Stat 5080/6080 Data Technologies Fall 2020

Homework Assignment 2 (10/29/2020)

Niranjan Poudel (A02304550)

Submitted on (11/23/2020)

100 (+ 8 EC) Points — Due Monday 11/23/2020 (via Canvas by 11:59pm)

(i) (70 (+6 EC) Points) **Lecture Attendance Points:**
In this question, you have to process and analyze the saved Chat files from our Zoom lecture discussion meetings, in particular when students entered their "yes" answers for the Lecture Attendance Points (LAPs) that are awarded in this course.

Overall, there are 11 "clean" and 1 "raw" Chat files available in the `HW02_Data.zip` file. Download this zip file from Canvas and unzip it on your computer. You should see the Chat files in the resulting `HW02_Data/Chats` folder on your computer.

In a question like this, it will be impossible to check all your results. But, we are statisticians and can sample. How many lines are in a particular external Chat file? How many "yes" answers did a certain student provide in a certain lecture? Is the maximum number of LAPs correct? And so on ...

Feel free to work with a single Chat file, ideally `Chat_10_22_2020_Raw.txt`, first and then extend your R code to work with all Chat files later on once your code is working correctly for this single file.

You are not allowed to manually modify the data files using Word, Excel, or any other external software. All data processing has to be done in R. You are also not allowed to use certain constants, such as doing something for all xx lines in Chat file called ABC. Instead, obtain the length or dimension of that file and use this in your next step. Or work with functions from the apply family or use tidyverse functionality.

Overall, it is fine to answer this question using baseR functionality. However, working with pipes and tidyverse functionality will considerably help to shorten and simplify the R code you have to write. In case you work with pipes, check the result after each step. Do not only look at the content, but also at the type, e.g., whether something is a data.frame (or tibble), character string (or factor), and so on.

Always include your R code and show all your results as specified in each question part.

Finally, here is a list of useful help pages and questions & answers from stackoverflow and other sources that may be helpful when answering this question. Likely, you need to look up several other sources as well:

- `https://dplyr.tidyverse.org/reference/select.html`

- `https://dplyr.tidyverse.org/reference/filter.html`
- `https://dplyr.tidyverse.org/reference/mutate.html`
- `https://dplyr.tidyverse.org/reference/group_by.html`
- `https://tibble.tidyverse.org/reference/add_column.html`
- `https://tibble.tidyverse.org/reference/enframe.html`
- `https://tidyr.tidyverse.org/reference/spread.html`
- `https://tidyr.tidyverse.org/reference/nest.html`
- `https://lubridate.tidyverse.org/`
- `https://stackoverflow.com/questions/3397885/how-do-you-read-in-multiple-txt-files-into-r`
- `https://stackoverflow.com/questions/50777607/convert-list-of-list-object-to-dataframe-in-r`
- `https://stackoverflow.com/questions/10432993/named-list-to-from-data-frame`
- `https://stackoverflow.com/questions/10128617/test-if-characters-are-in-a-string`
- `https://rstudio-pubs-static.s3.amazonaws.com/221386_a6b7054b6536462fb3ba49e0341142e5.html`

You can earn up to **6 extra credit points** if you format your output in parts (b), (k), and (m) into a meaningful LaTeX table using the `kable` R function or the `xtable` R package. See here for useful information how to use these.

- `https://bookdown.org/yihui/rmarkdown-cookbook/kable.html`
- `https://bookdown.org/yihui/rmarkdown-cookbook/kableextra.html`
- `https://haozhu233.github.io/kableExtra/awesome_table_in_pdf.pdf`
- `https://cran.r-project.org/web/packages/xtable/vignettes/xtableGallery.pdf`

(a) (6 Points) First load all R packages for this question in this first part. Then read in the data from all 12 external Chat files. Arrange the data in a data.frame, called `chatText1`, with two variables: `name` should contain the path and file name from where each line of text originates. `value` should contain the lines of text (one–by–one) from the Chat files. Alternatively, you can use a tibble whenever I speak of a data.frame in this question. Show the head and tail of your `chatText1`.

Shown below is the result for the tail I got:

```
> tail(chatText1)
                                          name                                                       value
743 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27\t From  Sam Rands  to  Juergen Symanzik(Privately) : yes
744 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                        14:46:27\t From  Alex Ryan Ollerton : yes
745 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                          14:46:28\t From  Kristen Kay Sohm : yes
746 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                              14:46:30\t From  Will Raymer : Yes
747 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                             14:46:32\t From  Tyler Clayson : yes
748 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                     14:46:32\t From  Tristan James Peterson : yes
```

<u>Answer:</u>

```r
# Calling the packages required
library(plyr)
library(magrittr)
library(kableExtra)
library(tidyverse)
```

```r
# Reading file paths
filepath <- Sys.glob(file.path("HW02_Data/Chats", "*.txt"))

# Reading the data as a list
data <- lapply(filepath, function(x) read.delim(x, header = FALSE))

# Providing file path as the name to each data
names(data) <- filepath

# Arranging as data frame
ldply(data, data.frame, .id = "name") %>%
  mutate(value = paste(V1, V2, sep = "\t")) %>%
  select(name, value) ->
  chatText1

# Head and tail of the data
head(chatText1)

##                                         name
## 1 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 2 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 3 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 4 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 5 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 6 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
##
## 1                                        14:07:43\t From  Scott Greenberg : Is there a tool for scheduling offic
## 2 14:22:38\t From  Scott Greenberg : How much do we need to remember about different statistical distributions (
## 3
## 4
## 5
## 6
```

```r
tail(chatText1)
```

```
##                                         name
## 743 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 744 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 745 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 746 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 747 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 748 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
##                                                                value
## 743 14:46:27\t From  Sam Rands  to  Juergen Symanzik(Privately) : yes
## 744                         14:46:27\t From  Alex Ryan Ollerton : yes
## 745                           14:46:28\t From  Kristen Kay Sohm : yes
## 746                                14:46:30\t From  Will Raymer : Yes
## 747                             14:46:32\t From  Tyler Clayson : yes
## 748                     14:46:32\t From  Tristan James Peterson : yes
```

(b) (4 Points) How many lines did we read in from each of the 12 Chat files? This might be a good place to manually check that your code works correctly so far by counting the number of lines in some of the Chat files (or by opening some of the files in a text editor or in RStudio where line numbers are shown).

Answer:

```r
# Getting name of the file
filename <- names(sapply(data, nrow))

# Getting the number of rows
rows <- as.numeric(sapply(data, nrow))

# Combining them as dataframe
dt <- cbind(filename, rows)

# Output in a formatted table
dt %>%
  kbl(col.names = c("Filename with file path", "Number of rows"),
      align = "clc", valign = "t",
      caption = "Filename (filepath) and number of rows in each file.") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 1: Filename (filepath) and number of rows in each file.

Filename with file path	Number of rows
HW02_Data/Chats/Chat_09_15_2020_Clean.txt	68
HW02_Data/Chats/Chat_09_17_2020_Clean.txt	70
HW02_Data/Chats/Chat_09_22_2020_Clean.txt	62
HW02_Data/Chats/Chat_09_24_2020_Clean.txt	61
HW02_Data/Chats/Chat_09_29_2020_Clean.txt	67
HW02_Data/Chats/Chat_10_01_2020_Clean.txt	58
HW02_Data/Chats/Chat_10_06_2020_Clean.txt	50
HW02_Data/Chats/Chat_10_08_2020_Clean.txt	64
HW02_Data/Chats/Chat_10_13_2020_Clean.txt	55
HW02_Data/Chats/Chat_10_15_2020_Clean.txt	68
HW02_Data/Chats/Chat_10_20_2020_Clean.txt	63
HW02_Data/Chats/Chat_10_22_2020_Raw.txt	62

(c) (4 Points) Using regular expressions, extract the date information from the `name` variable from your `chatText1` data.frame and add this as an additional variable called `date` to the new `chatText2` data.frame. The date information should be formatted as `mm/dd/yyyy`. Keep the date information as character (or factor) and do not into any of the date classes in R in this step. Show the head and tail of your `chatText2`.

Shown below is the result for the tail I got:

```
> tail(chatText2)
                                         name                                                               value       date
743 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27\t From  Sam Rands  to  Juergen Symanzik(Privately) : yes 10/22/2020
744 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                           14:46:27\t From  Alex Ryan Ollerton : yes 10/22/2020
745 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                             14:46:28\t From  Kristen Kay Sohm : yes 10/22/2020
746 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                                  14:46:30\t From  Will Raymer : Yes 10/22/2020
747 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                                14:46:32\t From  Tyler Clayson : yes 10/22/2020
748 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt                       14:46:32\t From  Tristan James Peterson : yes 10/22/2020
```

<u>Answer:</u>

```
# Extracting the date information
chatText1 %>%
  mutate(date = str_extract(name, "\\d+_\\d+_\\d+")) %>%
  mutate(date = gsub("_", "/", date)) ->
  chatText2

# Head and tail of the data
head(chatText2)

##                                      name
## 1 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 2 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 3 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 4 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 5 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
## 6 HW02_Data/Chats/Chat_09_15_2020_Clean.txt
##
## 1                                        14:07:43\t From  Scott Greenberg : Is there a tool for scheduling offic
## 2 14:22:38\t From  Scott Greenberg : How much do we need to remember about different statistical distributions (
## 3
## 4
## 5
## 6
##         date
## 1 09/15/2020
## 2 09/15/2020
## 3 09/15/2020
## 4 09/15/2020
## 5 09/15/2020
## 6 09/15/2020

tail(chatText2)

##                                      name
## 743 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 744 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 745 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 746 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 747 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
## 748 HW02_Data/Chats/Chat_10_22_2020_Raw.txt
##                                                                 value       date
## 743 14:46:27\t From  Sam Rands  to  Juergen Symanzik(Privately) : yes 10/22/2020
## 744                        14:46:27\t From  Alex Ryan Ollerton : yes 10/22/2020
## 745                          14:46:28\t From  Kristen Kay Sohm : yes 10/22/2020
## 746                               14:46:30\t From  Will Raymer : Yes 10/22/2020
## 747                             14:46:32\t From  Tyler Clayson : yes 10/22/2020
## 748                    14:46:32\t From  Tristan James Peterson : yes 10/22/2020

# Checking for if date is character or not
is.character(chatText2$date)

## [1] TRUE
```

(d) (4 Points) Using regular expressions and/or string operations, extract the time, sender (the from–part), and chat text information from the `value` variable from your `chatText2` data.frame. Add this information as three additional variables called `time`, `student`, and `comment` to the new `chatText3` data.frame. In this step, still leave the information in the `student` variable whether something was sent privately to me. You can keep the original `value` variable in your new data frame or simply delete it. Show the head and tail of your `chatText3`.

Shown below is the result for the tail I got:

```
> tail(chatText3)
                                         name     time                                    student comment       date
743 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27 Sam Rands  to  Juergen Symanzik(Privately)     yes 10/22/2020
744 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27                         Alex Ryan Ollerton     yes 10/22/2020
745 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:28                           Kristen Kay Sohm     yes 10/22/2020
746 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:30                                Will Raymer     Yes 10/22/2020
747 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:32                              Tyler Clayson     yes 10/22/2020
748 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:32                     Tristan James Peterson     yes 10/22/2020
```

Answer:

```
chatText2 %>%
  mutate(time = str_extract(value, "\\d+:\\d+:\\d+")) %>%
  mutate(student = gsub(".*From  ", "", value)) %>%
  mutate(student = gsub(" :.*", "", student)) %>%
  mutate(comment = gsub(".*From ", "", value)) %>%
  mutate(comment = gsub(".*: ", "", comment)) %>%
  .[c("name", "time", "student", "comment", "date")] ->
  chatText3

# Head and tail of the data
head(chatText3)

##                                         name     time             student
## 1 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:07:43     Scott Greenberg
## 2 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:22:38     Scott Greenberg
## 3 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:51              landon
## 4 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:52     Scott Greenberg
## 5 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:52 Justin Kory Wheeler
## 6 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:53    Kristen Kay Sohm
##
## 1                                        Is there a tool for scheduling office hours with you or do we just emai
## 2 How much do we need to remember about different statistical distributions (like how much operational knowledge
## 3
## 4
## 5
## 6
##          date
## 1 09/15/2020
## 2 09/15/2020
## 3 09/15/2020
## 4 09/15/2020
## 5 09/15/2020
## 6 09/15/2020

tail(chatText3)

##                                         name     time
## 743 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:27
## 744 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:27
## 745 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:28
## 746 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:30
## 747 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:32
## 748 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:32
##                                         student comment       date
## 743 Sam Rands  to  Juergen Symanzik(Privately)     yes 10/22/2020
## 744                        Alex Ryan Ollerton     yes 10/22/2020
## 745                          Kristen Kay Sohm     yes 10/22/2020
## 746                               Will Raymer     Yes 10/22/2020
## 747                             Tyler Clayson     yes 10/22/2020
## 748                    Tristan James Peterson     yes 10/22/2020
```

(e) (8 Points) Using regular expressions and time/date functions, further extend your data.frame: Add four additional variables called `datetime`, `studentname`, `isprivate`, and `containsyes` to the new `chatText4` data.frame. `datetime` should be a date/time object of class POSIXct and/or POSIXt. `studentname` should just be the name of a student, without the information whether that Chat line was sent privately to me or someone else. `isprivate` should be a logical vector that is TRUE of something was sent privately. `containsyes` should be a logical vector that is TRUE when the line contains the word "yes" in any capitalization. There could be additional text in this line, e.g., times, numbers, spaces, exclamation marks, and more. Just ignore those. All that matters is that the word "yes" appears in the line. So, "yessayer" and "yes (12:40pm)" both should result in TRUE. You can keep the original `student` variable in your new data frame or simply delete it. Show the head and tail of your `chatText4`.

Shown below is the result for the tail I got:

```
> tail(chatText4)
                                         name     time                                     student
743 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27 Sam Rands  to  Juergen Symanzik(Privately)
744 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:27                          Alex Ryan Ollerton
745 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:28                            Kristen Kay Sohm
746 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:30                                 Will Raymer
747 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:32                               Tyler Clayson
748 HW02_Data\\Chats\\Chat_10_22_2020_Raw.txt 14:46:32                      Tristan James Peterson
    comment       date            datetime            studentname isprivate containsyes
743     yes 10/22/2020 2020-10-22 14:46:27              Sam Rands      TRUE        TRUE
744     yes 10/22/2020 2020-10-22 14:46:27     Alex Ryan Ollerton     FALSE        TRUE
745     yes 10/22/2020 2020-10-22 14:46:28       Kristen Kay Sohm     FALSE        TRUE
746     Yes 10/22/2020 2020-10-22 14:46:30            Will Raymer     FALSE        TRUE
747     yes 10/22/2020 2020-10-22 14:46:32          Tyler Clayson     FALSE        TRUE
748     yes 10/22/2020 2020-10-22 14:46:32 Tristan James Peterson     FALSE        TRUE
```

Answer:

