Assignment 10: Data Scraping

Julia Kagiliery

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on data scraping.

Directions

1. Rename this file `<FirstLast>_A10_DataScraping.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure your code is tidy; use line breaks to ensure your code fits in the knitted output.
5. Be sure to **answer the questions** in this assignment document.
6. When you have completed the assignment, **Knit** the text and code into a single PDF file.

Set up

1. Set up your session:

- Load the packages `tidyverse`, `rvest`, and any others you end up using.
- Check your working directory

```r
#1
library(tidyverse)
library(rvest)
library(dplyr)
library(lubridate)

getwd()
```

```
## [1] "/Users/juliakagiliery/Library/Mobile Documents/com~apple~CloudDocs/GitHub Links/EDAClas2025"
```

2. We will be scraping data from the NC DEQs Local Water Supply Planning website, specifically the Durham's 2024 Municipal Local Water Supply Plan (LWSP):

- Navigate to https://www.ncwater.org/WUDC/app/LWSP/search.php
- Scroll down and select the LWSP link next to Durham Municipality.
- Note the web address: https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=03-32-010&year=2024

Indicate this website as the as the URL to be scraped. (In other words, read the contents into an `rvest` webpage object.)

```r
#2
webpage <- read_html('https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=03-32-010&year=2024')
```

3. The data we want to collect are listed below:

- From the “1. System Information” section:
- Water system name
- PWSID
- Ownership
- From the “3. Water Supply Sources” section:
- Maximum Day Use (MGD) - for each month

In the code chunk below scrape these values, assigning them to four separate variables.

> HINT: The first value should be “Durham”, the second “03-32-010”, the third “Municipality”, and the last should be a vector of 12 numeric values (represented as strings)“.

```r
#3
water_system_name <- webpage |>
  html_nodes("div+ table tr:nth-child(1) td:nth-child(2)") |>
  html_text()
water_system_name
```

```
## [1] "Durham"
```

```r
pwsid <- webpage |>
  html_nodes("td tr:nth-child(1) td:nth-child(5)") |>
  html_text()
pwsid
```

```
## [1] "03-32-010"
```

```r
ownership <- webpage |>
  html_nodes("div+ table tr:nth-child(2) td:nth-child(4)") |>
  html_text()
ownership
```

```
## [1] "Municipality"
```

```r
max_day_use_mgd <- webpage |>
  html_nodes("th~ td+ td") |>
  html_text()
max_day_use_mgd
```

```
##  [1] "34.5000" "36.0600" "37.3300" "32.1000" "46.6500" "37.3600" "38.2000"
##  [8] "41.9000" "36.5800" "36.7300" "42.9600" "34.4500"
```

4. Convert your scraped data into a dataframe. This dataframe should have a column for each of the 4 variables scraped and a row for the month corresponding to the withdrawal data. Also add a Date column that includes your month and year in data format. (Feel free to add a Year column too, if you wish.)

   TIP: Use `rep()` to repeat a value when creating a dataframe.

