Week 3 Assignment

Elina Azrilyan

Data 607

09/13/2018

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

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

```
#install.packages("stringr", repos='http://cran.us.r-project.org')
library("stringr")
raw.data <- "555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-6542Rev.Timothy
Lovejoy555 8904Ned Flanders636-555-3226Simpson, Homer5553642Dr.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.

```
# This code removes all titles and middle initials
name2 <- str_replace(name, "\\s*\\w{1,3}[.]",  "")
name2
```

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

```
# This code reorders first and last names if they were in the wrong format.
#I started with the code found on the following help page: https://stackoverflow.com/
questions/33826650/last-name-first-name-to-first-name-last-name
str_replace(name2, "(\\w+),\\s(\\w+)","\\2 \\1")
```

```
## [1] "Moe Szyslak"      "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.).

```
name
```

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

```
str_detect(name, "\\s*\\w{2,3}[.]")
```

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

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

```
name
```

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

```
str_detect(name, "\\s{1}\\w{1}[.]")
```

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

Problem 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]+\$

Answer: It is 1 or more numbers followed by a “$”.

Examples are:

```
test<-"jfdlsae43543fjdk$slfjsdk2354325$jfdksaf45$"
unlist(str_extract_all(test, "[0-9]+\\$"))
```

```
## [1] "2354325$" "45$"
```

b. \b[a-z]{1,4}\b

Answer: It will return all lower case words between 1 and 4 letters.

Examples are:

```
test<-"What a lovely day it is, there is not even a rain drop. 43543rere 235"
unlist(str_extract_all(test, "\\b[a-z]{1,4}\\b"))
```

```
##  [1] "a"    "day"  "it"   "is"   "is"   "not"  "even" "a"    "rain" "drop"
```

c. .*?\.txt$

Answer: It will return experession of any length and any characters as long as it includes ".txt"" in the end.

Example is:

```
test<-"af45$ 4832$fjdsl something.txt gfdsgf54665$%&&ggfdsf.txt"
unlist(str_extract_all(test, ".*?\\.txt$"))
```

```
## [1] "af45$ 4832$fjdsl something.txt gfdsgf54665$%&&ggfdsf.txt"
```

d. \d{2}/\d{2}/\d{4}

Answer: This expression will return 2digits/2digits/4digits - similar to a date format.

Example is:

```
test<-"af45$ 4832$fjdsl something.txt 06/30/1984 sfjdskaj 34666 44"
unlist(str_extract_all(test, "\\d{2}/\\d{2}/\\d{4}"))
```

```
## [1] "06/30/1984"
```

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

Answer: This will return any character or multiple characters surrounded by < > followed by another character or any number of characters and then followed by </ and initial character.

Example:

```
test<- "fdsjkflsdaj <(sfssfdsfs)>   <tada>ffffaaaa</tada>"
unlist(str_extract_all(test, "<(.+?)>.+?</\\1>"))
```

```
## [1] "<tada>ffffaaaa</tada>"
```

Problem 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.

```
lcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0Tanwo
Uwisdij7Lj8kpf03AT5Idr3coc0bt7yczjatOaootj55t3Nj3ne6c4Sfek.r1w1YwwojigO
d6vrfUrbz2.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5
fy89n6Nd5t9kc4fE905gmc4Rgxo5nhDk!gr
```

```
coded.message <- "clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0T
anwoUwisdij7Lj8kpf03AT5Idr3coc0bt7yczjatOaootj55t3Nj3ne6c4Sfek.r1w1YwwojigOd6vrfUrbz2
.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5fy89n6Nd5t9kc4fE905gmc4R
gxo5nhDk!gr"
unlist(str_extract_all(coded.message, "[[:punct:]A-Z]"))
```

```
##  [1] "C" "O" "N" "G" "R" "A" "T" "U" "L" "A" "T" "I" "O" "N" "S" "." "Y"
## [18] "O" "U" "." "A" "R" "E" "." "A" "." "S" "U" "P" "E" "R" "N" "E" "R"
## [35] "D" "!"
```

Yes… Yes, I am:)