Week Three - R Character Manipulation and Date Processing [02/13 - 02/19]

Week 3 Overview

This week, we'll work with character strings and dates in R.

Events and Deliverables

Thursday 02/16	Meetup, 6:30 p.m. EST
Thursday 02/16	Discussion Post due
Sunday 02/19	Week 3 Assignment due
Sunday 02/19	Discussion Response due
Sunday 09/26	Project 1 due

Week 3 Reading

Please read:

- *Automated Data Collection in R*, chapter 8.
- *Data Science for Business*, chapter 3.

Here is are some additional resources that you may find helpful.

- *R for Data Science*, chapters 11 ("Strings with stringr") and 13 ("Dates and Times with lubridate").
- "Handling and Processing Strings in R," Gaston Sanchez, http://gastonsanchez.com/Handling_and_Processing_Strings_in_R.pdf, Includes link to freely downloadable (113 pp.!) eBook with good material on base R and `stringr` package string manipulation and `grep`.

Regular Expressions

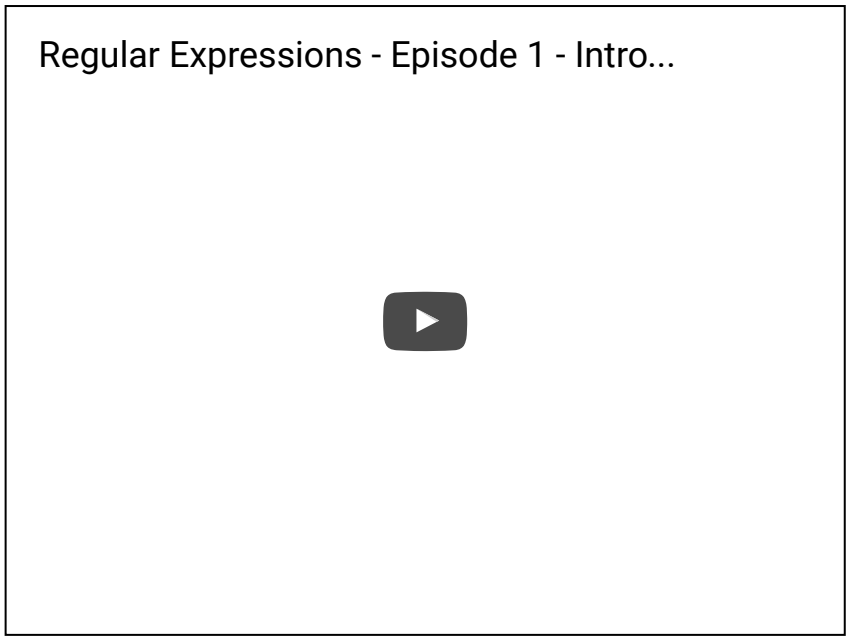
Regular expressions provide advanced text processing capabilities. Often, you can choose between using R's string manipulation functions and "regex" functions. For messy text based source data, like scraping text from HTML-based web pages, regular expressions are often the best way forward.

Regular expressions are implemented somewhat differently in different programming languages, so you'll need to carefully test any regex code that you bring from another environment into R.

You may want to consult the overview article, “Regular Expressions as used in R”, https://stat.ethz.ch/R-manual/R-devel/library/base/html/regex.html

The best introduction that I've found to Regular Expressions was put together by Software Carpentry for its (open source) Python course. Please also watch the video below:

Source: Software Carpentry, Inc. https://www.youtube.com/watch?v=c-Ov1JUMDv4



Optional. If you're interested in learning more about Regular Expressions, here are links to the other Software Carpentry videos on regular expressions, with example code in Python.

- Regular Expresssion 2: Operators, https://www.youtube.com/watch?v=G7_HnivvnyE
- Regular Expressions 3: Mechanics, https://www.youtube.com/watch?v=iixnLh55wp0
- Regular Expressions 4: Patterns, https://www.youtube.com/watch?v=FgxQyukp39A
- Regular Expressions 5: More Tools, https://www.youtube.com/watch?v=RGN5tS-2Zm0

Discussion 3: Analyzing sets of text/log files

For this week’s data science context discussion, you’re asked to identify (or imagine) a “use case” that involves “loading a set of text files, and then performing

an analysis on the loaded, combined dataset."

A short description of your use case is sufficient. Here is an example:

> "Each day, my organization's primary database creates a log file with about 50 MB of activity information. I would create an automated process to load the log files generated by my database server, and generate a daily report that tells me what are the five most frequently run queries and five slowest running queries for the previous day, and I would track how these frequent and/or slow queries change over time. This would help me to prioritize my database performance tuning efforts, such as deciding when to add an index to a specific table."

Please post by end of day on Thursday February 16th, and weigh in on at least one class mate's post by end of day on Sunday February 19th.

