Assignment 7

Feiyu Zheng (fz114)

2022/3/22

Problem 1

1. From the worldometer webpage https://www.worldometers.info/coronavirus/, extract the country-wise COVID data. Treat it as a static HTML webpage. Create a tibble showing the country-wise data for the following four variables only: total cases, new cases, total deaths and new deaths. Clean the data to make them amenable to analysis.

Download the table from worldometer webpage

```
url <- "https://www.worldometers.info/coronavirus"
country_covid_table <- url %>%
  read_html() %>%
  html_nodes("table") %>%
  html_table(fill = TRUE) %>%
  .[[1]]
country_covid_table
```

```
## # A tibble: 243 x 22
##      `#` `Country,Other` TotalCases  NewCases   TotalDeaths NewDeaths
##    <int> <chr>           <chr>       <chr>      <chr>           <int>
##  1    NA "North America" 96,354,898  "+919"     1,438,290          15
##  2    NA "Asia"          137,340,569 "+256,981" 1,398,120         438
##  3    NA "South America" 55,979,556  "+80"      1,286,260          NA
##  4    NA "Europe"        175,606,618 ""         1,764,530          NA
##  5    NA "Oceania"       5,204,072   "+59,899"  8,719              14
##  6    NA "Africa"        11,691,492  ""         252,441            NA
##  7    NA ""              721         ""         15                 NA
##  8    NA "World"         482,177,926 "+317,879" 6,148,375         467
##  9     1 "USA"           81,621,888  ""         1,003,467          NA
## 10     2 "India"         43,020,723  ""         521,066            NA
## # ... with 233 more rows, and 16 more variables: TotalRecovered <chr>,
## #   NewRecovered <chr>, ActiveCases <chr>, `Serious,Critical` <chr>,
## #   `Tot<U+00A0>Cases/1M pop` <chr>, `Deaths/1M pop` <chr>, TotalTests <chr>,
## #   `Tests/1M pop` <chr>, Population <chr>, Continent <chr>,
## #   `1 Caseevery X ppl` <chr>, `1 Deathevery X ppl` <chr>,
## #   `1 Testevery X ppl` <int>, `New Cases/1M pop` <chr>,
## #   `New Deaths/1M pop` <dbl>, `Active Cases/1M pop` <chr>
```

Clean the data in the table

```r
country_covid_table_cleaned <- country_covid_table %>%
  filter(!is.na(`#`)) %>% # filter out rows that are not country
  select(`Country,Other`, TotalCases, NewCases, TotalDeaths, NewDeaths) %>% # choose columns
  rename(Country = `Country,Other`) %>%
  mutate_at(c("TotalCases", "NewCases", "TotalDeaths", "NewDeaths"), ~as.integer(str_replace_all(., "[,
country_covid_table_cleaned
```

```
## # A tibble: 227 x 5
##    Country  TotalCases NewCases TotalDeaths NewDeaths
##    <chr>         <int>    <int>       <int>     <int>
##  1 USA        81621888       NA     1003467        NA
##  2 India      43020723       NA      521066        NA
##  3 Brazil     29842418       NA      658926        NA
##  4 France     25029573       NA      141672        NA
##  5 UK         20691123       NA      164454        NA
##  6 Germany    20251037       NA      128947        NA
##  7 Russia     17762742       NA      367351        NA
##  8 Turkey     14800677       NA       97800        NA
##  9 Italy      14364723       NA      158782        NA
## 10 S. Korea   12003054   187213       15186       287
## # ... with 217 more rows
```

