DATA 607 Assignment 3

Samuel I Kigamba

September 15, 2019

```
#install.packages("stringr",repos='http://mirrors.nics.utk.edu/cran/')
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 3.5.3
```

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

```
raw.data <- "555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-6542Rev. Timothy Lovejoy555 8904N

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"
```

```
phone <- unlist(str_extract_all(raw.data, "\\(?(\\d{3})?\\)?(-| )?\\d{3}(-| )?\\d{4}"))
phone
```

```
## [1] "555-1239"       "(636) 555-0113" "555-6542"       "555 8904"
## [5] "636-555-3226"   "5553642"
```

```
data.frame(name = name, phone = phone)
```

```
##                   name          phone
## 1          Moe Szyslak       555-1239
## 2 Burns, C. Montgomery (636) 555-0113
## 3 Rev. Timothy Lovejoy       555-6542
## 4         Ned Flanders       555 8904
## 5        Simpson,Homer   636-555-3226
## 6   Dr. Julius Hibbert        5553642
```

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

```
# First name for strings with commas
firtcomma <- str_trim(str_sub(name, start = str_locate(name, ",")[,1] + 1, end = str_length(name)))

# Last name for strings with commas
lastcomma <- str_trim(str_sub(name, start = 1, end = str_locate(name, ",")[,1] - 1))

# First name for strings without commas and a single space
firstspace <- str_sub(name, start = 1, end = str_locate(name, " ")[,1] - 1)
```

```r
# Last name for strings without commas and a single space
lastspace <- str_sub(name, start = str_locate(name, " ")[,1] + 1, end = str_length(name))

# Everything after the first space for all strings
afterspace <- str_sub(name, start = str_locate(name, " ")[,1] + 1, end = str_length(name))

# Extract First_Name
firstname <- ifelse(str_detect(name, ","), firtcomma, ifelse(str_count(name, " ") == 2, ifelse(str_dete

# Extract_Last_Name
lastname <- ifelse(str_detect(name, ","), lastcomma, ifelse(str_count(name, " ") == 2, ifelse(str_detec

# Formatted name
fullname <- str_c(firstname, " ", lastname)
fullname
```

```
## [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.).

```r
# search for the presence of a title
str_detect(name, "Maid|Madam|Mx.|Ms.|Miss|Mrs.|Mr.|Mister|Rev.|Reverend|Dr.|Doctor|Prof.|Professor|Fath
```

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

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

```r
name_count <- name

str_count(str_trim(str_replace_all(name_count, "Maid|Madam|Mx.|Ms.|Miss|Mrs.|Mr.|Mister|Rev.|Reverend|D
```

```
## [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]+\$

```r
amount <- "This laptop cost 999$ after a 20% discount from the manufacturer."
cost <- (unlist(str_extract_all(amount, "[0-9]+\\$")))
cost
```

```
## [1] "999$"
```

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

```r
amount <- "This laptop cost 999$ after a 20% discount from the manufacturer."
item <- (unlist(str_extract_all(amount, "\\b[a-z]{1,4}\\b")))
item
```

```
## [1] "cost" "a"    "from" "the"
```

(c) .*?\.txt$

```
# Create example strings that end in .txt
t <- c(".txt","alpha.txt","test.txt", "huge.txt", "99999.txt")

unlist(str_extract_all(t, ".*?\\.txt$"))
```

```
## [1] ".txt"      "alpha.txt" "test.txt"  "huge.txt"  "99999.txt"
```

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

```
amount <- "This laptop cost 999$ after a 20% discount from the manufacturer. The date of manufacture is
item <- (unlist(str_extract_all(amount, "\\d{2}/\\d{2}/\\d{4}")))
item
```

```
## [1] "12/12/2019" "12/12/2020"
```

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

```
# Create example strings of a HTML nature
t <- c("<h1>Report Heading</h1>","<h1></h1>","<p>S. Kigamba<br>89 Born<br>Sydney</p>","<img src='school
unlist(str_extract_all(t, "<(.+?)>.+?</\\1>"))
```

```
## [1] "<h1>Report Heading</h1>"
## [2] "<p>S. Kigamba<br>89 Born<br>Sydney</p>"
```

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.

```
rawdata <- ("clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0Tanwo
Uwisdij7Lj8kpf03AT5Idr3coc0bt7yczjat0aootj55t3Nj3ne6c4Sfek.r1w1YwwojigO
d6vrfUrbz2.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5
fy89n6Nd5t9kc4fE905gmc4Rgxo5nhDk!gr")
```

```
message <- unlist(str_extract_all(rawdata, "[[:upper:].]"))
combine <- paste(message, collapse = '')
secret_message <- str_replace_all(combine, "\\.", " ")
secret_message
```

```
## [1] "CONGRATULATIONS YOU ARE A SUPERNERD"
```