OmerOzeren\_HMW\_2\_DATA\_607

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

library(stringr)  
raw.data <-"555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-6542Rev. Timothy Lovejoy555 8904Ned Flanders636-555-3226Simpson, Homer5553642Dr. Julius Hibbert"  
names <- unlist(str\_extract\_all(raw.data, "[[:alpha:]., ]{2,}"))  
names

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

**3.1 Use the tools of this chapter to rearrange the vector so that all the elements conform to the standard first\_name last\_name format.**

In order to get standart first name ,last name we need to remove middle names and titles

**remove middle names :**

names\_no\_middle\_name <- sub(" [A-z]{1}\\. ","",names)  
names\_no\_middle\_name

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

**remove titles from the names :**

names\_final <- sub("[A-z]{2,3}\\. ","",names\_no\_middle\_name)  
names\_final

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

**DataFrame**

df.names <- data.frame(names\_final)  
df.names

## names\_final  
## 1 Moe Szyslak  
## 2 Burns,Montgomery  
## 3 Timothy Lovejoy  
## 4 Ned Flanders  
## 5 Simpson, Homer  
## 6 Julius Hibbert

**3.2 Construct a logical vector indicating whether a character has a title**

#Recall the original sample 'name2' from part a  
titles <- str\_detect(names\_no\_middle\_name, "[[:alpha:]]{2,}\\.")  
titles

## [1] FALSE FALSE TRUE FALSE FALSE TRUE

df.titles<- data.frame(names,titles)  
df.titles

## names titles  
## 1 Moe Szyslak FALSE  
## 2 Burns, C. Montgomery FALSE  
## 3 Rev. Timothy Lovejoy TRUE  
## 4 Ned Flanders FALSE  
## 5 Simpson, Homer FALSE  
## 6 Dr. Julius Hibbert TRUE

**3.3 Contruct a logical vector that indicates if a character has a second name**

secondname <- str\_detect(names, "[A-Z]\\.{1}")  
df.secondname <- data.frame(names,secondname)  
df.secondname

## names secondname  
## 1 Moe Szyslak FALSE  
## 2 Burns, C. Montgomery TRUE  
## 3 Rev. Timothy Lovejoy FALSE  
## 4 Ned Flanders FALSE  
## 5 Simpson, Homer FALSE  
## 6 Dr. Julius Hibbert FALSE

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

**4.1** [0-9]+$

**Any numbers 0-9 zero or more followed by the dollar $ string**

sample <- c("5748900000$","omer35$", "38$","38")   
expression = "[0-9]+\\$"  
str\_detect(sample, expression)

## [1] TRUE TRUE TRUE FALSE

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

**Any word that has anywhere between 1 to 4 letters**

sample\_2 <- c("car","cats","door", "hi", "datascience")  
expression\_2 <-"\\b[a-z]{1,4}\\b"  
str\_detect(sample\_2, expression\_2)

## [1] TRUE TRUE TRUE TRUE FALSE

**4.3** .\*?\.txt$

**Any string that ends with a .txt**

sample\_3 <- c("cars.txt", "txt", "timeseries.txt","code3434.txt")  
expression\_3 <-".\*?\\.txt$"  
str\_detect(sample\_3, expression\_3)

## [1] TRUE FALSE TRUE TRUE

**4.4** \d{2}/\d{2}/\d{4}

**Any Numbers that are written in format dd/dd/dddd**

sample\_4 <- c("100/1000/10000", "02/12/2019", "2/12/2019")  
expression\_4 <-"\\d{2}/\\d{2}/\\d{4}"  
str\_detect(sample\_4, expression\_4)

## [1] FALSE TRUE FALSE

**4.5** <(.+?)>.+?</\1>

**Text that starts and ends <> with and also at the end string starts with “/”"**

sample\_5 <- c("<omer>hello</omer>", "<omer>hello<omer>")  
expression\_5 <-"<(.+?)>.+?</\\1>"  
str\_detect(sample\_5, expression\_5)

## [1] TRUE FALSE

### 9) Extra Credit-The following code hides a secret message. Crack it with R and regular expressions.

code <-"clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0TanwoUwisdij7Lj8kpf03AT5Idr3coc0bt7yczjatOaootj55t3Nj3ne6c4Sfek.r1w1YwwojigOd6vrfUrbz2.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5fy89n6Nd5t9kc4fE905gmc4Rgxo5nhDk!gr"  
code

## [1] "clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0TanwoUwisdij7Lj8kpf03AT5Idr3coc0bt7yczjatOaootj55t3Nj3ne6c4Sfek.r1w1YwwojigOd6vrfUrbz2.2bkAnbhzgv4R9i05zEcrop.wAgnb.SqoU65fPa1otfb7wEm24k6t3sR9zqe5fy89n6Nd5t9kc4fE905gmc4Rgxo5nhDk!gr"

#Find all uppercase letters  
str\_extract\_all(code, "[[:upper:]]")

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