo a?c
abc akc
```

Match any 2 character file with an ‘a’ and a digit:

```
$ echo a[0-9]
a4
```

Match any file starting with ‘a’, then a ‘b’ or a digit, and then any other characters (including none):

```
$ echo a[b0-9]*
a4 a78 a8jjj abc
```

Match any file not ending with the letters ‘c’ or ‘t’:

```
$ echo *[^ct]
a4 a78 a8jjj b6U6ii defz
```

What happens if there is no match?

```
$ echo *.pdf
*.pdf
```

Using `shopt -s nullglob` tells the shell to return an empty string when no match instead of the pattern:

```
$ shopt -s nullglob
$ echo *.pdf

```

To put the default behavior back:

```
$ shopt -u nullglob
```

Match any file starting with ‘b’, then a digit 5-7, then any character, then any upper or lowercase letter or the digit 6, then any other characters including none:

```
$ echo b[5-7]?[a-zA-Z6]*
b6U6ii
```

Brace Expansion

Brace expansion is not file globbing, so its response does not depend on any files on the file system unless you use globbing characters in the expansion.

Show numbers 3 through 14:

```
$ echo {3..14}
3 4 5 6 7 8 9 10 11 12 13 14
```

Show numbers 10 through 1 counting down:

```
$ echo {10..1}
10 9 8 7 6 5 4 3 2 1
```

Show numbers 10 through 1 counting down by 3’s:

```
$ echo {10..1..3}
10 7 4 1
```

Show numbers 33 through 47 counting up by 3’s and prefixed with “abc”:

```
$ echo abc{33..47..3}
```

```
abc33 abc36 abc39 abc42 abc45
```

Show letters ‘a’ through ‘d’ prefixed with “ABC-” and suffixed by “.pdf”:

```
$ echo ABC-{a..d}.pdf
ABC-a.pdf ABC-b.pdf ABC-c.pdf ABC-d.pdf
```

You can use file globbing characters as part of the brace expansion.

Generate the strings “f??.txt” and “g*.txt” then match:

```
$ echo {f??,g*}.txt
foo.txt goo.txt
```

Generate the strings “a*”, “b*”, “f*”, and “y*” then match:

```
$ echo {a,b,f,y}*
a78 a8jjj abc akc b6U6ii foo.txt y*
```

```
$ shopt -s nullglob;echo {a,b,f,y}*;shopt -u nullglob
a78 a8jjj abc akc b6U6ii foo.txt
```

You can use more than one brace expansion at a time. Each one is expanded in turn. Also, you can also make an empty string part of the expansion too.

Show strings “8”, and “xy” prefixed with “JKL/” and suffixed with either nothing or a “.jpg”:

```
$ echo JKL/{8,xy}{,.jpg}
JKL/8 JKL/8.jpg JKL/xy JKL/xy.jpg
```

You can also nest brace expansions:

```
$ echo a/{b,KK/{c,d,p1}}
a/b a/KK/c a/KK/d a/KK/p1
```

Extended File Globbing

`?(pattern-list)`	Matches zero or one occurrence of the given patterns
`+(pattern-list)`	Matches one or more occurrences of the given patterns
`@(pattern-list)`	Matches one of the given patterns
`!(pattern-list)`	Matches anything except one of the given patterns

A pattern list is one or more patterns separated by a `|`.

Let’s start with the same file list:

```
$ ls -a
.  ..  a4  a78  a8jjj  abc  akc  b6U6ii  .cucug  defz  foo.txt  goo.txt
```

Enable extended file globbing:

```
$ shopt -s extglob
```

Match files that of exactly three characters that start with ‘a’, have a digit or the letter ‘k’ in the middle, and end with any character:

```
$ echo @(a[0-9k]?)
a78 akc
```

You can make inclusive-or logic with two or more patterns.

Use the previous match, but then also match any files that end with “.txt”:

```
$ echo @(a[0-9k]?|*.txt)
a78 akc foo.txt goo.txt
```

Use the previous match, but then instead match any files it doesn’t match:

```
$ echo !(a[0-9k]?|*.txt)
a4 a8jjj abc b6U6ii defz
```

Using `!(...)` is handy for doing things to all files but one.

Move all files in the current directory except for `mydir` into `mydir`:

```
$ mv !(mydir) mydir
```

You can nest patterns.

Match files that start with ‘a’ followed by any character, then 0 or more ‘j’s:

```
$ echo @(a?*(j))
a4 a8jjj
```

Match files that start with ‘a’ followed by any character, then 1 or more ‘j’s:

```
$ echo @(a?+(j))
a8jjj
```

Got your Ovaltine decoder ring handy?:

```
$ echo a@(?(zzz)*([0-9])!(j|q|y)c)
abc akc
```