

ULI101

Week 08

Week Overview

- Regular Expressions Basics
 - Literal Matching
 - Delimiters
 - Special Characters
 - . wildcard
 - * Repetition
 - Character Classes
 - ^ Beginning of Line
 - \$ End of Line
 - Grouping
- Search and Replace in vi

Regular Expression Basics

- Define set of characters using a simple expression or a pattern
- Used mainly for searching and/or replacing strings
- Used by various UNIX utilities:
 - vi
 - grep
 - sed
 - awk
- Regular expressions match input within a line
- Regular expressions are very different than shell meta-characters

Literal Matching

- Contains no special characters
- Matches only itself (literally)
- Matches entire words or parts of it

Examples:

- `/disk/` - will match: `diskette`, `disk`, `disks...`
- `/my book/` - will match `my book`, `dummy book`

Delimiters

- Delimiters mark the beginning and end of the regular expression
- Delimiters are required if there are any special characters, such as spaces or asterisks, which may be misinterpreted by the shell
- Delimiters vary by the utility and the situation
 - For grep the delimiter can be the double or single quote, depending if variable substitution is desired

Special Characters

- Special characters and expressions can be used to build regular expressions for pattern matching
 - Standard special characters include:
., *, [], ^, \$
- Depending on the utility and its version, some versions support standard regular expressions and some support extended regular expressions
- Although some of them may look like shell expansion characters they usually mean something else
- Whenever you wish to match a special character literally it must be quoted

. Wildcard

- The only wildcard in regular expressions
- Matches any single character

Example:

`/.nix/`

- will match **Unix**, **unix**...

`/leaf./`

- will match **leafs**, **leafy**...

`/a.t/`

- will match **a t**, **ant**, **act**...

* Repetition

- Represents zero or more occurrences of the part of the regular expression **directly preceding it**
 - By itself, it does not match anything – it is NOT a wildcard
- It is used in conjunction with literal matches, a period, or other special characters

Example:

- `/cart*/` - will match **car, carpool, cart, caret**
- `/of.*ice/` - will match **ofice, office, off the ice**

Character Classes

- Enclose a character class or group, similar to the shell
- Any single character within the brackets will be matched
- Hyphen can be used for defining a range of characters
- Most special characters lose their special meaning
- The caret sign at the beginning of the list means exclusion (`[^a]` = not a)
- Examples:
 - `/practi[cs]ing/` - will match `practicing` and `practising`
 - `/file[12s]/` - will match `file1`, `file2` and `files`

^ Beginning of Line

- Matches strings at the beginning of the line (anchoring it)
- Special only if the beginning of the regular expression, otherwise means a literal match

For example:

`^[0-9]` – will match any input that begins with a digit

- Inside square brackets means character exclusion

For example:

`1[^0-3]` – will match `1a`, `1.`, `145`, but not `10` and `13`

\$ End of Line

- Matches strings at the end of the line, (anchoring matches to the end of the line)

Example:

A\$ - will only match lines that end with capital letter A

Grouping

- Parentheses can be used to create bracketed regular expressions
 - The parentheses group the regular expression inside
 - The parentheses are not matched, only what is inside
- Grouping offers alternation/choice represented by the pipe (|)
For example:
“(Mr|Mrs) Smith” - will match “Mr Smith” and “Mrs Smith”
- When a grouped expression is followed by a quantifier such as the asterisk, the quantifier applies to the entire group
For example:
a(abc)*z - will match: az, aabcz, aabcabcz ...
- `grep` requires the `-E` option to enable grouping, or use `egrep`, because grouping and alternation are extended regex symbols

Search and Replace in vi

- Utilities such as vi are able to perform string substitution
- Such substitution is done using regular expressions
 - You need to be careful when using non alpha-numeric characters – quote them if necessary
- Substitution syntax:
:[address]s/original-string/replacement-string/[g]

[address]

–specifies a line range,
if not supplied only current line is used

[g]

–global substitution,
more than one per line

Example vi Substitution Ranges

- 7 – line 7
Example: `:7s/a/A/`
- 7,10 – lines 7 to 10 (inclusive)
Example: `:7,10s/a/A/`
- % – entire document
Example: `:%s/a/A/`
- 1,. – beginning of document to current line
Example: `:1,.s/a/A/`
- .,.+5 – between current line and following 5 (inclusive)
Example: `.,.+5s/a/A/`