Lab 9: Regular Expressions I

36-350, Statistical Computing

Friday, 11 November 2011

Agenda: Write and use basic regular expressions in R.

Instructions: Save all your answers in a single plain text file (Word files will not be graded), and upload it to Blackboard, using the page for this assignment. (There is no general digital dropbox any more.) When a question asks you to do something, give the command you use to do it. For questions which ask you to explain, write a short explanation in coherent, complete sentences. (You will be graded on your written explanation, not what you might say to the TA.)

1. Enter the following command:

```
x <- readLines('http://www.stat.cmu.edu/~cshalizi/statcomp/labs/09/dates.txt')
```

   (a) (10) Write a regular expression to match the dates in the angled brackets in the character vector `x`. They are of the following form:

```
<July 11> <June 5> <May 27> <Aug  23> <Dec  9>
```

   (b) (5) Verify that your regular expression works correctly by using the `gregexpr()` with `x`.

   (c) (25) `gregexpr()` returns a list of vectors of match positions with the attribute `match.length`. The match lengths can be extracted from the vectors using the `attr()` function. For example,

```
i <- gregexpr( ... )
attr(i[[1]] , "match.length")
```

   Use the `substring()` function together with `gregexpr()` to extract the dates. Your result should be 100 strings of the form: `month day`.

2. This is a continuation and improvement of the example from the lecture on Monday. Enter the following command to import the text of Steve Jobs' commencement address into R.

```
sj <- readLines('http://www.stat.cmu.edu/~cshalizi/statcomp/labs/09/stevejobs.txt')
```

   (a) (5) What kind of object is `sj`, what is its length, and what does each element correspond to?

   (b) (15) Assume that the sentences in the text are separated by one of the following characters, possibly with whitespace surrounding it:

```
. ! ?
```

   Split the text into sentences and store them in a character vector named `sj.sentences`. Each element of `sj.sentences` should correspond to one sentence. It is okay if you get sentences that were quotations. Hint: 1) First make a new string that is the concatenation of all of the lines in `sj`. 2) Be careful with lists!

(c) (10) Use the function `grep` to find the sentences where Steve Jobs used either the word "life" or "death." How many are there?

(d) (10) Assume that the words in the text are separated by one or more whitespace characters or the sentence separators described in the previous part. Split the text into lower-case words and store them in a character vector named `sj.words`. You can either convert to lowercase before or after splitting. You may start either from `sj` or `sj.words` — whichever is easier for you.

(e) (15) Use the function `gsub()` to remove any initial or final punctuation marks from `sj.words`, i.e. punctuation marks that appear at either the beginning or the end of the string. Hint: Use `^` and `$`.

(f) (5) Use the function `table` to count the number of occurrences of each word in `sj.words`.