• Homework 1 Note
  – some of you have already submitted Homework 1
  – there is some question as to whether you should treat the letter ‘y’ as a consonant or vowel
  – (I just used the traditional 5 (orthographical) vowels: a, e, i, o and u
  – but either way is fine, up to you: as long as you state your assumptions
  – I’m looking for letter sequences within a word only when it comes to palindromes, e.g. levels or racecar
Regexp: Recap

• **Grouping**
  – metacharacters ( and ) delimit a group
  – inside a regexp, each group can be referenced using **backreferences** \1, \2, *and so on...*
  – outside a regexp, each group is stored in a variable $1, $2, *and so on...*

• **Example:**
  ```perl
  open (F,$ARGV[0]) or die "$ARGV[0] not found!\n";
  
  while (<F>) {
    print $1, "\n" if  
      (/(\b\w*([aeiou])\2\w*\b)/);
  }
  ```

• **Grouping:**
  – (1h(2e)ed) $1 \1
  – (1b(2o)oks)
More on Perl and regexps

- In the previous example:
  ```perl
  print $1 if (/((\b\w*([aeiou])\2\w*\b))/);
  ....
  ```
  it is assumed by default we are matching with the variable `$_`

- We can also match against a variable of our own choosing using the `=~` operator
  ```perl
  Example:
  $x = "this string";
  if ($x =~ /^this/) {
      print "ok"
  }
  ```

- Note:
  `=~` returns 1 (to be interpreted as boolean true) when there is a match, "" (false) otherwise
More on Perl and regexps

• Matching is by default case sensitive: this can be changed using the modifier \i
• /regexp/i

• **Example:**
  - /the/i
  - /[tT][hH][eE]/
  are equivalent
Perl and regexps

• Normally, Perl takes the first match only
• Multiple matches within a string can be made using the \texttt{g} modifier with a loop:
  • \texttt{/regexp/g}

\textbf{Example:}

\begin{verbatim}
$x = "the cat sat on the mat";

while ( $x =~ /the/g )
  { print "match!\n" }

prints match! twice
\end{verbatim}

The number 0, the strings \texttt{"0"} and \texttt{""}, the empty list \texttt{()}, and \texttt{undef} are all false in a boolean context. All other values are true.
Perl and regexps

• For multiple matching cases:
  `/regexp/g`
Perl must “remember” where in the string it has matched up to
• the function
  `pos string`
can be used to keep track of where it is

**Example:**
```perl
$x = "heed head book";
while ($x =~ /([aeiou])\1/g) {
  print "Match ends at position ", pos $x, "\n"
}
```

<table>
<thead>
<tr>
<th>h</th>
<th>e</th>
<th>e</th>
<th>d</th>
<th>h</th>
<th>e</th>
<th>a</th>
<th>d</th>
<th>b</th>
<th>o</th>
<th>o</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
</tbody>
</table>
Perl and regexps

• **Substitution:**
  - `s/re_1/re_2/`
  - replace 1st regexp \(re_1\) with 2nd regexp \(re_2\)
  - `s/re_1/re_2/g`
  - global \(g\) replacement version
  - *relevant if \(re_1\) occurs more than once in a line*

• **Example:**
  ```perl
  $x = "\$150 million was spent";
  $x =~ s/\d+ million/$,000,000/;
  ```

  `=~` means apply expression on the right hand side of `=~`
  to the string referenced on the left hand side

  NB. `$` in the string needs to be escaped in Perl because `$` starts a variable
Perl and regexps

- Homework file
  
  `wsj2000.txt` contains sentences with spaces separating punctuation characters

- Let’s try writing a Perl program to eliminate those extra spaces

- **Example:**
  - Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
  - Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.

- **Modified:**
  - Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
  - Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.
Perl and regexps

• Sample code:

```perl
open (F,$ARGV[0]) or die "$ARGV[0] not found!\n";

while (<F>) {
  s/ ([.,?;:])/$1/g;
  print;
}
```

• Example:

– Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
– Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.

• Modified:

– Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov. 29.
– Mr. Vinken is chairman of Elsevier N.V., the Dutch publishing group.

concepts: search and replace, grouping, global
print takes $_[ as default
Perl and regexps

- Good start, but more needed ...
  - "\s % $"
  - exercise for the reader...

- "We have no useful information on whether users are at risk," said James A. Talcott of Boston's Dana-Farber Cancer Institute.

- Exports in October stood at $5.29 billion, a mere 0.7% increase from a year earlier, while imports increased sharply to $5.39 billion, up 20% from last October.
Next time

- New topic: Regular grammars
- Make sure you’ve installed SWI-Prolog
...