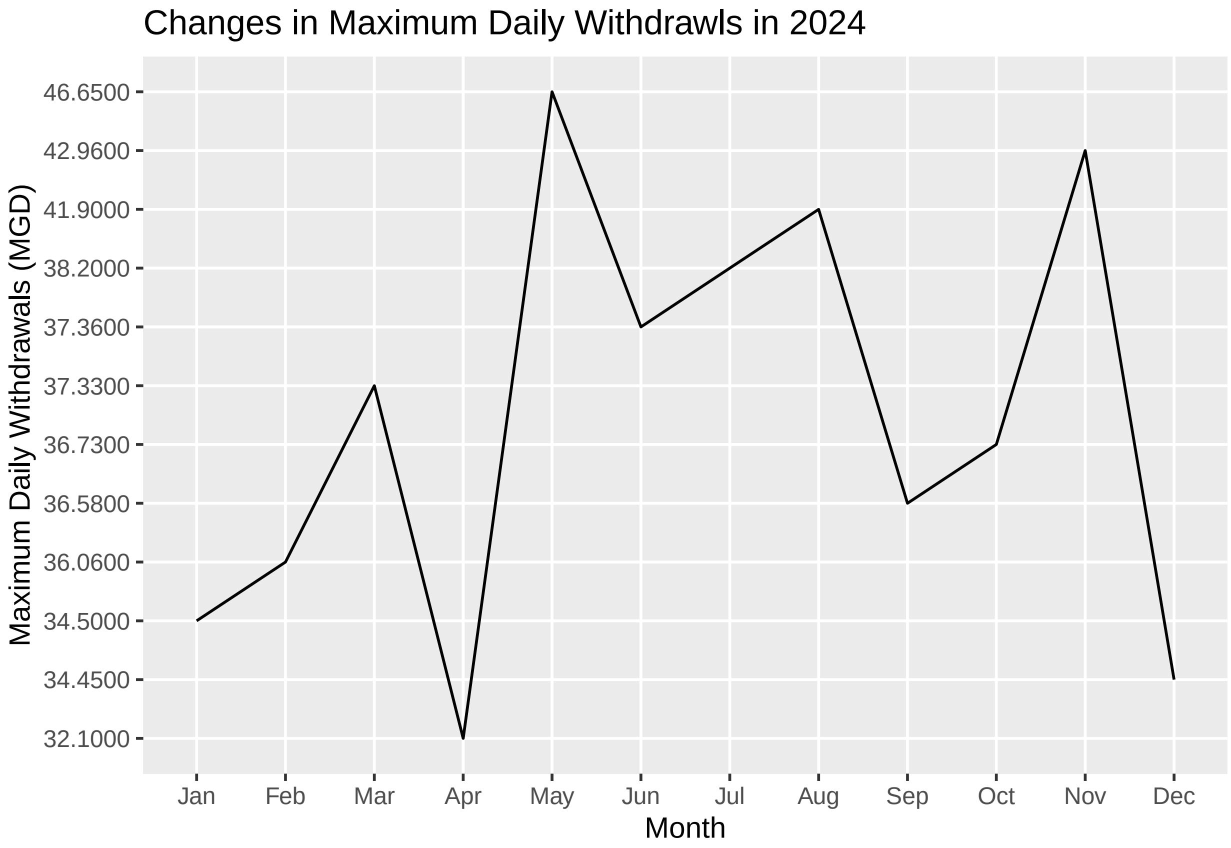
   NOTE: It's likely you won't be able to scrape the monthly widthrawal data in chronological order. You can overcome this by creating a month column manually assigning values in the order the data are scraped: "Jan", "May", "Sept", "Feb", etc... Or, you could scrape month values from the web page...

5. Create a line plot of the maximum daily withdrawals across the months for 2024, making sure, the months are presented in proper sequence.

```
#4
NcPowerData <- data.frame(
  "Month" = 1:12,
  "Year" = rep(2024, 12),  # Ensure it's numeric
  "Date" = as.Date(paste(2024, 1:12, 1, sep = "-")),
  "Water System" = rep("Durham", 12),
  "Ownership" = rep("Municipality", 12),
  "PWSID" = rep("03-32-010", 12),
  "Maximum Daily Withdrawals" = as.numeric(max_day_use_mgd)
)


print(NcPowerData)
```

```
##    Month Year       Date Water.System    Ownership     PWSID
## 1      1 2024 2024-01-01       Durham Municipality 03-32-010
## 2      2 2024 2024-02-01       Durham Municipality 03-32-010
## 3      3 2024 2024-03-01       Durham Municipality 03-32-010
## 4      4 2024 2024-04-01       Durham Municipality 03-32-010
## 5      5 2024 2024-05-01       Durham Municipality 03-32-010
## 6      6 2024 2024-06-01       Durham Municipality 03-32-010
## 7      7 2024 2024-07-01       Durham Municipality 03-32-010
## 8      8 2024 2024-08-01       Durham Municipality 03-32-010
## 9      9 2024 2024-09-01       Durham Municipality 03-32-010
## 10    10 2024 2024-10-01       Durham Municipality 03-32-010
## 11    11 2024 2024-11-01       Durham Municipality 03-32-010
## 12    12 2024 2024-12-01       Durham Municipality 03-32-010
##    Maximum.Daily.Withdrawals
## 1                      34.50
## 2                      36.06
## 3                      37.33
## 4                      32.10
## 5                      46.65
## 6                      37.36
## 7                      38.20
## 8                      41.90
## 9                      36.58
## 10                     36.73
## 11                          42.96
## 12                          34.45
```

```
#5
NcPowerData |>
  ggplot(aes(x = factor(Month, levels = 1:12, labels = month.abb),
             y = max_day_use_mgd, group = 1)) +
  geom_line() +
  labs(x="Month", y="Maximum Daily Withdrawals (MGD)",
title="Changes in Maximum Daily Withdrawls in 2024")
```



