

Regular Expressions

- Regular Expression (RE):
 - defines one or more strings of characters.
 - is said to match any string it defines.
 - Example:
 - abc is a RE which matches "abc"
- character - any character except newline.
 - a-z A-Z 0-9 () = ; : ,
 - special characters - a character that represents something other than itself.
 - * . [] ? + / -
- delimiter - special character marking the start or end of a regular expression.

Who /uses/ [Rr]egular [Ee]xpre[s*]ions\?

- Regular expressions are used in many UNIX Utilities
 - sed - UNIX stream editor
 - ed - UNIX line editor
 - vi - UNIX visual editor (full screen)
 - emacs - GNU screen editor for UNIX
 - awk - pattern scanning and processing language
 - fgrep, grep, egrep - pattern matching tool

UNIX Wildcards

- A similar idea called wildcard is used with UNIX commands to specify multiple objects (files, directories) to act on.
 - ? question mark - matches a single character.
 - memo? matches memo1 memo2 memoX
 - hel? matches help helm hel3 helF
 - * asterisk ("star") - matches zero or more characters
 - a* matches all files starting with a
 - *old matches all files ending with old
 - c*d matches all files starting with c and ending with d
- Examples:
 - ls *.c
 - rm c*.*

Wildcards with Commands

```
ls /tmp
```

```
mark zippy harry carol  
cindy megan shawn allie  
lizzie daffy goofy Merlin  
happy dopey sneezY gandalf
```

```
ls /tmp/m*
```

```
mark megan
```

```
ls *y
```

```
zippy harry cindy daffy  
goofy happy dopey
```

```
ls ???????
```

```
troika Merlin sneezY
```

Using Grep

- Examples
 - `grep "abc" myfile`
 - outputs all lines in myfile which contain "abc" in them.
 - `grep -i "abc" myfile`
 - outputs all lines in myfile with "abc" in them ignoring case.
- Quotes are optional around regular expressions that do not contain spaces.
- `grep -iv abc myfile`
 - outputs all lines in myfile without "abc" in them, ignoring case

Regular Expression Special Characters

- The Period .

- matches any single character.

a.c matches abc adc aec a=c a:c

x..x matches xaax xavx a=kx

- The Asterisk *

- Zero or more occurrences of previous character.

- (This is different from wildcard meaning!)

ab*c matches ac abc abbc abbbbbc

a* matches { } a aa aaaaa

a*b*c* matches abbc aaaacc a b c

.* matches anything (like wildcard *)

Character Class Symbol [] for Regular Expressions

- square brackets indicate a set of characters

- * \$ lose their significance
- ^ has a special meaning (NOT).
- - has a special meaning (range).

[mM]ark matches mark or Mark

t[aeiou]x matches tax tex tix tox tux

abc.* matches abc followed by anything.

[a-z][a-z] matches any two character lower-case string

[a-zA-Z]* matches string made up of upper and lower case characters

[^abc].* matches any string that does not start with a or b or c

- [a-zA-Z0-9]* matches any string made up of a-z and/or A-Z and/or 0-9

Special Characters for Regular Expressions

- Caret ^
 - Outside of a character class means "beginning of line"
 - ^T.* matches all lines starting with T
 - ^[Tt].* matches lines starting with upper or lower case t.
- Dollar Sign \$
 - Outside of a character class means "end of line" or "end of file."
 - :\$ matches a colon at the end of a line.
 - ^\$ matches lines which contain no characters.

Quoting Special Characters in Regular Expressions

- Question: How can we find all occurrences of `a*b` in a file?
- Answer: You can quote any special character (except a digit or a parenthesis) by preceding it with a backslash `"\"`.
- Quoting a special character makes it represent itself.
- Example: `a*b` matches `a*b`.

- Question: How could we find all monetary values in a file of the form `#####.##?`
- Answer: `\$[0-9]*\.[0-9][0-9]`

Regular Expression Rule

- Match Longest String Possible:
- Regular expressions will always match the longest string possible starting from the beginning of the line.

- Example

This (rug) is not what it once was (a long time ago), is it?

- Th.*is matches: “This (rug is not what it once was (a long time ago), is”