Problem Set #6

Nicole Wood (in group NJ with Jade Levandofsky and Jessika Viveros)

November 3, 2023

Overview:

In this problem set, you will be using the **stringr** package (part of tidyverse) to work with strings, and the **lubridate** package for working with dates and times. We will ask you to load Twitter data that is saved as an .Rdata file.

Question 1: Working with strings

1. Load the following packages in the code chunk below: `tidyverse` and `lubridate`.

```
library(tidyverse)
library(lubridate)
```

2. Using `str_c()` and the following objects as input, create the string: `"Roses are red, Violets are blue"`

    - We encourage you to first sketch out what you want to do on some scratch paper.
    - Recall from the lecture example on "Using str_c() on vectors of different lengths", when multiple vectors of different length are provided in the str_c() function, the elements of shorter vectors are recycled. See below.

```
str_c("@", c("ozanj ", "joebruin ", "josiebruin "), sep = "", collapse = ",")

#[1] "@ozanj ,@joebruin ,@josiebruin "
```

- Now try it yourself.

```
vec_1 <- c("Roses", "Violets")
vec_2 <- c("red", "blue")

str_1 <- "are"

# Write your code here
str_c(vec_1, str_1, vec_2, sep = " ", collapse = ", ")
```

```
## [1] "Roses are red, Violets are blue"
```

```
print(str_c)
```

```
## function (..., sep = "", collapse = NULL)
## {
##     check_string(sep)
##     check_string(collapse, allow_null = TRUE)
##     dots <- list(...)
##     dots <- dots[!map_lgl(dots, is.null)]
##     vctrs::vec_size_common(!!!dots)
##     inject(stri_c(!!!dots, sep = sep, collapse = collapse))
## }
## <bytecode: 0x0000026855b44c20>
## <environment: namespace:stringr>
```

3. Pig Latin is a language game in which the first consonant of each word is moved to the end of the word, then `"ay"` is appended to create a suffix. For example, the word `"Wikipedia"` would become `"Ikipediaway"`.

    - Using `str_c()` and `str_sub()`, turn the given `pig_latin` vector into the string: `"igpay atinlay"`
    - We encourage you to first sketch out what you want to do on some scratch paper.
        - First, think about what the final outcome will look like.
        - Then, think about how you can get there. Play around with the `str_sub()` function. What happens when you include different values in the `str_sub()` function?

    - this is low-key the trickiest question in the problem set. So if you get stuck, ask a question to your group or github and move on. and come back to it later.

```
pig_latin <- c('pig', 'latin')

# Write your code here

pl1 <- str_sub(pig_latin, start = 2, end = -1)
pl2 <- str_sub(pig_latin, start = 1, end = 1)

str_c(pl1, pl2, "ay", sep = "", collapse = " ")
```

```
## [1] "igpay atinlay"
```

4. Using `str_c()` and `str_sub()`, decode the given `secret_message`. Your output should be a string.

    - Follow the same logic from above.
    - Sketch out what you want to do on some scratch paper. Break it down step by step. Play around with different values for the `str_sub()` function.

```
secret_message <- c('ollowfay', 'ouryay', 'earthay')

# Write your code here

sm1 <- str_sub(secret_message, start = -3, end = -3)
sm2 <- str_sub(secret_message, start = 1, end = -4)

str_c(sm1, sm2, sep = "", collapse = " ")
```

```
## [1] "follow your heart"
```

```
str_c
```

```
## function (..., sep = "", collapse = NULL)
## {
##     check_string(sep)
##     check_string(collapse, allow_null = TRUE)
##     dots <- list(...)
##     dots <- dots[!map_lgl(dots, is.null)]
##     vctrs::vec_size_common(!!!dots)
##     inject(stri_c(!!!dots, sep = sep, collapse = collapse))
## }
## <bytecode: 0x0000026855b44c20>
## <environment: namespace:stringr>
```

Question 2: Working with Twitter data

1. You will be using Twitter data we fetched from the following Twitter handles: `UniNoticias`, `FoxNews`, and `CNN`.

    - This data has been saved as an Rdata file.
    - Use the `load()` and `url()` functions to download the `news_df` dataframe from the url: `https://github.com/anyone-can-cook/rclass1/raw/master/data/twitter/twitter_news.RData`
    - Report the dimensions of the `news_df` data frame (rows and columns). Use the `dim()` function.

```
load(url("https://github.com/anyone-can-cook/rclass1/raw/master/data/twitter/twitter_news.RData"))

dim(news_df)
```

```
## [1] 1000   90
```

2. Subset your dataframe `news_df` and create a new dataframe called `news_df2` keeping only the following variables: `user_id`, `status_id`, `created_at`, `screen_name`, `text`, `followers_count`, `profile_expanded_url`.

