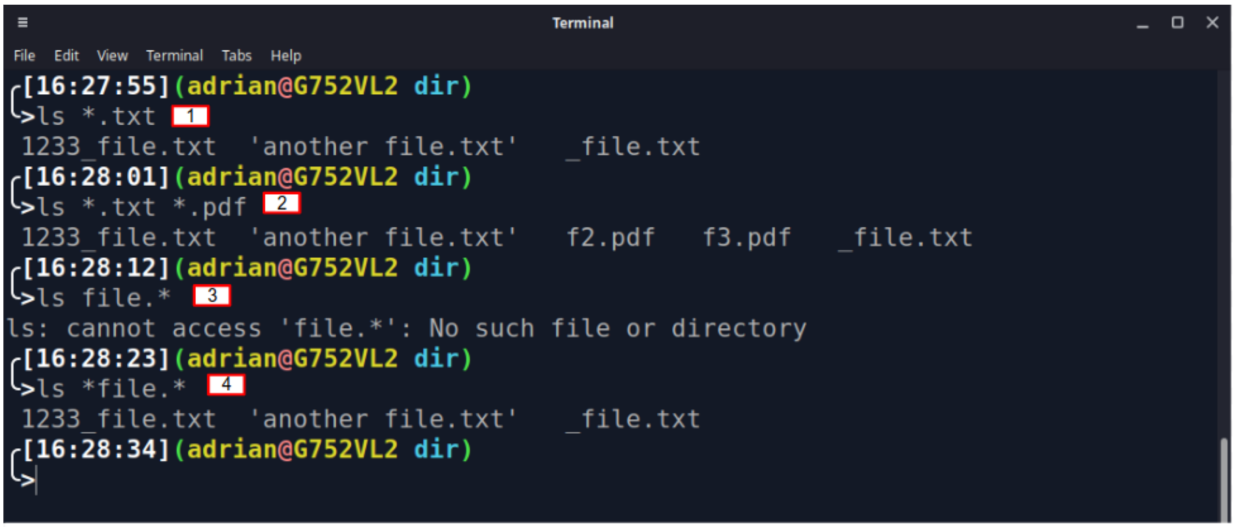
Weekly Report 6

Using Wildcards (File Globbing)

- Wildcard represents letters and characters used to specify a file name for searches.
- **File globbing** - is the process of pattern matching using wildcards.
- Wildcards are officially called metacharacter wildcards.
    - Examples of using wildcards:
        - Get a long list of all files in the current directory starting with "new"
        - Manage directories faster
        - Move or delete a group of files
        - Locate files based on a portion of their filenames
        - Create files and directories quicker
    - Wildcards are not regular expressions. A *regular expression* is a sequence of characters that define a search pattern.

The * Wildcard

- The main wildcard is a star, or asterisk (*) character.
- A star alone matches anything and nothing and matches any number of characters.
    - For instance, **ls .txt* will match all files that end in .txt regardless of the size of the file name.
    - Examples of when to use the * wildcard:
        - When you want to list all files with a particular file extension
        - When you do not remember the complete name of a file but you remember a portion of the name
        - When you want to copy, move, or remove all files that match a particular naming convention