6. Note that the PWSID and the year appear in the web address for the page we scraped. Construct a function with two input - "PWSID" and "year" - that:

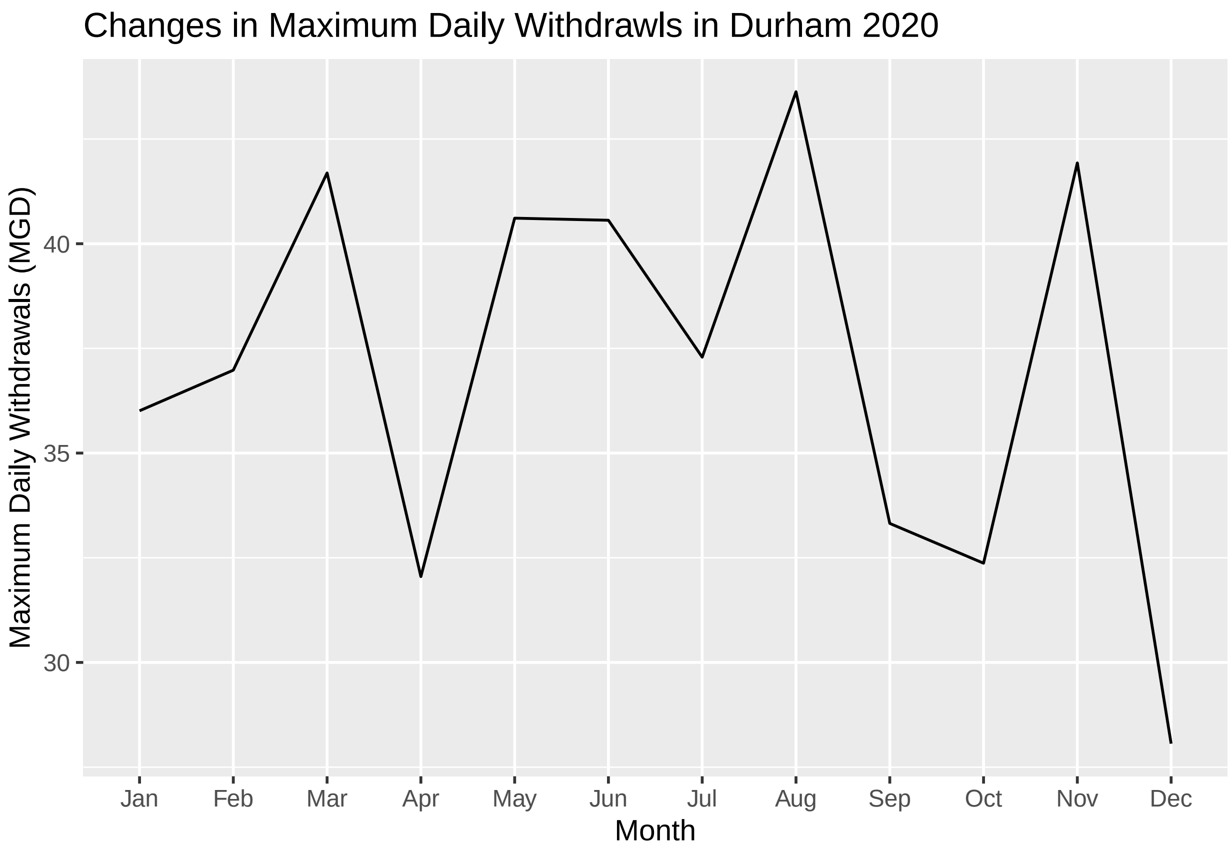
- Creates a URL pointing to the LWSP for that PWSID for the given year
- Creates a website object and scrapes the data from that object (just as you did above)
- Constructs a dataframe from the scraped data, mostly as you did above, but includes the PWSID and year provided as function inputs in the dataframe.
- Returns the dataframe as the function's output

```
#6.

PWSID_tag <- 'td tr:nth-child(1) td:nth-child(5)'
water_system_tag <- 'div+ table tr:nth-child(1) td:nth-child(2)'
maximum_daily_use_tag <- 'th~ td+ td'
```

```r
ownership_tag <- 'div+ table tr:nth-child(2) td:nth-child(4)'

base_url <- 'https://www.ncwater.org/WUDC/app/LWSP/report.php?'
PWSID <- '03-32-010'
the_year <- 2024

scrape_url <- paste0('https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=',
                     PWSID, '&year=', the_year)

scrapewebsite <- read_html(scrape_url)

PWSID <- webpage |>
  html_nodes(PWSID_tag) |>
  html_text()

WaterSystem <- webpage |>
  html_nodes(water_system_tag) |>