    - Note in the following questions we will ask you to create a new column and that means you have to assign `<-` the new changes you are making to the existing dataframe `news_df2`. Ex. `news_df2 <- news_df %>% mutate(newvar = mean(oldvar))`

```
news_df2 <- news_df %>%
  select(user_id, status_id, created_at, screen_name, text, followers_count, profile_expanded_url)

names(news_df2)
```

```
## [1] "user_id"              "status_id"            "created_at"
## [4] "screen_name"          "text"                 "followers_count"
## [7] "profile_expanded_url"
```

3. Create a new column in `news_df2` called `text_len` that contains the length of the character variable `text`.

- What is the `class` and `type` of this new column? Make sure to include your code in the code chunk below.
  - **ANSWER**: The class and type of text_len are both integer.

```
news_df2 <- news_df2 %>%
  mutate(text_len = str_length(text))

head(news_df2)
```

```
## # A tibble: 6 x 8
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 2 more variables: profile_expanded_url <chr>, text_len <int>
```

```
class(news_df2$text_len)
```

```
## [1] "integer"
```

```
typeof(news_df2$text_len)
```

```
## [1] "integer"
```

4. Create an additional column in `news_df2` called `handle_followers` that stores the twitter handle and the number of followers associated with that twitter handle in a string. For example, the entries in the `handle_followers` column should look like this: `@[twitter_handle] has [number] followers.`

   - What is the `class` and `type` of this new column? Make sure to include your code in the code chunk below.
     - **ANSWER**: The class and type of handle_followers in character.

```
news_df2 <- news_df2 %>%
  mutate(handle_followers = str_c("@", screen_name, " has ", followers_count, " followers."))

head(news_df2)
```

```
## # A tibble: 6 x 9
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 3 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>
```

```
class(news_df2$handle_followers)
```

```
## [1] "character"
```

```
typeof(news_df2$handle_followers)
```

```
## [1] "character"
```

5. Lastly, create a column in `news_df2` called `short_web` that contains a short version of the `profile_expanded_url` without the `http://www.` part of the url. For example, the entries in that column should look something like this: `nytimes.com`.

```
news_df2 <- news_df2 %>%
  mutate(short_web = str_sub(profile_expanded_url, start = 12, end = -1))

head(news_df2)
```

```
## # A tibble: 6 x 10
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 4 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>
```

```
tail(news_df2)
```

```
## # A tibble: 6 x 10
##   user_id status_id        created_at          screen_name text  followers_count
##   <chr>   <chr>            <dttm>              <chr>       <chr>           <int>
## 1 759251  132364613709459~ 2020-11-03 15:20:05 CNN         "The~        50562050
## 2 759251  132349866149036~ 2020-11-03 05:34:04 CNN         "Joe~        50562050
## 3 759251  132344483513577~ 2020-11-03 02:00:11 CNN         "Aft~        50562050
## 4 759251  132387426967223~ 2020-11-04 06:26:36 CNN         "CNN~        50562050
## 5 759251  132344096087807~ 2020-11-03 01:44:48 CNN         "Pre~        50562050
## 6 759251  132349036344577~ 2020-11-03 05:01:06 CNN         "\"I~        50562050
## # i 4 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>
```

Question 3: Working with dates/times

1. Using the column `created_at`, create a new column in `news_df2` called `dt_chr` that is a character version of `created_at`.

   - What is the `class` of the `created_at` and `dt_chr` columns? Make sure to include your code in the code chunk below.

- **ANSWER**: The class of the created_at column is POSIXct and POSIXt. The class of the dt_chr column is character.

```
news_df2 <- news_df2 %>%
  mutate(dt_chr = as.character(created_at))

head(news_df2)
```

```
## # A tibble: 6 x 11
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 5 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>
```

```
class(news_df2$created_at)
```

```
## [1] "POSIXct" "POSIXt"
```

```
class(news_df2$dt_chr)
```

```
## [1] "character"
```

2. Create another column in `news_df2` called `dt_len` that stores the length of `dt_chr`.

```
news_df2 <- news_df2 %>%
  mutate(dt_len = str_length(dt_chr))

head(news_df2)
```

```
## # A tibble: 6 x 12
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 6 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>, dt_len <int>
```

3. Next, create additional columns in `news_df2` for each of the following date/time components:

   a. Create a new column `date_chr` for date (e.g. `2020-03-26`) using the column `dt_chr` and the `str_sub()` function.

b. Do the same for year `yr_chr` (e.g. 2020).
c. Do the same for month `mth_chr` (e.g. 03).
d. Do the same for day `day_chr` (e.g. 26).
e. Do the same for time `time_chr` (e.g. 22:41:09).

```
news_df2 <- news_df2 %>%
  mutate(date_chr = str_sub(dt_chr, start = 1, end = 10),
         yr_chr = str_sub(dt_chr, start = 1, end = 4),
         mth_chr = str_sub(dt_chr, start = 6, end = 7),
         day_chr = str_sub(dt_chr, start = 9, end = 10),
         time_chr = str_sub(dt_chr, start = 12, end = -1))

head(news_df2)
```

```
## # A tibble: 6 x 17
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 11 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>, dt_len <int>,
## #   date_chr <chr>, yr_chr <chr>, mth_chr <chr>, day_chr <chr>, time_chr <chr>
```