```
chatText3 %>%
  mutate(datetime = as.POSIXct(strptime(paste(gsub("/", "-", date),
                                              time), "%m-%d-%Y %H:%M:%S"))) %>%
  mutate(studentname = gsub("  to.*", "", student)) %>%
  mutate(isprivate = grepl("(Privately)", student)) %>%
  mutate(containsyes = grepl("yes", ignore.case = TRUE, comment)) ->
  chatText4

# Head and tail of the data
head(chatText4)

##                                         name     time             student
## 1 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:07:43     Scott Greenberg
## 2 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:22:38     Scott Greenberg
## 3 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:51              landon
## 4 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:52     Scott Greenberg
## 5 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:52 Justin Kory Wheeler
## 6 HW02_Data/Chats/Chat_09_15_2020_Clean.txt 14:37:53    Kristen Kay Sohm
##
## 1                                        Is there a tool for scheduling office hours with you or do we just emai
## 2 How much do we need to remember about different statistical distributions (like how much operational knowledge
## 3
## 4
## 5
## 6
##         date            datetime         studentname isprivate containsyes
## 1 09/15/2020 2020-09-15 14:07:43     Scott Greenberg     FALSE       FALSE
## 2 09/15/2020 2020-09-15 14:22:38     Scott Greenberg     FALSE       FALSE
## 3 09/15/2020 2020-09-15 14:37:51              landon     FALSE        TRUE
## 4 09/15/2020 2020-09-15 14:37:52     Scott Greenberg     FALSE        TRUE
## 5 09/15/2020 2020-09-15 14:37:52 Justin Kory Wheeler     FALSE        TRUE
## 6 09/15/2020 2020-09-15 14:37:53    Kristen Kay Sohm     FALSE        TRUE
```

```r
tail(chatText4)
```

```
##                                        name     time
## 743 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:27
## 744 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:27
## 745 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:28
## 746 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:30
## 747 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:32
## 748 HW02_Data/Chats/Chat_10_22_2020_Raw.txt 14:46:32
##                                        student comment       date
## 743 Sam Rands  to  Juergen Symanzik(Privately)     yes 10/22/2020
## 744                          Alex Ryan Ollerton     yes 10/22/2020
## 745                            Kristen Kay Sohm     yes 10/22/2020
## 746                                 Will Raymer     Yes 10/22/2020
## 747                               Tyler Clayson     yes 10/22/2020
## 748                       Tristan James Peterson     yes 10/22/2020
##                datetime            studentname isprivate containsyes
## 743 2020-10-22 14:46:27              Sam Rands      TRUE        TRUE
## 744 2020-10-22 14:46:27     Alex Ryan Ollerton     FALSE        TRUE
## 745 2020-10-22 14:46:28       Kristen Kay Sohm     FALSE        TRUE
## 746 2020-10-22 14:46:30            Will Raymer     FALSE        TRUE
## 747 2020-10-22 14:46:32          Tyler Clayson     FALSE        TRUE
## 748 2020-10-22 14:46:32 Tristan James Peterson     FALSE        TRUE
```

(f) (6 Points) How many different unique studentnames do we have? List them in alphabetical sorting based on first names. Alex should be first and Yuxin should be last.

At the start of the semester, we had 21 students. So, we need to combine several of the Zoom user IDs to match actual studentnames. Using regular expressions, modify the original `studentname` variable.

You can start with the code chunk below, but this is not complete and needs further modifications. Look carefully! Moreover, write an additional general regular expression that replaces all middle initials and middle names and just keeps the first part of a studentname and the final part of a studentname. So, you cannot explicitly say to remove "James" in "Tristan James Peterson", but your regular expression must do this as well for "Tristan J Peterson" or maybe even "Tristan Jim Peterson". Show all of your resulting modified studentnames.

Here is the initial code chunk to start with:

```
chatText4 %>%
  mutate(studentname = gsub("Darryl Phillip Gatrell", "Darryl Gatrell", studentname)) %>%
  mutate(studentname = gsub("Jonathan Maxwell Diaz", "Jonathan Diaz", studentname)) %>%
  mutate(studentname = gsub("joshrwatt", "Josh Watteyne", studentname)) %>%
  mutate(studentname = gsub("Joshua Rick Watteyne", "Josh Watteyne", studentname)) %>%
  mutate(studentname = gsub("landon", "Landon Erik Anderson", studentname)) %>%
  mutate(studentname = gsub("Samuel Winter Rands", "Sam Rands", studentname)) %>%
  mutate(studentname = gsub("Scott G", "Scott Greenberg", studentname)) %>%
  mutate(studentname = gsub("Sydney Kay Geisler", "Sydney Geisler", studentname)) %>%
  mutate(studentname = gsub("Tyler", "Tyler Clayson", studentname)) %>%
  mutate(studentname = gsub("Will Raymer", "William Raymer", studentname)) %>%
  mutate(studentname = gsub("William Samson Raymer", "William Raymer", studentname)) %>%
  mutate(studentname = gsub("xiaomeng Zhang", "Xiaomeng Zhang", studentname)) %>%
  mutate(studentname = gsub(..., studentname)) ->
  chatText5
```

Shown below is the result for the modified studentnames I got:

```
> sort(unique(chatText5$studentname))
 [1] "Alex Ollerton"    "Alyssa Cable"     "Darryl Gatrell"   "Demi Culianos"    "Dirk Broadhead"
 [6] "Jonathan Diaz"    "Jorge Hernandez"  "Josh Watteyne"    "Justin Wheeler"   "Kristen Sohm"
[11] "Landon Anderson"  "Niranjan Poudel"  "Sam Rands"        "Scott Greenberg"  "Sydney Geisler"
[16] "Tristan Peterson" "Tyler Clayson"    "Varsha Mandadi"   "William Raymer"   "Xiaomeng Zhang"
[21] "Yuxin Chen"
```

Answer:

```
# Creating unique student names
chatText4 %>%
  mutate(studentname = gsub("landon", "Landon Erik Anderson",
                            studentname)) %>%
  mutate(studentname = gsub("joshrwatt", "Joshua Watteyne",
                            studentname)) %>%
  mutate(studentname = gsub("\\<Tyler\\>", "Tyler Clayson",
                            studentname)) %>%
  mutate(studentname = gsub("Will Raymer", "William Raymer",
                            studentname)) %>%
  mutate(studentname = gsub("xiaomeng Zhang", "Xiaomeng Zhang",
                            studentname)) %>%
  mutate(studentname = gsub("\\<Scott G\\>", "Scott Greenberg",
                            studentname)) %>%
  mutate(studentname = gsub("Sam Rands", "Samuel Winter Rands",
                            studentname)) %>%
  mutate(studentname = gsub("^(\\w+).*\\b(\\w+)$", "\\1 \\2",
                            studentname)) %>%
  mutate(studentname = gsub("Joshua Watteyne", "Josh Watteyne",
                            studentname)) %>%
  mutate(studentname = gsub("Samuel Rands", "Sam Rands",
                            studentname)) ->
  chatText5

# Display unique names as sorted
sort(unique(chatText5$studentname))

##  [1] "Alex Ollerton"    "Alyssa Cable"     "Darryl Gatrell"   "Demi Culianos"
##  [5] "Dirk Broadhead"   "Jonathan Diaz"    "Jorge Hernandez"  "Josh Watteyne"
##  [9] "Justin Wheeler"   "Kristen Sohm"     "Landon Anderson"  "Niranjan Poudel"
## [13] "Sam Rands"        "Scott Greenberg"  "Sydney Geisler"   "Tristan Peterson"
## [17] "Tyler Clayson"    "Varsha Mandadi"   "William Raymer"   "Xiaomeng Zhang"
## [21] "Yuxin Chen"
```

(g) (4 Points) Starting with `chatText5`, show the variables `datetime`, `studentname`, and `comment` that are private. There should be 21 resulting rows. Keep in mind that I only kept private replies in our very last Chat file. All private replies from earlier Chat files have been removed before I provided you with access to our Zoom Chat files.

Answer:

```
# Selecting and displaying only three columns
chatText5 %>%
  filter(isprivate == TRUE) %>%
  select(datetime, studentname, comment)

##                datetime      studentname comment
## 1  2020-10-22 14:11:30   Sydney Geisler     yes
## 2  2020-10-22 14:11:31     Alyssa Cable     yes
## 3  2020-10-22 14:11:32       Yuxin Chen     Yes
## 4  2020-10-22 14:11:33   Darryl Gatrell     yes
## 5  2020-10-22 14:11:36  Scott Greenberg     yes
## 6  2020-10-22 14:11:36   Dirk Broadhead     yes
## 7  2020-10-22 14:11:36  Jorge Hernandez     yes
## 8  2020-10-22 14:11:38  Landon Anderson     yes
## 9  2020-10-22 14:11:41   Xiaomeng Zhang     Yes
## 10 2020-10-22 14:11:45    Tyler Clayson     Yes
## 11 2020-10-22 14:12:45    Josh Watteyne     yes
## 12 2020-10-22 14:41:40   William Raymer     Yes
## 13 2020-10-22 14:41:41    Alex Ollerton     yes
## 14 2020-10-22 14:41:43   Xiaomeng Zhang     Yes
## 15 2020-10-22 14:41:44        Sam Rands     yes
## 16 2020-10-22 14:41:45  Niranjan Poudel     Yes
## 17 2020-10-22 14:41:53     Kristen Sohm     yes
## 18 2020-10-22 14:41:54    Josh Watteyne     yes
## 19 2020-10-22 14:41:56  Scott Greenberg     yes
## 20 2020-10-22 14:42:01 Tristan Peterson     yes
## 21 2020-10-22 14:46:27        Sam Rands     yes
```

(h) (4 Points) Starting with `chatText5`, show the variables `datetime`, `studentname`, and `comment` that do not contain yes. There should be 63 resulting rows. Use an R function that truncates `comment` to a maximum of 30 characters that get printed. Suppress any additional characters here.

Answer:

```
# Displaying required columns
chatText5 %>%
  filter(containsyes == FALSE) %>%
  select(datetime, studentname, comment) %>%
  mutate(comment = substr(comment, start = 1, stop = 30))
```

```
##               datetime      studentname                            comment
## 1  2020-09-15 14:07:43  Scott Greenberg     Is there a tool for scheduling
## 2  2020-09-15 14:22:38  Scott Greenberg     How much do we need to remembe
## 3  2020-09-15 14:39:04     Kristen Sohm     Is using knitr to convert an r
## 4  2020-09-15 14:53:05   Xiaomeng Zhang     Do you have a deadline to upgr
## 5  2020-09-15 14:54:05   Xiaomeng Zhang              Good to know, thanks.
## 6  2020-09-15 15:01:30  Scott Greenberg     You say at least schedule offi
## 7  2020-09-15 15:03:59  Scott Greenberg                             gotcha
## 8  2020-09-15 15:12:33  Scott Greenberg     Is TeX Live instead of MikTeX,
## 9  2020-09-15 15:17:48   Xiaomeng Zhang                             Thanks
## 10 2020-09-15 15:17:48   Varsha Mandadi                       Thank you..!
## 11 2020-09-17 14:01:32       Yuxin Chen                                yea
## 12 2020-09-17 14:28:21        Sam Rands     How would the R script handle 
## 13 2020-09-17 14:37:44  Scott Greenberg     For my intro stats class we di
## 14 2020-09-17 14:38:16  Scott Greenberg     what was recorded was whether 
## 15 2020-09-17 14:38:55  Scott Greenberg     no what flavor it was (grape l
## 16 2020-09-17 14:40:34   Justin Wheeler     Would data technologies be use
## 17 2020-09-17 14:44:57        Sam Rands     I know UVU parking has a simil
## 18 2020-09-17 14:55:32   Varsha Mandadi                          Thank you
## 19 2020-09-22 14:36:37   Dirk Broadhead     Is there is a certain number o
## 20 2020-09-22 14:39:18  Scott Greenberg             Someone had a question
## 21 2020-09-24 14:53:06   Xiaomeng Zhang                             THanks
## 22 2020-09-29 14:12:30     Kristen Sohm                             1 or 2
## 23 2020-09-29 14:13:43   Dirk Broadhead                     11, 12, 21, 22
## 24 2020-09-29 14:15:01   Dirk Broadhead                             1 or 2
## 25 2020-09-29 14:15:25    Alex Ollerton                      2 or 3  or  4
## 26 2020-09-29 14:15:25   Dirk Broadhead                            2, 3, 4
## 27 2020-09-29 14:19:24   Dirk Broadhead     Can you explain how we fill ou
## 28 2020-09-29 14:20:22        Sam Rands     So we donâ\200\231t sum the 1st roll
## 29 2020-09-29 14:38:43        Sam Rands     For what itâ\200\231s worth, if you
## 30 2020-10-01 14:10:48     Kristen Sohm                               chat
## 31 2020-10-06 14:14:13    Tyler Clayson     So for determining the probabi
## 32 2020-10-08 14:10:13     Kristen Sohm                            23 secs
## 33 2020-10-08 14:10:39    Alex Ollerton                            10 secs
## 34 2020-10-08 14:11:02    Alex Ollerton       you could get 0 as an answer
## 35 2020-10-08 14:13:24        Sam Rands                    3600 * 24 * 365
## 36 2020-10-08 14:14:02        Sam Rands                           31536000
## 37 2020-10-08 14:14:14        Sam Rands                            seconds
## 38 2020-10-08 14:14:52   Dirk Broadhead                            3153600
## 39 2020-10-08 14:14:53  Niranjan Poudel                            3153600
## 40 2020-10-08 14:18:27     Kristen Sohm                                chat
## 41 2020-10-13 14:59:03  Niranjan Poudel                              got it
## 42 2020-10-15 14:15:59        Sam Rands     Google searching error message
## 43 2020-10-15 14:20:34     Alyssa Cable     Can you talk about the very la
## 44 2020-10-15 14:21:03     Alyssa Cable     In particular, when we comment
## 45 2020-10-15 14:25:21     Alyssa Cable                          Thank you!
## 46 2020-10-15 14:26:21    Alex Ollerton     Do we want to remove 999999 an
## 47 2020-10-15 14:29:53    Tyler Clayson                      ok, thank you!
## 48 2020-10-15 14:40:53  Niranjan Poudel     What if you place a letter som
## 49 2020-10-15 14:42:50  Niranjan Poudel             Can we try last one to
## 50 2020-10-15 15:05:54 Tristan Peterson     Has anyone ever mined twitter
## 51 2020-10-15 15:13:40   Sydney Geisler     I canâ\200\231t stay after on Tuesda
## 52 2020-10-20 14:17:02  Niranjan Poudel                            remove ^
## 53 2020-10-20 14:18:00        Sam Rands     Could you do space * before an
## 54 2020-10-20 14:19:11        Sam Rands                *[d|D][o|O][g|G] *
## 55 2020-10-20 14:22:33  Niranjan Poudel                             at both
## 56 2020-10-20 14:23:34  Niranjan Poudel            grep(\\\\<[dD][oO][gG]$
## 57 2020-10-20 14:23:42        Sam Rands          \\\\<[d|D][o|O][g|G]\\\\>
## 58 2020-10-20 14:23:51 Tristan Peterson \\\\<[Dd][Oo][gG]\\\\> seems to wo
## 59 2020-10-20 14:29:38   Dirk Broadhead grep(\\\\<dog\\\\>, practiceString
## 60 2020-10-20 14:42:12  Niranjan Poudel     Can you explain line 241 and 2
## 61 2020-10-20 14:44:36  Niranjan Poudel                              got it
## 62 2020-10-22 14:11:23  Niranjan Poudel                             A to M?
## 63 2020-10-22 14:12:39        Sam Rands     Is there any way to get the ch
```

(i) (4 Points) Starting with `chatText5`, group by date and by studentname and count the number of "yes" answers for that date for each student. Create a new data.frame, called `lapCountByDateAndStudent`, that contains the variables `date`, `studentname`, and `numlaps`. Show the head and tail of your `lapCountByDateAndStudent`.

Shown below is the result for the tail I got:

```
> tail(lapCountByDateAndStudent)
          date      studentname numlaps
225 10/22/2020   Sydney Geisler       4
226 10/22/2020 Tristan Peterson       4
227 10/22/2020    Tyler Clayson       3
228 10/22/2020   William Raymer       4
229 10/22/2020   Xiaomeng Zhang       4
230 10/22/2020       Yuxin Chen       3
```

Answer:

```
# Getting count of yes for each student by date
chatText5 %>%
  group_by(date, studentname) %>%
  summarise(numlaps = sum(containsyes == TRUE)) %>%
  as.data.frame ->
  lapCountByDateAndStudent

## `summarise()` regrouping output by 'date' (override with `.groups` argument)

# Head and tail of the data
head(lapCountByDateAndStudent)

##         date     studentname numlaps
## 1 09/15/2020    Alyssa Cable       3
## 2 09/15/2020  Darryl Gatrell       3
## 3 09/15/2020   Demi Culianos       3
## 4 09/15/2020  Dirk Broadhead       3
## 5 09/15/2020   Jonathan Diaz       3
## 6 09/15/2020 Jorge Hernandez       3

tail(lapCountByDateAndStudent)

##           date      studentname numlaps
## 225 10/22/2020   Sydney Geisler       4
## 226 10/22/2020 Tristan Peterson       4
## 227 10/22/2020    Tyler Clayson       3
## 228 10/22/2020   William Raymer       4
## 229 10/22/2020   Xiaomeng Zhang       4
## 230 10/22/2020       Yuxin Chen       3
```

(j) (4 Points) Use your `lapCountByDateAndStudent` data.frame. Are there any students who left early after just one "yes" answer on a certain date? For which students and when did this happen?

Moreover, are there any students who "cheated" and provided four or more "yes" answers on a certain date? For which students and when did this happen? Recall that I encouraged students at the end of our 10/22/2020 Zoom lecture discussion meeting to answer "yes" a fourth time. So, we should have at least a few students with four or more "yes" answers on that date.

Answer:

```
# who left early (after only 1 yes answer)?
lapCountByDateAndStudent %>%
  filter(numlaps == 1)

##         date    studentname numlaps
## 1 09/15/2020 Sydney Geisler       1

# who "cheated" (with 4 or more yes answers)?
lapCountByDateAndStudent %>%
  filter(numlaps > 3)

##          date      studentname numlaps
## 1  09/15/2020  Niranjan Poudel       4
## 2  09/17/2020  Scott Greenberg       4
## 3  09/22/2020  Scott Greenberg       4
## 4  10/13/2020  Scott Greenberg       4
## 5  10/15/2020  Niranjan Poudel       4
## 6  10/15/2020   William Raymer       4
## 7  10/22/2020    Alex Ollerton       4
## 8  10/22/2020    Josh Watteyne       4
## 9  10/22/2020  Scott Greenberg       4
## 10 10/22/2020   Sydney Geisler       4
## 11 10/22/2020 Tristan Peterson       4
## 12 10/22/2020   William Raymer       4
## 13 10/22/2020   Xiaomeng Zhang       4
```

(k) (4 Points) Use your `lapCountByDateAndStudent` data.frame. We are ultimately interested in the total number of "yes" answers for each student.

I do not want to penalize students at this time for answering "yes" four or more times, but I also do not want to award them for doing so. Therefore, three is the maximum allowed number of "yes" answers for a student on any given date.

Group by studentname and calculate the total number of "yes" answers for each student, but keep in mind that three is the maximum number of "yes" answers per date.

Show the results for all students. Recall what the maximum for the total can be at most.

<u>Answer:</u>

```
# Displaying total number of yes per student
lapCountByDateAndStudent %>%
  mutate(numlaps = replace(numlaps, numlaps > 3, 3)) %>%
  group_by(studentname) %>%
  summarise(numlapstotal = sum(numlaps)) %>%
  as.data.frame %>%
  kbl(col.names = c("Student Name", "Total Yes"),
      caption = "Total number of 'Yes' per student") %>%
  kable_styling(latex_options = c("striped", "hold_position"))

## `summarise()` ungrouping output (override with `.groups` argument)
```

Table 2: Total number of 'Yes' per student

Student Name	Total Yes
Alex Ollerton	32
Alyssa Cable	34
Darryl Gatrell	36
Demi Culianos	18
Dirk Broadhead	36
Jonathan Diaz	34
Jorge Hernandez	35
Josh Watteyne	36
Justin Wheeler	27
Kristen Sohm	31
Landon Anderson	34
Niranjan Poudel	36
Sam Rands	35
Scott Greenberg	33
Sydney Geisler	34
Tristan Peterson	35
Tyler Clayson	35
Varsha Mandadi	5
William Raymer	36
Xiaomeng Zhang	34
Yuxin Chen	36

(l) (4 Points) Starting with `chatText5`, group by studentname and by date and count the number of "yes" answers for that date for each student. Create a new data.frame, called `lapCountByStudentAndDate`, that contains the variables `studentname`, `date`, and `numlaps`. **Note that this is different from what we did before in part (i)!** Show the head and tail of your `lapCountByStudentAndDate`.

Shown below is the result for the tail I got:

```
> tail(lapCountByStudentAndDate)
    studentname       date numlaps
225  Yuxin Chen 10/06/2020       3
226  Yuxin Chen 10/08/2020       3
227  Yuxin Chen 10/13/2020       3
228  Yuxin Chen 10/15/2020       3
229  Yuxin Chen 10/20/2020       3
230  Yuxin Chen 10/22/2020       3
```

Answer:

```
# Number of yes per student per date
chatText5 %>%
  group_by(studentname, date) %>%
  summarise(numlaps = sum(containsyes == TRUE)) %>%
  as.data.frame ->
  lapCountByStudentAndDate

## `summarise()` regrouping output by 'studentname' (override with `.groups` argument)

# Head and tail of data
head(lapCountByStudentAndDate)

##     studentname       date numlaps
## 1 Alex Ollerton 09/17/2020       3
## 2 Alex Ollerton 09/22/2020       2
## 3 Alex Ollerton 09/24/2020       3
## 4 Alex Ollerton 09/29/2020       3
## 5 Alex Ollerton 10/01/2020       3
## 6 Alex Ollerton 10/06/2020       3

tail(lapCountByStudentAndDate)