  html_text()

MaxDayUse <- webpage |>
  html_nodes(maximum_daily_use_tag) |>
  html_text()

Ownership <- webpage |>
  html_nodes(ownership_tag) |>
  html_text()

scrape_it <- function(the_year, PWSID) {
  website <- read_html(paste0('https://www.ncwater.org/WUDC/app/LWSP/report.php?pwsid=', PWSID, '&year=

  # Extract data
  PWSID_value <- website |> html_nodes(PWSID_tag) |> html_text()
  WaterSystem_value <- website |> html_nodes(water_system_tag) |> html_text()
  MaxDayUse_value <- website |> html_nodes(maximum_daily_use_tag) |> html_text() |> as.numeric()
  Ownership_value <- website |> html_nodes(ownership_tag) |> html_text() |> trimws()

  # Ensure MaxDayUse_value has 12 elements (one per month)
  if (length(MaxDayUse_value) != 12) {
    stop("Error: Maximum daily use data does not contain 12 values.")
  }

  # Construct the dataframe
  dataframe_LWSP <- data.frame(
    "Month" = 1:12,
    "Year" = rep(the_year, 12),
    "Max Daily Use" = MaxDayUse_value,
    "Ownership" = rep(Ownership_value, 12),
    "PWSID" = rep(PWSID_value, 12),
    "Water System" = rep(WaterSystem_value, 12),
    "Date" = as.Date(paste(the_year, 1:12, 1, sep = "-"))
  )

  return(dataframe_LWSP)
}
```