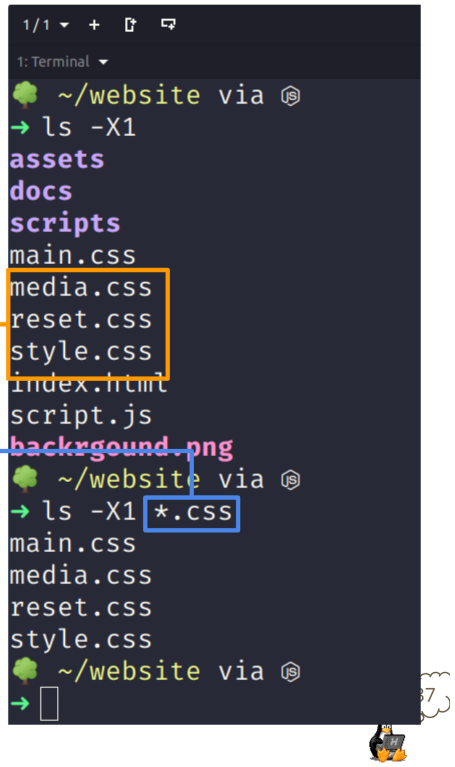
1. ls *.txt lists all the files that end in .txt
2. ls *.txt *.pdf list all the files that end in .txt and .pdf
3. ls file.* lists all the files that start with the string “file.” regardless of their file extension. In this example, there were no files in the directory that matched this criteria.
4. ls *file.* list all the files that have any letter before the string “file.” and after as well.

When working with wildcards, keep in mind two things:

- **What do the files have in common?**
  - This is the part of the file name which is the same in all the files you want to match. For example, the file extension.
- **What part of the file name is irrelevant?**
  - This is the part of the file name that it is irrelevant in the matching. In our example, this is the file name

Let's assume that you want to list only the **css** files inside this directory in a single column.
The command for achieving this would be **`ls -1 *.css`**

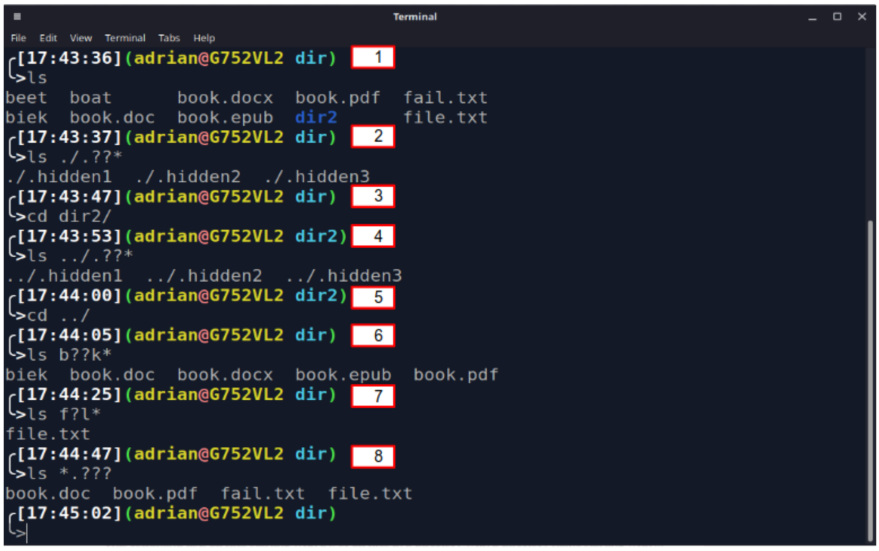
The *** wildcard replaces the name of the files** because we do not care about the file name in this instances. In the command **we keep the extension because that is what the files must have in common**.



