DATA 607 - Week 3 Assignment

Adam Douglas

9/11/2018

Problems

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

```
# Introductory text

raw.data <-"555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-6542
Rev. Timothy Lovejoy555 8904Ned Flanders636-555-3226Simpson, Homer5553642
Dr. Julius Hibbert"

name <- unlist(str_extract_all(raw.data, "[[:alpha:]., ]{2,}"))

name
```

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

(a) Use the tools of this chapter to rearrange the vector so that all elements conform to the standard `first_name last_name`.

```
# Because we have two formats: First Last and Last, First
#   the first name is either a word terminated by a space (including
#   a single initial with a period) or
#   a word at the end of the string.

first <- str_trim(str_extract(name,"(\\w+ )|(\\w{1}\\. )?+(\\w+$)"))

first
```

```
## [1] "Moe"           "C. Montgomery" "Timothy"       "Ned"
## [5] "Homer"         "Julius"
```

```
# So, the last name is either the word before the comma or
#   the word at the end of the string

last <- str_extract(name,"(\\w+,)|(\\w+$)")

last
```

```
## [1] "Szyslak"  "Burns,"   "Lovejoy"  "Flanders" "Simpson," "Hibbert"
```

```
# Now we can combine into one vector
firstLast <- str_c(first," ",last)

firstLast
```

```
## [1] "Moe Szyslak"          "C. Montgomery Burns," "Timothy Lovejoy"
## [4] "Ned Flanders"         "Homer Simpson,"       "Julius Hibbert"
```

(b) Construct a logical vector indicating whether a character has a title (i.e. Rev. and Dr.)

```
# For a title, we look for 2 or more letters and a period

title <- str_detect(name,"\\w{2,}\\.")

title
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE  TRUE
```

(c) Construct a logical vector indicating whether a character has a second name.

```
# To locate whether someone has a second name, we look for a period in the
#   first name from the initial

twoNames <- str_detect(first,"\\.")

twoNames
```

```
## [1] FALSE  TRUE FALSE FALSE FALSE FALSE
```

---

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

(a) `[0-9]+\\$`

This pattern matches one or more digits between 0 and 9 and a dollar sign. One example that would match: "129$"

(b) `\\b[a-z]{1,4}\\b`

This pattern matches words that are any combination of 1,2,3, or 4 lower-case letters. An example of a match here would be "adam"

(c) `.*?\\.txt$`

Text that matches this pattern would contain any character (or no characters!) with the string ".txt" at the end. The text "filename.txt" would be matched by this, for example.

(d) `\\d{2}/\\d{2}/\\d{4}`

This pattern matches 2 digits, a slash, 2 more digits, another slash, and 4 digits. This matches date strings such as "11/30/2017".

(e) `<(.+?)>.+?</\\1>`

This regex matches HTML tags such as `<tr>Row Title</tr>` since it looks for angle brackets with any characters inside, characters after that, and the same angle-bracket text with a / added to the beginning (closing tag).