7. Use the function above to extract and plot max daily withdrawals for Durham (PWSID='03-32-010') for each month in 2020

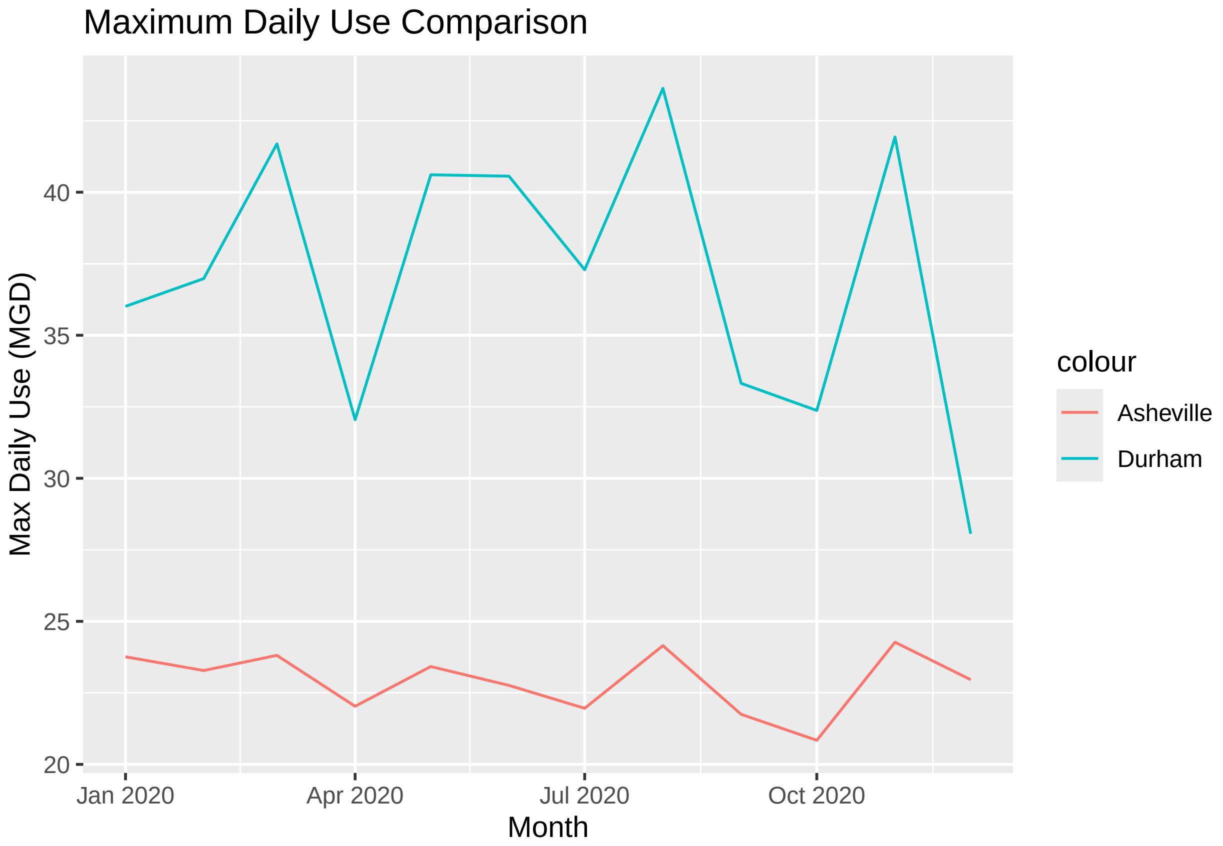
```
#7
Durham2020Data <- scrape_it(2020, '03-32-010')

Durham2020Data |>
  ggplot(aes(x = factor(Month, levels = 1:12, labels = month.abb),
             y = Max.Daily.Use, group = 1)) +
  geom_line() +
  labs(x="Month", y="Maximum Daily Withdrawals (MGD)",
title="Changes in Maximum Daily Withdrawls in Durham 2020")
```



8. Use the function above to extract data for Asheville (PWSID = '01-11-010') in 2020. Combine this data with the Durham data collected above and create a plot that compares Asheville's to Durham's water withdrawals.

```
#8
Ashville2020Data <- scrape_it(2020, '01-11-010')

DUAS202Data <- left_join(Durham2020Data, Ashville2020Data, by = "Date")
```

```
DUAS202Data |>
  ggplot() +
geom_line(aes(x=Date, y=Max.Daily.Use.x, color="Durham")) +
geom_line(aes(x=Date, y=Max.Daily.Use.y, color="Asheville")) +
labs(title="Maximum Daily Use Comparison", x="Month", y="Max Daily Use (MGD)")
```