Week 3 Assignment

Please deliver links to an R Markdown file (in GitHub and rpubs.com) with solutions to problems 3 and 4 from chapter 8 of *Automated Data Collection in R*. Problem 9 is extra credit. You may work in a small group, but please submit separately with names of all group participants in your submission.

Here is the referenced code for the introductory example in #3:

```
raw.data <-"555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-65
```

Due end of day Sunday February 19th.

3. Copy the introductory example. The vector `name` stores the extracted names.

```
R> name
[1] "Moe Szyslak"       "Burns, C. Montgomery" "Rev. Timothy Lovejoy"
[4] "Ned Flanders"      "Simpson, Homer"       "Dr. Julius Hibbert"
```

1. Use the tools of this chapter to rearrange the vector so that all elements conform to the standard `first_name last_name`.
2. Construct a logical vector indicating whether a character has a title (i.e., `Rev.` and `Dr.`).
3. Construct a logical vector indicating whether a character has a second name.

4. Describe the types of strings that conform to the following regular expressions and construct an example that is matched by the regular expression.

1. `[0-9]+\\$`
2. `\\b[a-z]{1,4}\\b`
3. `.*?\\.txt$`
4. `\\d{2}/\\d{2}/\\d{4}`
5. `<(.+?)>.+?</\\1>`

9. The following code hides a secret message. Crack it with `R` and regular expressions. *Hint: Some of the characters are more revealing than others! The code snippet is also available in the materials at www.r-datacollection.com.*

```
clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0Tanwo
Uwisdij7Lj8kpf03AT5Idr3coc0bt7yczjatOaootj55t3Nj3ne6c4Sfek.r1w1YwwojigO
d6vrfUrbz2.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5
fy89n6Nd5t9kc4fE905gmc4Rgxo5nhDk!gr
```

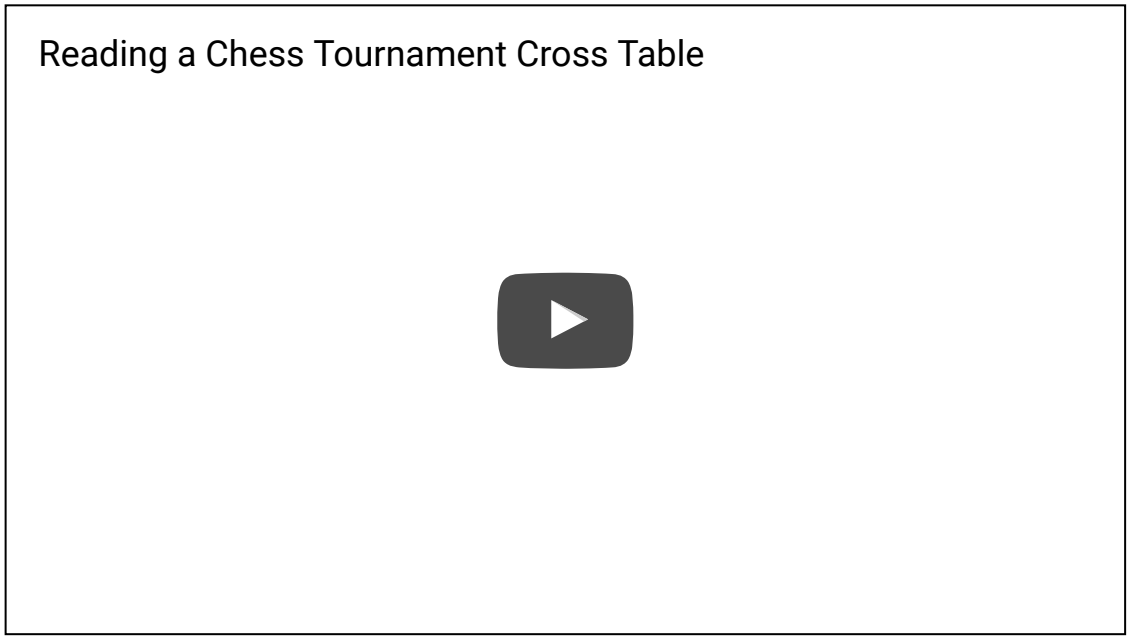
Project 1

Attached Files: Project 1.pdf (120.332 KB)
tournamentinfo.txt (17.418 KB)

This project is due on Sunday February 26th. We'll review the most interesting solutions in our meetup on Thursday March 9th. (There is no meetup on Thursday March 2nd]

As before, please deliver your code in GitHub, with submission links to your GitHub repository and your published R Markdown file. You may work in a small group on this project.

Here is a 3 minute video on reading chess tournament cross-tables that you may find helpful:



[Chess ELO ratings have been adopted for use in ranking players and teams in other sports, as well as in a few human resources applications. You can learn more about ELO rating use cases here: https://fivethirtyeight.com/tag/elo-ratings/]