##     studentname       date numlaps
## 225  Yuxin Chen 10/06/2020       3
## 226  Yuxin Chen 10/08/2020       3
## 227  Yuxin Chen 10/13/2020       3
## 228  Yuxin Chen 10/15/2020       3
## 229  Yuxin Chen 10/20/2020       3
## 230  Yuxin Chen 10/22/2020       3
```

(m) (4 Points) Starting with `lapCountByStudentAndDate`, transfer this data.frame into a new "wide" (Excel–spreadsheet like) data.frame, called `lapCountByStudentAndDateWide`, where the rows are the studentnames and the columns are the different dates. The cells are the counts of the "yes" answers for a student on a given date. Keep the original counts, so if a cell shows four or more, that is fine. This may allow to detect students who tried to "cheat" systematically, e.g., after having missed a Zoom lecture discussion meeting. Show the entire `lapCountByStudentAndDateWide`.

Shown below is the result for the tail (and not the entire data.frame) I got:

```
> tail(lapCountByStudentAndDateWide)
        studentname 09/15/2020 09/17/2020 09/22/2020 09/24/2020 09/29/2020 10/01/2020
16 Tristan Peterson          2          3          3          3          3          3
17    Tyler Clayson          3          3          3          3          2          3
18   Varsha Mandadi          3          2         NA         NA         NA         NA
19   William Raymer          3          3          3          3          3          3
20   Xiaomeng Zhang          3          2          3          3          3          2
21       Yuxin Chen          3          3          3          3          3          3
   10/06/2020 10/08/2020 10/13/2020 10/15/2020 10/20/2020 10/22/2020
16          3          3          3          3          3          4
17          3          3          3          3          3          3
18         NA         NA         NA         NA         NA         NA
19          3          3          3          4          3          4
20          3          3          3          3          3          4
21          3          3          3          3          3          3
```

Answer:

```r
# Creating wide dataframe
lapCountByStudentAndDate %>%
  pivot_wider(id_cols = studentname,
              names_from = date, values_from = numlaps) %>%
  as.data.frame %>%
  .[, c(1, 13, 2:12)] ->
  lapCountByStudentAndDateWide

# Displaying as a table
lapCountByStudentAndDateWide %>%
  kbl(align = "clc", valign = "t",
      caption = "Yes count per student per date") %>%
  kable_styling(latex_options = c("striped", "hold_position",
                                  "scale_down"), position = "center") %>%
  landscape()
```

Table 3: Yes count per student per date

studentname	09/15/2020	09/17/2020	09/22/2020	09/24/2020	09/29/2020	10/01/2020	10/06/2020	10/08/2020	10/13/2020	10/15/2020	10/20/2020	10/22/2020
Alex Ollerton	NA	3	2	3	3	3	3	3	3	3	3	4
Alyssa Cable	3	3	3	3	3	3	3	2	3	3	2	3
Darryl Gatrell	3	3	3	3	3	3	3	3	3	3	3	3
Demi Culianos	3	3	3	3	3	3	NA	NA	NA	NA	NA	NA
Dirk Broadhead	3	3	3	3	3	3	3	3	3	3	3	3
Jonathan Diaz	3	3	3	3	3	2	2	3	3	3	3	3
Jorge Hernandez	3	3	3	3	3	3	2	3	3	3	3	3
Josh Watteyne	3	3	3	3	3	3	3	3	3	3	3	4
Justin Wheeler	3	3	3	3	3	3	3	3	NA	3	NA	NA
Kristen Sohm	3	3	3	3	3	2	NA	3	3	3	3	2
Landon Anderson	3	3	3	3	3	3	3	3	2	2	3	3
Niranjan Poudel	4	3	3	3	3	3	3	3	3	4	3	3
Sam Rands	3	3	3	3	3	3	3	2	3	3	3	3
Scott Greenberg	3	4	4	3	3	3	NA	3	4	3	3	4
Sydney Geisler	1	3	3	3	3	3	3	3	3	3	3	4
Tristan Peterson	2	3	3	3	3	3	3	3	3	3	3	4
Tyler Clayson	3	3	3	3	2	3	3	3	3	3	3	3
Varsha Mandadi	3	2	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
William Raymer	3	3	3	3	3	3	3	3	3	4	3	4
Xiaomeng Zhang	3	2	3	3	3	2	3	3	3	3	3	4
Yuxin Chen	3	3	3	3	3	3	3	3	3	3	3	3

(n) (6 Points) Run the following code chunk and identify unusual student behaviors. The plot shows the timeline of "yes" answers for each of the 12 dates for which we have a Chat files. The horizontal axis shows time (according to the Eastern Time zone). The vertical axis shows the number of each "yes" answer, starting with 1. Each student name is color–coded. This color–coding should be improved for a static version of this graph, but it is OK here for the interactive version.

It will help to enlarge the graph via RStudio Zoom option and then extend it to full screen. Then move your mouse over unusual observations. As a reminder, on 10/22/2020, I asked students at the end to provide a fourth "yes" answer, so this is not unusual here.

Who was late? Who replied early? Is there an unusual systematic pattern for any of the students? List at least 15 interesting cases. Arrange them in chronological order, starting on 09/15/2020. Be specific and indicate the datetime and studentname and describe what makes this an unusual observation.

In case an unusual observation is related to your own name, please explain what caused the unusual observation, e.g., a technical problem, having to leave early, multitasking, etc. There will be no LAP point deduction, but it would be helpful for me to understand what leads to such unusual observations and then decide whether I still will accept similar unusual observations in the future.

```r
library(scales)
library(plotly)

chatText5 %>%
  filter(containsyes) %>%
  ggplot(aes(datetime, 1:length(datetime), color = studentname)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ date, ncol = 2, scales = "free")

ggplotly()

