Homework 2 Part 1

AUTHOR
Thomas Zwiller

There are 1000 observations, spread across 10 csv files in a zip.

First you will need to use `unzip` to unzip it. (I manually unzipped it)

Next, you will need to generate a vector of all of those files with `list.files`.

```
files_list <- list.files("/Users/TomTheIntern/Desktop/Mendoza/Mod 1/Wrangling/Homework 2/
                        all.files = TRUE, full.names = TRUE)
```

You can use a `for loop`, an `lapply`, or a `purrr::map` statement to bring all of those files together into one data frame. The columns are in the same order, so feel free to bind them together. If you end up with a list of data frames, you can use `do.call(rbind, your_object)`, `dplyr::bind_rows`, or `data.table::rbindlist` to bring them together.

```
files_frame <- lapply(files_list, FUN = read.csv)

raw_data <-  do.call(rbind, files_frame)
```

#This part should live in a separate qmd file.

#1. Create a function to clean ratings & length variables.

```
#Behold! My first R function, Time Aggregator! His legal first name is Gator, his friends

#we take four things: the column you want this applied to, what you want to replace and w
time_aggregator <- function(colnameandtype, thingtoreplace, whattoreplace, numeric) {
  #if numeric is 1, it is replaced and then converted
    if(numeric == 1){
      #is replaced
      colnameandtype <- gsub(thingtoreplace, whattoreplace, as.character(colnameandtype))
      #and is then converted to a numeric
      colnameandtype <- as.numeric(colnameandtype)
        return(colnameandtype)
    }
  #if numeric is 0, it is returned as a character
  else if(numeric == 0){
    colnameandtype <- gsub(thingtoreplace, whattoreplace, as.character(colnameandtype))
  #This might seem weird since we want to convert everything into a numeric value but it
    return(colnameandtype)
  }
}

#and here is where I used Chomp to manipulate the data, getting rid of hours and minutes,
raw_data$length_hours <-  time_aggregator(raw_data$length_hours, "hrs", "", 1)
raw_data$length_minutes <- time_aggregator(raw_data$length_minutes, "mins","", 1)
raw_data$rating_first_watch <- time_aggregator(raw_data$rating_first_watch, " stars", "",
raw_data$rating_second_watch <- time_aggregator(raw_data$rating_second_watch, " stars", "
```

#2. Create a total length (as in movie length) column.

```
#This part is not very complicated. Thanks to Chomp making both of these categories numer
#I multiplied hours by 60 and added the minutes on to get a run_time in minutes
raw_data$length_combined <- raw_data$length_hours * 60 + raw_data$length_minutes
```

#3. Create a date delta between release time and review time.

```
#imported lubridate
library(lubridate)
```

```
Warning: package 'lubridate' was built under R version 4.4.1


Attaching package: 'lubridate'

The following objects are masked from 'package:base':

    date, intersect, setdiff, union
```

```
#I used Chomp to get rid of the / in the dates and replaced them with -. I was originally
raw_data$review_date <- time_aggregator(raw_data$review_date, "/", "-", 0)

#used lubridate to convert the dates using the ymd format
raw_data$review_date <- ymd(raw_data$review_date)
#used lubridate to convert the dates using the mdy format
raw_data$release_date <- mdy(raw_data$release_date)
#and then created a delta in days
#https://campus.datacamp.com/courses/intermediate-r-for-finance/dates?ex=8
raw_data$date_delta <- difftime(raw_data$review_date, raw_data$release_date, units = "wee
```

#4. Pick a word from the reviews and count how many times that word appears.

```
#I imported the stringr library so I could use string detect
library(stringr)
#And then I string count to count how many times the word film appeared
#https://stringr.tidyverse.org/reference/str_count.html
raw_data$disappointing_count <- str_count(raw_data$review_text, "didn't")
```

#5. Create an aggregated data set for the unique movies. There should be movies, average ratings, release year, genre, and total number of reviews.

```
#I first aggregated the data here
review_count <- aggregate(raw_data$total_reviews ~ raw_data$title, FUN = sum)
```

```
average_rating_1 <- aggregate(raw_data$rating_first_watch ~ raw_data$title, FUN = mean)
average_rating_2 <- aggregate(raw_data$rating_second_watch ~ raw_data$title, FUN = mean)
movie_date <- aggregate(raw_data$release_date ~ raw_data$title, FUN = min)
movie_genre <- aggregate(raw_data$genre ~ raw_data$title, FUN = unique)

#and then just used a series of merge statements to collect the data since they all had t
merged_ratings <- merge(average_rating_1,
                        average_rating_2,
                        by = 'raw_data$title')

year_review <- merge(merged_ratings,
                    movie_date,
                    by = 'raw_data$title')

review_genre <- merge(year_review,
                    movie_genre,
                    by = 'raw_data$title')

final_product <- merge(review_genre,
                      review_count,
                      by = 'raw_data$title'
                      )

#https://sqlpad.io/tutorial/rename-columns-dataframes/#:~:text=examples%20and%20explanati
names(final_product) <- c('Movie Title',
                          'First Watch Rating',
                          'Second Watch Rating',
                          'Release Date',
                          'Genre',
                          'Total Reviews')
```

#6. You should have two data frames, so `save` those objects for the next step.

```
#saving out my data
save(final_product, file = "final_movie_product.rda")
save(raw_data, file = "raw_movie_data.rda")
```