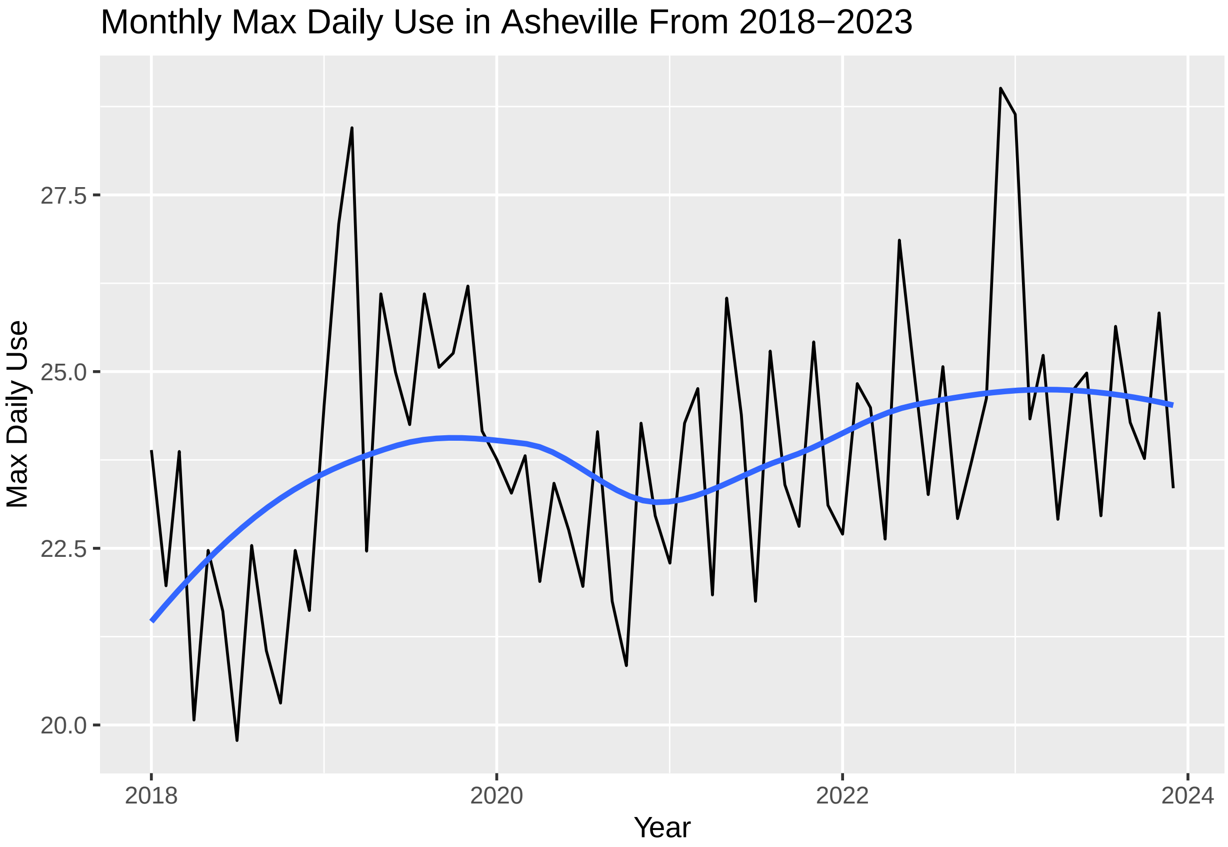


9. Use the code & function you created above to plot Asheville's max daily withdrawal by months for the years 2018 thru 2023.Add a smoothed line to the plot (method = 'loess').

   TIP: See Section 3.2 in the "10_Data_Scraping.Rmd" where we apply "map2()" to iteratively run a function over two inputs. Pipe the output of the map2() function to `bindrows()` to combine the dataframes into a single one, and use that to construct your plot.

```
#9
years <- rep(2018:2023)

Asheville <- lapply(X = years, FUN = scrape_it, PWSID = '01-11-010')
Asheville <- bind_rows(Asheville)

Asheville |>
  ggplot(aes(x = Date, y = Max.Daily.Use)) +
  geom_line() +
  geom_smooth(method = "loess", se=FALSE) +
  labs(
    title = "Monthly Max Daily Use in Asheville From 2018-2023",
    x = "Year",
    y = "Max Daily Use"
  )
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Question: Just by looking at the plot (i.e. not running statistics), does Asheville have a trend in water usage over time? > Answer: Just by looking at the graph, it does appear, based on the trend line, that Asheville does have an increasing trend in use over time. >