Unleashing the Shell

Hands-On Unix System Administration DeCal

Lab 6 — 27 February 2012
Due 5 March at 6:10 PM

1 Regular Expressions

You'll want to have some sort of regular expression reference open for this problem. One good website is `http://www.ternent.com/tech/regexp.html`; if you're on a campus-provided network connection or have configured your browser to use the Berkeley library proxy, there are also some good O'Reilly ebooks available on regular expressions through Safari (`http://proquest.safaribooksonline.com/`).

1. Write a regular expression that matches against hyphenated Social Security numbers.
2. Try to write a regular expression that matches URIs. (Don't worry about being too precise, but if you'd like some examples, look at RFC 3986. No, you're not allowed to use the regex it defines.)
3. Write a regex that matches RFC 822 dates, such as those returned by the `date -R` command. (If you're on a Solaris or BSD machine, `s/date/gdate/g`.)

2 Shell scripting

The Fall 2008 Sysadmin DeCal lecture on shell scripting may be a useful reference for this problem. You can find it at:

`http://www.ocf.berkeley.edu/decal/2008-fall/intermediate/week06_slides.pdf`

Write a shell script, `piglatin`, that converts English into Pig Latin, one word at a time. Assume that sentences only contain letters and spaces — don't worry about numbers and other special cases. You can either use standard input (look up the shell command "read"), or command line arguments, whichever you think is easier. That is, your program should have one of the following interfaces:

- `% echo "The quick brown fox jumped over the lazy dog" | piglatin`
- `% piglatin The quick brown fox jumped over the lazy dog`

The Pig Latin algorithm: For each word w,

- if w starts with a consonant, move its first letter to its end and append "-ay."
- if w starts with a vowel, just append "-ay."

And some tips:

- If you plan using a read loop, be warned that *read* will read a whole **line** into a variable. To split it into words, you'll want to change bash's Internal Field Separator.
- You can check the first character of a word using *case*. For example (note that your script has to handle uppercase letters as well):

```
case "$word" in
  [aeiou]*) ... vowel case ... ;;
  *) ... everything else ... ;;
esac
```