Regular Expression

Regular expressions are a form of pattern matching syntax which many commands use. (e.g. grep, sed)

A Regular Expression contains one or more of the following:

A character set: These are the **characters** retaining their literal meaning. The simplest type of Regular Expression consists only of a character set, with no metacharacters.

An anchor: These designate (anchor) the **position** in the line of text that the RE is to match. For example, ^, and $ are anchors.

Modifiers: These expand or narrow (modify) the **range** of text the RE is to match. Modifiers include the asterisk, brackets, and the backslash.

Regular Expression

- A specific search pattern entered to find a particular target string.
- They are very flexible and not quickly learnt.
- Some basic forms are easy to learn and very useful.
- Special characters used in regular expressions include:

. matches any one character

*** matches zero or more of the last character**

.* matches any string

[ ] matches any character in the range

^ represents the start of the line

$ represents the end of the line

< matches start of a word.

• > matches end of a word.

[^] matches any character not in the range

Regular Expression

- Interpretted by the command, and not by the shell.
- Not the same as shell wildmasks, although some are similar.
- Regex metacharacters overshadow the shell’s.

RE **. => ?** in shell

RE *** => 0+** occurrence of previous char

RE **.* => *** in shell means none or any char

Regular Expression

```
grep '^[aeiou].*' /usr/share/dict/words #begin with vowels
ls |egrep '[^a-c]..[a-c]' #4 letter word begins with non a,b,c and ends with a/b/c
ls -l | grep '^-rw-' #begins with -rw-
^$ #begin end: blank line
grep '^bash' /usr/share/dict/words #begins with bash
grep ‘shell$’ /usr/share/dict/words #ends with shell
grep '\<computer' /usr/share/dict/words
        #escaping < to see the beginning of word
grep 'computer\>' /usr/share/dict/words
        #escaping > to see the end of the word
sudo ls /proc/1/fd | grep '^[[:digit:]]$'
#all files which have single digit as filename
```

Regular Expression:metachars

More readable Named Character Classes exist in dealing with more complex expressions.

- [:alnum:] - alphanumeric characters; same as [a-zA-Z0-9]
- [:alpha:] - alphabetic characters; same as [a-zA-Z]
- [:digit:] - digits; same as [0-9]
- [:upper:] - upper case characters; same as [A-Z]
- [:lower:] - lower case characters; same as [a-z]
- [:space:] - any white space character, including tabs.
- [:punct:] - Punctuation characters.

```
ls -l | grep [[:digit:]] #display filenames containing digit
ls  | grep '^[a[:digit:]b]' #all files which start with digit or ‘a’ or ‘b’
```

Regular Expression: grep

```
grep "^mo.*ing$" /usr/share/dict/words
#begins with mo followed by any number of chars and ending with ing
grep '^e.*l\+y' /usr/share/dict/words
#begins with e, contains ly,lly,lly,...
 egrep '^e.*l+y' /usr/share/dict/words
# same as above, no escape for + in extended regular expression format
grep '^[[:upper:]].*w$' /usr/share/dict/words
#begins with upper case char and ends with w
grep '^[[:upper:]a].*w$' /usr/share/dict/words
#begins with either upper case char or ‘a’ and ends with w
```

Regular Expression

- Most of the metachars must be escaped (in BRE)!
- Asterisk/Kleene star (*) - matches 0+ occurence(s) of an expression.
- Optional (\?) - matches 0 or 1 occurence of an expression
- Alternation (\|) - matches either of the expressions it sits between.
- Plus (\+) - matches 1+ occurrence(s) of an expression

d*

Saviou\?r

M[sr]\|Miss

ho\+ray

- To avoid escaping, use egrep or grep -e to use ERE instead BRE.