Notes 7

How to use Wildcards:

Wildcards are a tool to allow the matching of files, usually apply on dense data storages.

- Asterisk(*): I tis the most common wildcard used to find files under certain characteristics

Pattern	Matches	Does NOT match
`*.txt`	`notes.txt`, `a.txt`, `longfilename.txt`	`notes.tx`, `txt`
`file*`	`file`, `file1`, `fileABC`, `file.txt`	`afile`, `myfile`
`*log*`	`syslog`, `mylogfile`, `log123.txt`	`logger`, `log` (if no matching file exists)

```
Command line:

ls *.txt
ls file*
ls *config*
```

- Question mark (?): it is a single character wildcard. It matches exactly one character, regardless of what that character is.

Pattern	Matches	Does NOT match
`file?.txt`	`file1.txt`, `fileA.txt`	`file.txt`, `file10.txt`
`??.sh`	`ab.sh`, `12.sh`	`a.sh`, `abc.sh`
`log?`	`log1`, `logA`	`log`, `log12`

```
Command line:

ls file?.txt
ls ?.txt
ls config?.json
```

- Square Brackets ([]): are used to find character ranges or sets on files names.

Pattern	Matches
`file[123].txt`	`file1.txt`, `file2.txt`, `file3.txt`
`image[abc].png`	`imagea.png`, `imageb.png`, `imagec.png`

Pattern	Matches

Pattern	Matches
`file[0-9].log`	`file0.log` … `file9.log`
`photo[A-Z].jpg`	`photoA.jpg` … `photoZ.jpg`

Pattern	Matches
`file[123].txt`	`file1.txt`, `file2.txt`, `file3.txt`
`image[abc].png`	`imagea.png`, `imageb.png`, `imagec.png`

```
Command line:

ls file[123].txt
ls photo[A-Z].jpg
ls server[!0-9].conf
```

How to use Brace Expansion to create entire directory structures:

Curly Braces({}): It allows to create multiple arguments from a single expression. This is particularly useful when you need to perform an operation on a set of files or directories with a common pattern.

```
Example:

mkdir -p project/{src/{components,utils},docs/{api,user-guide},tests/unit}

project/
├── src/
│   ├── components/
│   └── utils/
├── docs/
│   ├── api/
│   └── user-guide/
└── tests/
    └── unit/

mkdir -p build/{alpha,beta}_{1..3}/{debug,release}

build/
├── alpha_1/
│   ├── debug/
│   └── release/
├── alpha_2/
│   ...
├── beta_3/
│   └── ...
```