The ? Wildcard

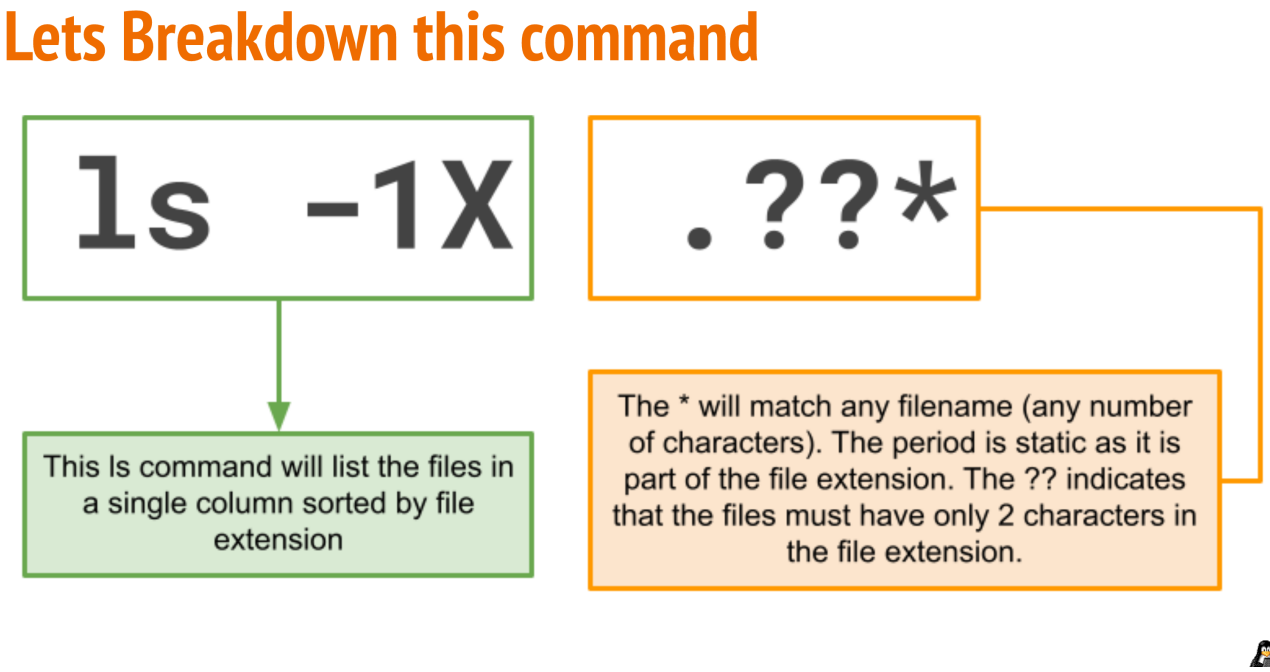
- The ? wildcard metacharacter matches **precisely one character**.
- The ? wildcard proves very useful when working with hidden files (a.k.a *dot files*)
  - Example:
    - To list all hidden files use: ls .??* which will match all files that start with a . or .. and have any character after it.

- Isn't this the same as using the * character on its own? **NO!**
  - The problem with dot files and wildcard expressions is that the **current directory** and the **parent directory** have a name.
  - The current directory is represented with a single dot (.) and the parent directory is represented with two dots (..)
    - *Remember cd ../../../ that's what I am talking about!*
  - If you use a wildcard expression such as `.*` to list all files that start with a dot. The shell will also match . and ..
  - To go around this problem you can use `./.*` to match all the files in the current directory with a file name starting with a dot.
  - You can also match all the files that start with a . in the parent directory using `../.*`

