Chapter 2a. file globbing

The shell is also responsible for **file globbing** (or dynamic filename generation). This chapter will explain **file globbing**.

2a.1. * asterisk

The asterisk ***** is interpreted by the shell as a sign to generate filenames, matching the asterisk to any combination of characters (even none). When no path is given, the shell will use filenames in the current directory. See the man page of **glob(7)** for more information.(This is part of LPI topic 1.103.3.)

```
[clim@sop gen]$ ls

file1  file2  file3  File4  File55  FileA  fileab  Fileab  FileAB  fileabc
[clim@sop gen]$ ls File*

File4  File55  FileA  Fileab  FileAB
[clim@sop gen]$ ls file*

file1  file2  file3  fileab  fileabc
[clim@sop gen]$ ls *ile55 File55

[clim@sop gen]$ ls F*ile55
File55

[clim@sop gen]$ ls F*55
File55

[clim@sop gen]$
```

2a.2. ? question mark

Similar to the asterisk, the question mark **?** is interpreted by the shell as a sign to generate filenames, matching the question mark with exactly one character.

```
[clim@sop gen]$ ls

file1  file2  file3  File4  File55  FileA  fileab  Fileab  FileAB  fileabc
[clim@sop gen]$ ls File?

File4  FileA

[clim@sop gen]$ ls Fil?4
File4

[clim@sop gen]$ ls Fil??

File4  FileA

[clim@sop gen]$ ls File??
File55  Fileab  FileAB
[clim@sop gen]$
```

2a.3. [ ] square brackets

The square bracket **[** is interpreted by the shell as a sign to generate filenames, matching any of the characters between **[** and the first subsequent **]**. The order in this list between the brackets is not important. Each pair of brackets is replaced by exactly one character.

```
[clim@sop gen]$ ls

file1  file2  file3  File4  File55  FileA  fileab  Fileab  FileAB  fileabc
[clim@sop gen]$ ls File[5A]

FileA

[clim@sop gen]$ ls File[A5]

FileA

[clim@sop gen]$ ls File[A5][5b]

File55

[clim@sop gen]$ ls File[a5][5b]

File55  Fileab

[clim@sop gen]$ ls File[a5][5b][abcdefghijklm]

ls: File[a5][5b][abcdefghijklm]: No such file or directory
[clim@sop gen]$ ls file[a5][5b][abcdefghijklm] fileabc

[clim@sop gen]$
```

You can also exclude characters from a list between square brackets with the exclamation mark **!**. And you are allowed to make combinations of these **wild cards**.

```
[clim@sop gen]$ ls

file1  file2  file3  File4  File55  FileA  fileab  Fileab  FileAB  fileabc
[clim@sop gen]$ ls file[a5][!Z]

fileab

[clim@sop gen]$ ls file[!5]* file1
       file2  file3  fileab  fileabc
[clim@sop gen]$ ls file[!5]? fileab

[clim@sop gen]$
```

2a.4. a-z and 0-9 ranges

The bash shell will also understand ranges of characters between brackets.

```
[clim@sop gen]$ ls

file1  file3  File55  fileab  FileAB   fileabc
file2  File4  FileA   Fileab  fileab2
[clim@sop gen]$ ls file[a-z]*

fileab  fileab2  fileabc
[clim@sop gen]$ ls file[0-9]
file1  file2  file3

[clim@sop gen]$ ls file[a-z][a-z][0-9]*
fileab2
```

2a.5. $LANG and square brackets

But, don't forget the influence of the **LANG** variable. Some languages include lower case letters in an upper case range (and vice versa).

```
clim@sop:~/test$ ls [A-Z]ile?
file1  file2  file3  File4
clim@sop:~/test$ ls [a-z]ile?
file1  file2  file3  File4
clim@sop:~/test$ echo $LANG
en_US.UTF-8

clim@sop:~/test$ LANG=C
clim@sop:~/test$ echo $LANG C

clim@sop:~/test$ ls [a-z]ile?
file1  file2  file3
clim@sop:~/test$ ls [A-Z]ile?

File4
clim@sop:~/test$
```

If **$LC_ALL** is set, then this will also need to be reset to prevent file globbing.

2a.6. preventing file globbing

The screenshot below should be no surprise. The **echo *** will echo a * when in an empty directory. And it will echo the names of all files when the directory is not empty.

```
clim@sop:~$ mkdir test42
clim@sop:~$ cd test42
clim@sop:~/test42$ echo *

*

clim@sop:~/test42$ touch file42 file33
clim@sop:~/test42$ echo *
```

Globbing can be prevented using quotes or by escaping the special characters, as shown in this screenshot.

```
clim@sop:~/test42$ echo *
file33 file42
clim@sop:~/test42$ echo \*

*

clim@sop:~/test42$ echo '*'

*

clim@sop:~/test42$ echo "*"

*
```

2a.7. practice: shell globbing

2a.1. Create a test directory and enter it.

2a.2. Create the following files :

```
file1
file10
file11
file2
File2
File3
file33
fileAB
filea
fileA
fileAAA
file(
file 2
```

(the last one has 6 characters including a space)

2a.3. List (with ls) all files starting with file

2a.4. List (with ls) all files starting with File

2a.5. List (with ls) all files starting with file and ending in a number.

2a.6. List (with ls) all files starting with file and ending with a letter

2a.7. List (with ls) all files starting with File and having a digit as fifth character.

2a.8. List (with ls) all files starting with File and having a digit as fifth character and nothing else.

2a.9. List (with ls) all files starting with a letter and ending in a number.

2a.10. List (with ls) all files that have exactly five characters.

2a.11. List (with ls) all files that start with f or F and end with 3 or A.

2a.12. List (with ls) all files that start with f have i or R as second character and end in a number.

2a.13. List all files that do not start with the letter F.

2a.14. Copy the value of $LANG to $MyLANG.

2a.15. Show the influence of $LANG in listing A-Z or a-z ranges.

2a.16. You receive information that one of your servers was cracked, the cracker probably replaced the **ls** command. You know that the **echo** command is safe to use. Can **echo** replace **ls** ? How can you list the files in the current directory with **echo** ?

2a.17. Is there another command besides cd to change directories ?