# I received help from these stackoverflow pages:
#
# https://stackoverflow.com/questions/32364018/plotting-posixct-timestamp-series-with-ggplot2/32364171
# https://stackoverflow.com/questions/36389096/ggplot-set-a-time-range-for-facets-plotting-based-on-date
```

Answer:

The following table (Table 4) shows the date, time, student name of the some of the interesting cases from the graph.

Table 4: Table showing some interesting cases.

S.N	Datetime	Student name	Description
1	2020-09-15 14:38:58	Johnathan Diaz	Around 40 seconds late than other students.
2	2020-09-17 14:38:31	Scott Greenberg	More than 3 minutes late than other students but he has yes at correct times also.
3	2020-09-17 14:55:34	Xiaomeng Zhang	Around 50 seconds late than other students.
4	2020-09-22 14:05:12	Sam Rands	Around 4 minutes late than others.
5	2020-09-22 14:35:53	Sam Rands	Aroud 3 minutes early or 13 minutes late as yes in the second LAP is missing.
6	2020-09-24 14:42:27	Johnathan Diaz	Around 7 minutes early than most of the students.
7	2020-09-24 14:48:08	Alyssa Cable	Around 50 seconds earlier than other students.
8	2020-10-01 14:21:03	Alyssa Cable	Around 35 seconds late than other students.
9	2020-10-06 14:03:38	Niranjan Poudel	Around 1 minute later than other: As far as memory serves my computer was not responding properly and made a minute or two late.
10	2020-10-08 14:25:10	Darryl Gatrell	Around 25 second late than others.
11	2020-10-13 14:43:20	Scott Greenberg	Around middle of two yes entries and has missing in previous two.
12	2020-10-13 14:51:56	Sydney Geisler	Around 2 and half minutes earlier than others.
13	2020-10-15 14:16:30	William Raymer	Around 40 seconds early but also has one in the correct time.
14	2020-10-15 14:43:39	Niranjan Poudel	I was not cheating here i was replying to you.
15	2020-10-22 14:12:45	Josh Watteyne	Around a minute late than others.

(o) (4 Points) I never recorded class attendance in any of my previous courses. However, for the web broadcast courses this semester, I made attendance in the regular part of the Zoom lecture discussion meetings mandatory and assigned lecture attendance points (LAPs) for participation. From an instructor's perspective, do LAPs work to ensure that students participate in the Zoom lecture discussion meetings?

It may be helpful to know that we started with 21 students. At the time of the 12^{th} Chat file for this HW, there were 18 active students remaining in this course.

Base your answer on the numerical and graphical data from the question parts above. Be specific when you refer to numbers, studentnames, and dates. If necessary, refer back to an earlier part of this question and mention it by letter (or refer to a specific table). You should write about 1/2 to 1 page to answer this part.

Answer:

First of all, i believe attending lectures can be very helpful to the students. When you have some form of presence in the discussion, even when you are not actively involved in discussion, you can learn or listen something which can help you later on while you are problem solving and you will also get a chance to understand a variety of questions and their solutions as well.

In my view LAPs can definitely play some part in ensuring student's participation in lectures. If we take a look at Table 2 above, we can see that only three students: 1) Varsha Mandadi, 2) Justin Wheeler, 3) Demi Culianos have fewer than 30 LAPs out of 36 possible LAPs. But also as we know three students had left the course by 12th lecture so these might be those three students. If i was an instructor, i would defiently say this was satisfactory attendance from students.

Again taking a look at Table 3 above, we can see most of the students (other than 3 mentioned above) have 3 LAps in most of the lectures, at least two where there is not 3. Some of the other cases are: 1) Syndey Geisler having 1 (yes) and it was on the first lecture of the course on 9/15/2020, 2) Alex Ollerton having 0(yes) responses on 9/15/2020, 3) Kriesten Sohm and Scott Greenberg also having 0 (yes) responses on date 10/06/2020. Also looking at the interactive plot from question (n) above in r-studio i would say there are few cases with major problems. If i was an instructor, in overall i would be satisfied as some errors are bound to happen and some observations will be always skewed from the distributions for human beings.

Further, as we are using internet to connect to the lectures, some of the missing LAPs or misplaced LAPs could be the result of bad internet connection, or some technical issues. So in overall i would suggest that there is a high correlation between the students participation and attendanc and helps ensure students participation to some degree.

(ii) (30 (+2 EC) Points) **IMDb Charts:**
In this question, we revisit the IMDb movie charts from `https://www.imdb.com/chart/top`. Use the R code from `L19_XML_IMDB_part3.R` as a basis for this question.

Always include your R code and show all your results as specified in each question part.

(a) (15 Points) When working with this web page in class, we missed another important piece of information: The names of the movie directors! These names become visible when you move a mouse over the movie title. The name of the director(s) is then listed before the "(dir.)" part, e.g., Frank Darabont for *The Shawshank Redemption*. Revisit the movie chart page and obtain the names of the directors.

Load all R packages you need for this question in this question part. List the name of the first 10 director names you got (for the top-10 listed movies). Note that this is a "live" web page, so names and orders may slightly change from day to day. Reuse as much of the R code from class as possible and work with pipes whenever possible!

Answer:

```r
library(httr)
library(XML)
library(janitor)
```

```r
# Getting the top 10 movie directors
GET("https://www.imdb.com/chart/top") %>%
  htmlParse() %>%
  xpathSApply("//a", xmlGetAttr, "title") %>%
  unlist() %>%
  .[1:10] %>%
  gsub("dir.*", "", .) %>%
  gsub(" (", "", ., fixed = TRUE) %>%
  kbl(col.names = "Top-10 movie directors",
      align = "clc", valign = "t",
      caption = "IMDB top 10 movie directors") %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 5: IMDB top 10 movie directors

Top-10 movie directors
Frank Darabont
Francis Ford Coppola
Francis Ford Coppola
Christopher Nolan
Sidney Lumet
Steven Spielberg
Peter Jackson
Quentin Tarantino
Sergio Leone
Peter Jackson

(b) (5 Points) You likely will notice that Peter Jackson, the director of the *The Lord of the Rings* trilogy, appears three times among the directors of these top-250 movies. What are the other directors that appear three (or more) times among the top-250 movies? List them in decreasing order (with respect to the number of times they are listed).

As for question (i), you can obtain up to **2 extra credit points** if you format your output in this part into a meaningful LaTeX table using the `kable` R function or the `xtable` R package. See question (i) for useful information how to use these.

Answer:

```
# Directors who appear more than once in top 250
GET("https://www.imdb.com/chart/top") %>%
  htmlParse() %>%
  xpathSApply("//a", xmlGetAttr, "title") %>%
  unlist() %>%
  .[1:250] %>%
  gsub("dir.*", "", .) %>%
  gsub(" (", "", ., fixed = TRUE) %>%
  tabyl(var1 = "Director") %>%
  select(1, 2) %>%
  arrange(desc(n)) %>%
  filter(n > 1) %>%
  kbl(col.names = c("Directors", "Number of movies"),
      longtable = TRUE,
      caption = "Number of movies per dircetor in IMDB top 250") %>%
  kable_styling(latex_options = c("striped", "hold_position",
                                  "repeat_header"))
```

Table 6: Number of movies per dircetor in IMDB top 250

Directors	Number of movies
Christopher Nolan	7
Martin Scorsese	7
Stanley Kubrick	7
Akira Kurosawa	6
Alfred Hitchcock	6
Charles Chaplin	6
Steven Spielberg	6
Billy Wilder	5
Hayao Miyazaki	5
Quentin Tarantino	5
Ingmar Bergman	4
Sergio Leone	4
Clint Eastwood	3
David Fincher	3

Table 6: Number of movies per dircetor in IMDB top 250 *(continued)*

Directors	Number of movies
Francis Ford Coppola	3
Frank Capra	3
James Cameron	3
Pete Docter	3
Peter Jackson	3
Ridley Scott	3
Andrei Tarkovsky	2
Andrew Stanton	2
Anthony Russo	2
Bong Joon Ho	2
Chan-wook Park	2
David Lean	2
Denis Villeneuve	2
Frank Darabont	2
Fritz Lang	2
Guy Ritchie	2
James Mangold	2
Joel Coen	2
Lee Unkrich	2
Mel Gibson	2
Milos Forman	2
Peter Weir	2
Richard Linklater	2
Rob Reiner	2
Robert Zemeckis	2
Roman Polanski	2
Ron Howard	2
Sam Mendes	2
Sidney Lumet	2

(c) (10 Points) Using regular expressions and/or string operations, arrange the names of the directors of the top-250 movies as “Last Name, First Name” and **display all of them** (about 150) in an alphabetical order with respect to “Last Name, First Name”. Note that prepositions (sometimes also called nobility particles) such as *de, De, del, Del, de la, De la, Mc, O’, van, Van, von, Von,* etc. are part of the last name and not a middle name. Thus, Florian Henckel von Donnersmarck (the director of “The Lives of Others”) should be sorted as “von Donnersmarck, Florian Henckel” and Brian De Palma (the director of “Scarface”) should be sorted as “De Palma, Brian”. Manually check that some of the “interesting” names appear in the right sorting position.

Hint: Rather than specifying which characters (and symbols) may be part of a last name, it may be easier to specify which characters are not a part of a last name!

