Today’s Goals

• Regular Expressions

---

Regular Expression

• More conventionally called a “pattern”
• An expression that describes a set of strings
• Gives a concise description of the set without listing all elements
• There are usually multiple regular expressions matching the same set
• The origin of regular expressions lies in automata theory and formal language theory

---

Alternation and Grouping

• Or – |
  • gray|grey \(\rightarrow\) gray, grey
• Grouping – parentheses
  • gr(a|e)y \(\rightarrow\) gray, grey

---

Expressions

• Fundamental expression
  • Single character matches itself
• Bracket expression [ ]
  • Matches any single character in that list
  • If preceded by ^ then it matches any character not in the list.
  • \([0123456789]\) \(^[0123456789]\)
• Range expression –
  • \([0-9]\)

---

Named Classes

• Predefined bracket expressions to save typing
  • [:alnum:][:alpha:][:cntrl:]
  • [:digit:][:graph:][:lower:]
  • [:upper:][:punct:][:space:]
  • [:ctrl:][:xdigit:]
• \w == [:alnum:]\n• \W == [^[:alnum:]]

---

Quantification

• e? 0 or 1 occurrence of e
  • color? \(\rightarrow\) color, colour
• e* 0 or more occurrence of e
  • google \(\rightarrow\) ggle, ggle, google, googogle …
• e+ 1 or more occurrence of e
  • google \(\rightarrow\) ggle, google … but NOT ggle
• e(n) n occurrences of e
• e(n,) n or more occurrences of e
• e(n,m) n-m occurrences of e
Which Regex?

- Vowels
- No letters
- Either a or b, 1 or more times
  - b, abba, baabaa ….
- 5 consecutive lower-case letters
- All English terms for an ancestor
  - father, mother, grand father, grand mother, great grand father, great grand mother, great great grand father …

Others

- . matches any character
- ^ matches the start of a line
- $ matches the end of a line
- \< \> matches the beginning and the end of a word
- \ escapes any special characters, i.e. if you actually want to match ., must match \\.

Which Regex?

- 3 letter string that ends with “at”
- 3 letter string that ends with “at”, except for “bat”
- “hat” or “cat”, but only if first thing on a line
- words with no vowels
- Floating point number

Back Reference

- \n matches the expression previously matched by the n\th parenthesized subexpression
- Find all matching html title tags, h1, h2 … h6 (i.e. <h1> text </h1>)
  - <h[1-6]>.*</h[1-6]>
  - <h[1-6]>*</\1>
  - n is indexed from 1

grep, egrep and regex

- grep supports traditional Unix regex, while egrep supports full posix extended regex, and is therefore more powerful.
- grep –e is equivalent to egrep
- When giving regex at command line, must quote entire expression so that the shell will not try to parse and interpret the expression
- Use single quotes instead of double quotes

grep/egrep

- Will find all lines that “contains” the matching regex, that often defeats expressions with ^
- Want to find lines with no digits in temp.txt
  - % egrep ‘[^0-9]’ temp.txt
  - % 5 4 3
    This is many 000000000
- Use grep –v ‘[0-9]’ temp.txt
### grep/egrep Flags

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-c</code></td>
<td>print matching line count instead</td>
</tr>
<tr>
<td><code>-i</code></td>
<td>ignore cases</td>
</tr>
<tr>
<td><code>-n</code></td>
<td>prefix each output line with line number</td>
</tr>
<tr>
<td><code>-r</code></td>
<td>recursively match all files in directory</td>
</tr>
<tr>
<td><code>-v</code></td>
<td>print non-matching lines, i.e. lines that do not contain the matching pattern</td>
</tr>
</tbody>
</table>

### Summary

- Regular expressions are very powerful
- There’s a lot more they can do!