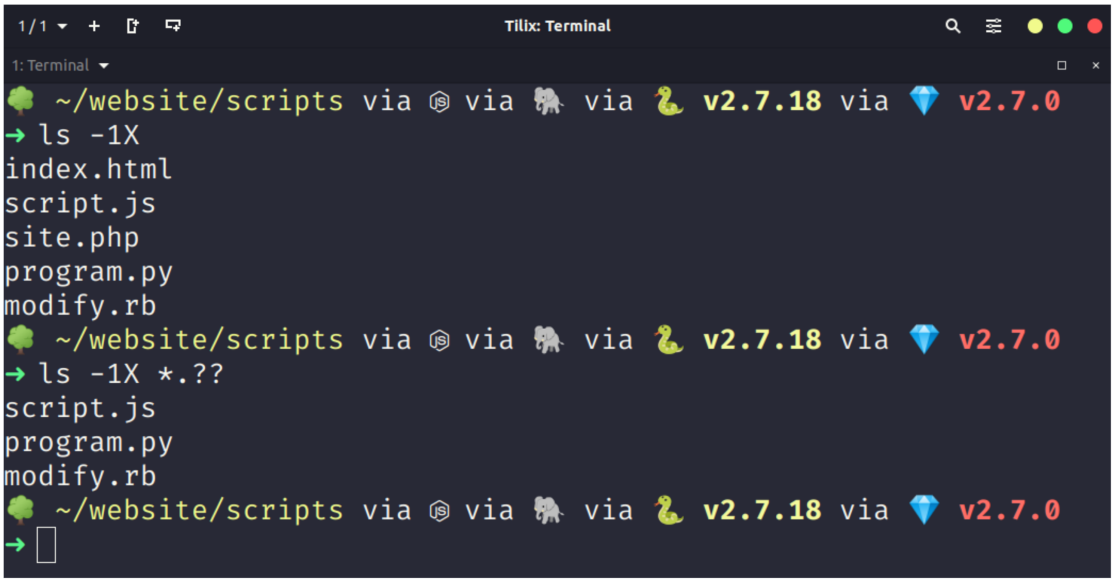


1. List all the files in the current directory (excluding hidden files)
2. List all the hidden files in the current directory
3. Changes the current working directory to dir2
4. List all the hidden files in the parent directory
5. Changes the current working directory to the previous directory (dir)
6. List all the files that have a two character between letter b and k.
7. List all the files that have a single character between letter f and l.
8. List all the files that have a 3 letter file extension.