Answer:

```
# First creating regex for prepositions
regex1 <-  c(" [Dd]e+$| [Dd]el+$| [Dd]e la+$| Mc+$|")
regex2 <- c( "O'+$| [Vv]an+$| [Vv]on+$")
regex <- paste0(regex1, regex2)

# Arranging the names of the directors
GET("https://www.imdb.com/chart/top") %>%
  htmlParse() %>%
  xpathSApply("//a", xmlGetAttr, "title") %>%
  unlist() %>%
  .[1:250] %>%
  gsub("dir.*", "", .) %>%
  gsub(" (", "", ., fixed = TRUE) %>%
  paste(str_extract(., "\\S+$"), ., sep = ", ") %>%
  gsub(" \\S+$", "", .) %>%
  paste(gsub(" ", "", str_extract(., regex)), .) %>%
  gsub("NA ", "", .) %>%
  gsub(regex, "", .) %>%
  unique() %>%
  sort()

##   [1] "Abrahamson, Lenny"                 "Allers, Roger"
##   [3] "Anderson, Paul Thomas"             "Anderson, Wes"
##   [5] "Aronofsky, Darren"                 "Avildsen, John G."
##   [7] "Benigni, Roberto"                  "Bergman, Ingmar"
##   [9] "Besson, Luc"                       "Boyle, Danny"
##  [11] "Bruckman, Clyde"                   "Cameron, James"
##  [13] "Campanella, Juan JosÃ©"            "Capra, Frank"
##  [15] "Carpenter, John"                   "Ceylan, Nuri Bilge"
##  [17] "Chaplin, Charles"                  "Chazelle, Damien"
##  [19] "Cimino, Michael"                   "Clouzot, Henri-Georges"
##  [21] "Coen, Ethan"                       "Coen, Joel"
##  [23] "Coppola, Francis Ford"             "Curtiz, Michael"
##  [25] "Darabont, Frank"                   "Dassin, Jules"
##  [27] "De Palma, Brian"                   "De Sica, Vittorio"
##  [29] "DeBlois, Dean"                     "del Toro, Guillermo"
##  [31] "Demme, Jonathan"                   "Docter, Pete"
##  [33] "Donen, Stanley"                    "Dreyer, Carl Theodor"
##  [35] "Eastwood, Clint"                   "Elliot, Adam"
##  [37] "Farhadi, Asghar"                   "Farrelly, Peter"
##  [39] "Fincher, David"                    "Fleming, Victor"
##  [41] "Forman, Milos"                     "Gayatri,"
##  [43] "George, Terry"                     "Gibson, Mel"
##  [45] "Gilliam, Terry"                    "Gondry, Michel"
##  [47] "HallstrÃ¶m, Lasse"                 "Hanson, Curtis"
##  [49] "Hill, George Roy"                  "Hirani, Rajkumar"
##  [51] "Hirschbiegel, Oliver"              "Hitchcock, Alfred"
##  [53] "Ho, Bong Joon"                     "Howard, Ron"
##  [55] "Huston, John"                      "IÃ±Ã¡rritu, Alejandro G."
##  [57] "Irmak, Ã‡agan"                     "Jackson, Peter"
##  [59] "Jeunet, Jean-Pierre"               "Jones, Terry"
##  [61] "Kail, Thomas"                      "Kamat, Nishikant"
##  [63] "Kashyap, Anurag"                   "Kassovitz, Mathieu"
##  [65] "Kaye, Tony"                        "Kazan, Elia"
##  [67] "Keaton, Buster"                    "Kershner, Irvin"
##  [69] "Khan, Aamir"                       "Klimov, Elem"
##  [71] "Kobayashi, Masaki"                 "Kramer, Stanley"
##  [73] "Kubrick, Stanley"                  "Kurosawa, Akira"
##  [75] "Labaki, Nadine"                    "Lang, Fritz"
##  [77] "Lasseter, John"                    "Lean, David"
##  [79] "Leone, Sergio"                     "Linklater, Richard"
##  [81] "Lubitsch, Ernst"                   "Lucas, George"
##  [83] "Lumet, Sidney"                     "Lynch, David"
##  [85] "Majidi, Majid"                     "Mangold, James"
##  [87] "Mankiewicz, Joseph L."             "Mann, Michael"
##  [89] "Marquand, Richard"                 "McCarthy, Tom"
##  [91] "McDonagh, Martin"                  "McQueen, Steve"
##  [93] "McTeigue, James"                   "McTiernan, John"
##  [95] "Mehra, Rakeysh Omprakash"          "Meirelles, Fernando"
##  [97] "Mendes, Sam"                       "Miller, George"
##  [99] "Miyazaki, Hayao"                   "Mukherjee, Hrishikesh"
## [101] "Mulligan, Robert"                  "Nakache, Olivier"
## [103] "Nolan, Christopher"                "O'Connor, Gavin"
## [105] "Ozu, YasujirÃ´"                    "Pablos, Sergio"
## [107] "Park, Chan-wook"                   "Paulo, Oriol"
## [109] "Penn, Sean"                        "Persichetti, Bob"
## [111] "Petersen, Wolfgang"                "Phillips, Todd"
## [113] "Polanski, Roman"                   "Pontecorvo, Gillo"
## [115] "Raghavan, Sriram"                  "Reed, Carol"
## [117] "Reiner, Rob"                       "Ritchie, Guy"
## [119] "Rosenberg, Stuart"                 "Russo, Anthony"
## [121] "Sciamma, CÃ©line"                  "Scorsese, Martin"
## [123] "Scott, Ridley"                     "Sheridan, Jim"
## [125] "Shinkai, Makoto"                   "Shyamalan, M. Night"
## [127] "Singer, Bryan"                     "Spielberg, Steven"
## [129] "Stanton, Andrew"                   "Stone, Oliver"
## [131] "Sturges, John"                     "Szifron, DamiÃ¡n"
## [133] "Takahata, Isao"                    "Tarantino, Quentin"
## [135] "Tarkovsky, Andrei"                 "Taylor, Tate"
## [137] "Tiwari, Nitesh"                    "Tornatore, Giuseppe"
## [139] "Truffaut, FranÃ§ois"               "Turgul, Yavuz"
## [141] "Unkrich, Lee"                      "Urushadze, Zaza"
## [143] "Van Sant, Gus"                     "Villeneuve, Denis"
## [145] "Vinterberg, Thomas"                "von Donnersmarck, Florian Henckel"
## [147] "Wachowski, Lana"                   "Weir, Peter"
## [149] "Welles, Orson"                     "Wenders, Wim"
## [151] "Wilder, Billy"                     "Wong, Kar-Wai"
## [153] "Wyler, William"                    "Yates, David"
## [155] "Zemeckis, Robert"
```

General Instructions

(i) Create a single pdf document, using R Markdown, Sweave, or knitr. When you take this course at the 6000–level, you have to use LaTeX in combination with Sweave or knitr. You only have to submit this one document to Canvas.

(ii) Include a title page that contains your name, your A–number, the number of the assignment, the submission date, and any other relevant information.

(iii) Start your answers to each main question on a new page (continuing with the next part of a question on the same page is fine). Clearly label each question and question part. Your answer to question (i) should start on page 2!

(iv) Show your R code and resulting graph(s) [if any] for each question part!

(v) Before you submit your homework, check that you follow all recommendations from Google's R Style Guide (see `http://web.stanford.edu/class/cs109l/unrestricted/resources/google-style.html`). Moreover, make sure that your R code is consistent, i.e., that you use the same type of assignments and the same type of quotes throughout your entire homework.

(vi) Give credit to external sources, such as stackoverflow or help pages. Be specific and include the full URL where you found the help (or from which help page you got the information). Consider R code from such sources as "legacy code or third–party code" that does not have to be adjusted to Google's R Style (even though it would be nice, in particular if you only used a brief code segment).

(vii) **Not following the general instructions outlined above will result in point deductions!**

(viii) For general questions related to this homework, please use the corresponding discussion board in Canvas! I will try to reply as quickly as possible. Moreover, if one of you knows an answer, please post it. It is fine to refer to web pages and R commands, but do not provide the exact R command with all required arguments or which of the suggestions from a stackoverflow web page eventually worked for you! This will be the task for each individual student!

(ix) Submit your single pdf file via Canvas by the submission deadline. Late submissions will result in point deductions as outlined on the syllabus.