4. Using the column we created in the previous question `time_chr`, create additional columns in `news_df2` for the following time components:

   a. Create a new column `hr_chr` for hour (e.g. 22) using the column `time_chr` and the `str_sub()` function.
   b. Do the same for minutes `min_chr` (e.g. 41).
   c. Do the same for seconds `sec_chr` (e.g. 09).

```
news_df2 <- news_df2 %>%
  mutate(hr_chr = str_sub(time_chr, start = 1, end = 2),
         min_chr = str_sub(time_chr, start = 4, end = 5),
         sec_chr = str_sub(time_chr, start = -2, end = -1))

head(news_df2)
```

```
## # A tibble: 6 x 20
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 14 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>, dt_len <int>,
```

```
## #   date_chr <chr>, yr_chr <chr>, mth_chr <chr>, day_chr <chr>, time_chr <chr>,
## #   hr_chr <chr>, min_chr <chr>, sec_chr <chr>
```

5. Now let's get some practice with the `lubridate` package.

    a. Using the `year()` function from the `lubridate` package, create a new column in `news_df2` called `yr_num` that contains the year (e.g. 2020) extracted from `date_chr`.
    b. Do the same for month `mth_num`.
    c. Do the same for day `day_num`.
    d. Do the same for hour `hr_num`, but extract from `created_at` column instead of `date_chr`.
    e. Do the same for minutes `min_num`.
    f. Do the same for seconds `sec_num`.

```
news_df2 <- news_df2 %>%
  mutate(yr_num = year(date_chr),
         mth_num = month(date_chr),
         day_num = day(date_chr),
         hr_num = hour(created_at),
         min_num = minute(created_at),
         sec_num = second(created_at))

head(news_df2)
```

```
## # A tibble: 6 x 26
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 20 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>, dt_len <int>,
## #   date_chr <chr>, yr_chr <chr>, mth_chr <chr>, day_chr <chr>, time_chr <chr>,
## #   hr_chr <chr>, min_chr <chr>, sec_chr <chr>, yr_num <dbl>, mth_num <dbl>,
## #   day_num <int>, hr_num <int>, min_num <int>, sec_num <dbl>
```

6. Using the **new numeric columns** (e.g. day_num, mth_num) you've created in the previous step, reconstruct the date and datetime columns. Namely, add the following columns to `news_df2`:

    a. Use `make_date()` to create new column called `my_date` that contains the date (year, month, day).
    b. Use `make_datetime()` to create new column called `my_datetime` that contains the datetime (year, month, day, hour, minutes, seconds).

    - What is the `class` of your `my_date` and `my_datetime` columns? Make sure to include your code in the code chunk below.
        - **ANSWER**:

```
news_df2 <- news_df2 %>%
  mutate(my_date = make_date(year = yr_chr, month = mth_chr, day = day_chr),
         my_datetime = make_datetime(yr_num, mth_num, day_num, hr_num, min_num, sec_num))

head(news_df2)
```

```
## # A tibble: 6 x 28
##   user_id  status_id       created_at          screen_name text  followers_count
##   <chr>    <chr>           <dttm>              <chr>       <chr>           <int>
## 1 35785401 13244149277722~ 2020-11-05 18:14:59 UniNoticias "¿Cu~         2075729
## 2 35785401 13244143417189~ 2020-11-05 18:12:40 UniNoticias "\U0~         2075729
## 3 35785401 13244133763285~ 2020-11-05 18:08:49 UniNoticias "Nev~         2075729
## 4 35785401 13244124183453~ 2020-11-05 18:05:01 UniNoticias "Com~         2075729
## 5 35785401 13244120855483~ 2020-11-05 18:03:42 UniNoticias "Joe~         2075729
## 6 35785401 13244107064066~ 2020-11-05 17:58:13 UniNoticias "\U0~         2075729
## # i 22 more variables: profile_expanded_url <chr>, text_len <int>,
## #   handle_followers <chr>, short_web <chr>, dt_chr <chr>, dt_len <int>,
## #   date_chr <chr>, yr_chr <chr>, mth_chr <chr>, day_chr <chr>, time_chr <chr>,
## #   hr_chr <chr>, min_chr <chr>, sec_chr <chr>, yr_num <dbl>, mth_num <dbl>,
## #   day_num <int>, hr_num <int>, min_num <int>, sec_num <dbl>, my_date <date>,
## #   my_datetime <dttm>
```

Create a GitHub issue

- Go to the class repository and create a new issue.
- Refer to rclass1 student issues readme for instructions on how to post questions or reflections.
- You are also required to respond to at least one issue posted by another student.
- Paste the url to your issue here: https://github.com/anyone-can-cook/rclass1_student_issues_f23/issues/723
- Paste the url to the issue you responded to here: https://github.com/anyone-can-cook/rclass1_student_issues_f23/issues/722

Knit to pdf and submit problem set

Knit to pdf by clicking the "Knit" button near the top of your RStudio window (icon with blue yarn ball) or drop down and select "Knit to PDF"

- Go to the class website and under the "Readings & Assignments" » "Week 6" tab, click on the "Problem set 6 submission link"
- Submit both .Rmd and pdf files
- Use this naming convention "lastname_firstname_ps#" for your .Rmd and pdf files (e.g. jaquette_ozan_ps6.Rmd & jaquette_ozan_ps6.pdf)