# CSE 391 Lecture 3

Regular expressions and egrep

slides created by Marty Stepp, modified by Jessica Miller, Ruth Anderson and Zorah Fung http://www.cs.washington.edu/391/

### Lecture summary

- regular expression syntax
- commands that use regular expressions
  - egrep (extended grep) search
- links
  - http://www.panix.com/~elflord/unix/grep.html
  - <u>http://www.robelle.com/smugbook/regexpr.html</u>
  - https://regex101.com/

### What is a regular expression?

- "[a-zA-Z\_\-]+@(([a-zA-Z\_\-])+\.)+[a-zA-Z]{2,4}"
- regular expression ("regex"): a description of a pattern of text
  - can test whether a string matches the expression's pattern
  - can use a regex to search/replace characters in a string
  - regular expressions are extremely powerful but tough to read
    - (the above regular expression matches basic email addresses)
- regular expressions occur in many places:
  - shell commands (grep)
  - many text editors allow regexes in search/replace
  - Java Scanner, String split (CSE 143 grammar solver)

### egrep and regexes

command	description
egrep	<u>extended grep;</u> uses regexes in its search patterns: equivalent to grep -E

#### egrep "[0-9]{3}-[0-9]{3}-[0-9]{4}" contact.html

• grep uses "basic" regular expressions instead of "extended"

- extended has some minor differences and additional metacharacters
- we'll just use extended syntax. See online if you're interested in the details.
- -i option before regex signifies a case-insensitive match
  - egrep -i "mart" matches "Marty S", "smartie", "WALMART", ...

### **Basic regexes**

#### "abc"

- the simplest regexes simply match a particular substring
- this is really a pattern, not a string!

• the above regular expression matches any line containing "abc"

- YES: "abc", "abcdef", "defabc", ".=.abc.=.", ...
- NO: "fedcba", "ab c", "AbC", "Bash",...

### Wildcards and anchors

- . (a dot) matches any character except n
- ".oo.y" matches "Doocy", "goofy", "LooPy", ...
- use  $\$ . to literally match a dot . character
- ^ matches the beginning of a line; \$ the end
  - "^fi\$" matches lines that consist entirely of fi
- \< demands that pattern is the beginning of a word; \> demands that pattern is the end of a word
  - "\<for\>" matches lines that contain the word "for"
  - Words are made up of letters, digits and \_ (underscore)
- *Exercise* : Find lines in ideas.txt that refer to the C language.
- Exercise : Find act/scene numbers in hamlet.txt.

# **Special characters**

#### means OR

- "abc | def | g" matches lines with "abc", "def", or "g"
- precedence of ^(Subject Date) vs. ^Subject Date:
- There's no AND symbol.
- () are for grouping and forcing precedence
- "(Homer|Marge) Simpson" matches lines containing "Homer Simpson" or "Marge Simpson"

#### \ starts an escape sequence

- many characters must be escaped to match them: /\\$.[]()^\*+?
- "\.\\n" matches lines containing ".\n"

# Quantifiers: \* + ?

- \* means 0 or more occurrences
- "ab<u>c\*</u>" matches "ab", "abc", "abcc", "abccc", ...
- "a(bc)\*" matches "a", "abc", "abcbc", "abcbcbc", ...
- "a.\*a" matches "aa", "aba", "a8qa", "a!?\_a", ...
- + means 1 or more occurrences
  - "a(bc)+" matches "abc", "abcbc", "abcbcbc", ...
  - "Goo+gle" matches "Google", "Gooogle", "Gooogle", ...
- ? means 0 or 1 occurrences
- "Martina?" matches lines with "Martin" or "Martina"
- "Dan(iel)?" matches lines with "Dan" or "Daniel"

• *Exercise* : Find all ^^ or ^\_^ type smileys in chat.txt.

# More quantifiers

{min,max} means between min and max occurrences

"a(bc){2,4}" matches "abcbc", "abcbcbc", or "abcbcbcbc"

• *min* or *max* may be omitted to specify any number

- "{2,}" means 2 or more
- "{,6}" means up to 6
- "{3}" means exactly 3

### **Character sets**

[ ] group characters into a character set; will match any single character from the set

- "[bcd]art" matches strings containing "bart", "cart", and "dart"
- equivalent to "(b|c|d)art" but shorter
- inside [ ], most modifier keys act as normal characters
  - "what[.!\*?]\*" matches "what", "what.", "what!", "what?\*\*!", ...

• *Exercise* : Match letter grades in 143.txt such as A, B+, or D-.

## **Character ranges**

- inside a character set, specify a range of characters with -
  - "[a-z]" matches any lowercase letter
  - "[a-zA-Z0-9]" matches any lower- or uppercase letter or digit
- an initial ^ inside a character set negates it
  - "[^abcd]" matches any character <u>other than</u> a, b, c, or d
- inside a character set, can sometimes be tricky to match
  - Try escaping it (use \) or place it last in the brackets
  - "[+\-]?[0-9]+" matches optional + or -, followed by eone digit

• Exercise : Match phone #s in contact.html, e.g. (206) 685-2181.

# **Back-references**

every span of text captured by () is given an internal number

- you can use \number to refer to captured text
- \1 is the first parenthetical capture
- \2 is the second parenthetical capture

- Back-references can be used in egrep pattern matching
  - Match "A" surrounded by the same character: "(.)A\1"

### **Other tools**

• find supports regexes through its -regex argument

find . -regex ".\*CSE 14[23].\*"

• Many editors understand regexes in their Find/Replace feature



# Yay Regular Expressions!