Breakdown of ls-1X command



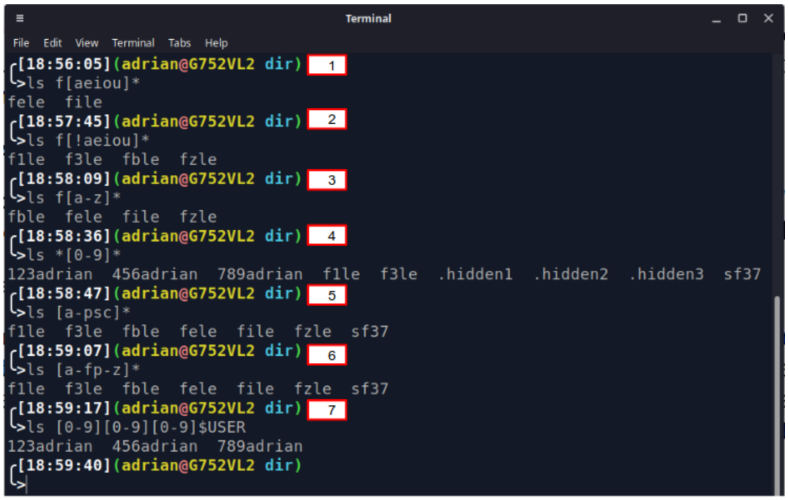
Here is the command in practice



The [] Wildcard

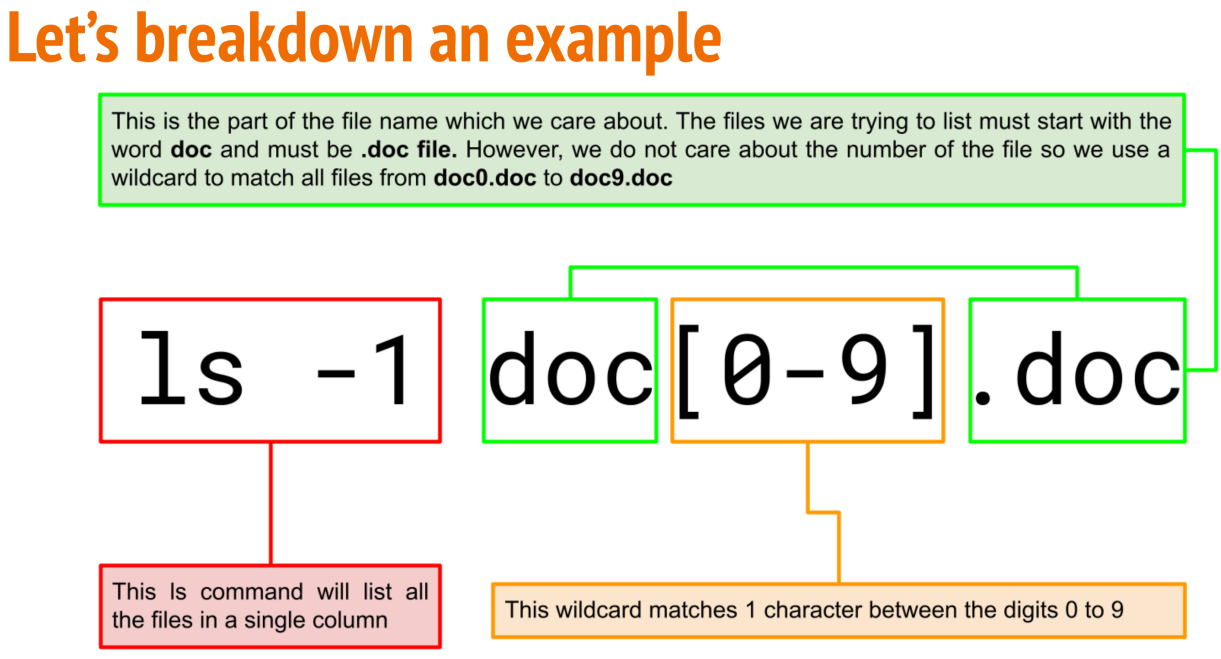
- The brackets wildcard match a single character in a range.
- The brackets wildcard use the exclamation mark to reverse the match.
    - For instance, match everything except vowels [!aeiou] or any character except numbers [!0-9].

- **Examples:**
    - To match all files that have a vowel after letter f:
        - `ls f[aeiou]*`
    - To match all files that do not have a vowel after letter f:
        - `ls f[!aeiou]*`
    - To match all files that have a range of letters after f:
        - `ls f[a-z]*`
    - To match all files whose name has at least one number:
        - `ls *[0-9]*`
    - To match all the files whose name does not have a number in their file name:
        - `ls *[!0-9].*`
    - To match all files whose name begins with a letter from a-p or start with letters s or c:
        - `ls [a-psc]*`
    - To match all files whose name begins with any of these two sets of characters: letters from a-f or p-z:
        - `ls [a-fp-z]*`
    - To match all files whose name begins with any 3 combination of numbers and the current user's username:
        - `ls [0-9][0-9][0-9]$USER`



1. To match all files that have a vowel after letter f
2. To match all files that do not have a vowel after letter f
3. To match all files that have a range of letters after f
4. To match all files whose name has at least one number
5. To match all files whose name begins with a letter from a-p or start with letters s or c
6. To match all files whose name begins with any of these two sets of characters: letters from a-f or p-z
7. To match all files whose name begins with any 3 combination of numbers and the current user's username

Breakdown of ls -1 command



Quick Reference of Using Wildcards/File Globbing

Using Wildcards / File Globbing (quick reference)

Wildcard	Description
*	Matches zero or more characters in a filename
?	Matches any one character in a filename
[acf]	Matches one of multiple characters in a filename; in this example, a, c, or f
[a-f]	Matches one of a range of characters in a filename; in this example, any character from a through f
[!a-f]	Matches filenames that don't contain a specified range of characters; in this example, filenames that don't contain a through f

Using Brace Expansion

- Brace expansion {} is **not** a wildcard but another feature of bash that allows you to generate arbitrary strings to use with commands.
    - Examples:
        - To create a whole directory structure in a single command:
            - `mkdir -p music/{jazz,rock}/{mp3files,vidoes,oggfiles}/new{1..3}`
        - To create a N number of files use:
            - `touch website{1..5}.html`
            - `touch file{A..Z}.txt`
            - `touch file{001..10}.py`
            - `touch file{{a..z},{0..10}}.js`
        - Remove multiple files in a single directory
            - `rm -r {dir1,dir2,dir3,file.txt,file.py}`