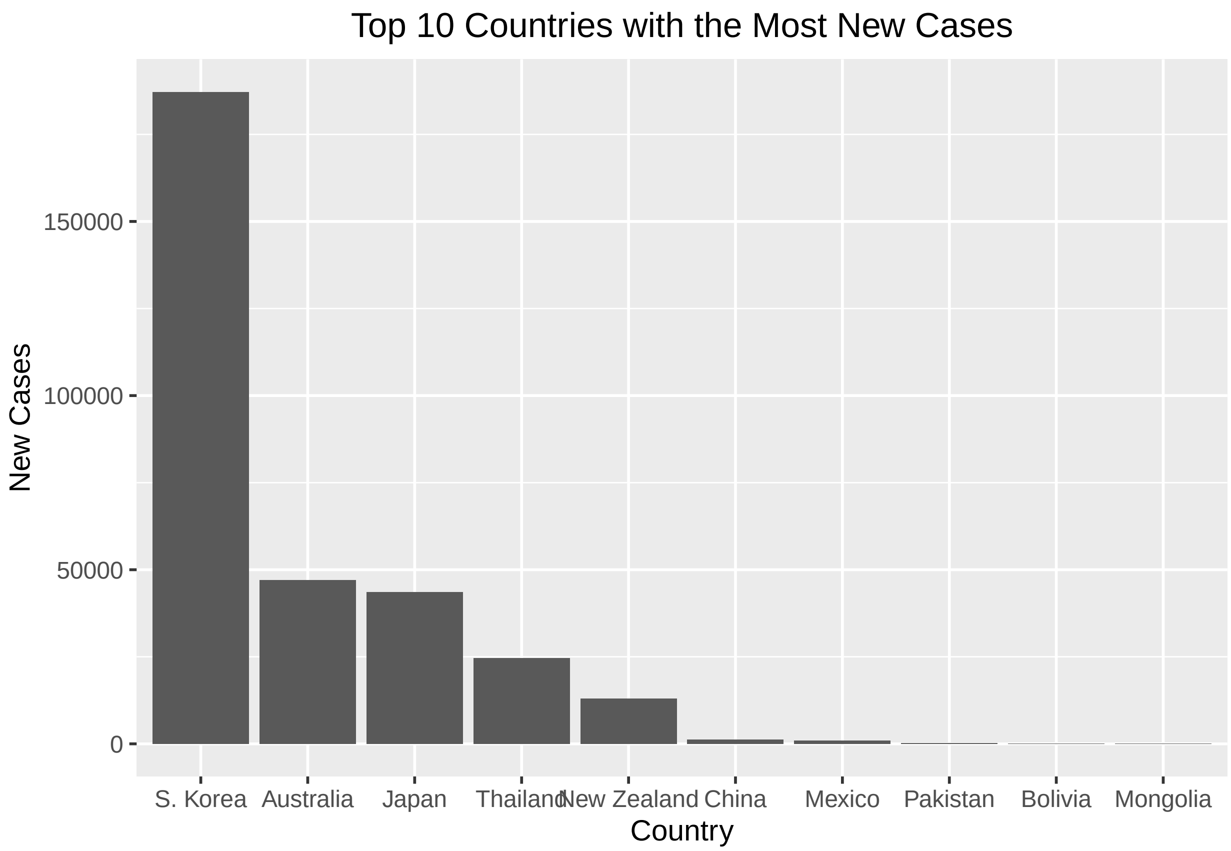
2. Identify the top ten countries reporting most new cases on the day you are analyzing the data.

```r
top10NewCases <- country_covid_table_cleaned %>%
  arrange(desc(NewCases)) %>% # sort in descending order of NewCases
  head(10) # top 10
top10NewCases
```

```
## # A tibble: 10 x 5
##    Country     TotalCases NewCases TotalDeaths NewDeaths
##    <chr>            <int>    <int>       <int>     <int>
##  1 S. Korea      12003054   187213       15186       287
##  2 Australia      4326294    46957        5897         6
##  3 Japan          6377719    43565       27767        68
##  4 Thailand       3553720    24635       24883        81
##  5 New Zealand     610687    12942         231         8
##  6 China           144515     1275        4638        NA
##  7 Mexico         5650896      919      322750        15
##  8 Pakistan       1524086      186       30346         1
##  9 Bolivia         901367       75       21487        NA
## 10 Mongolia        468051       69        2177        NA
```

3. For these ten countries, generate a bar plot showing the number of new cases, arranged in order of magnitude.

```r
top10NewCases %>%
  ggplot(aes(reorder(Country, -NewCases), NewCases)) +
  geom_bar(stat = "identity") +
  scale_y_continuous(labels = function(x) format(x, scientific = F)) +
  labs(
    x = "Country",
    y = "New Cases",
    title = "Top 10 Countries with the Most New Cases") +
  theme(plot.title = element_text(hjust = 0.5))
```



Problem 2

1. Obtain your free API for https://spoonacular.com/food-api

```
api_key <- "40d925facf66429f9d837537060df81d"
print(paste("API Key: ", api_key, sep = ""))
```

```
## [1] "API Key: 40d925facf66429f9d837537060df81d"
```

2. Use it to find out all Italian recipes available in the website that have carbohydrates not exceeding 30 grams. How many such recipes are there? Find the top ten having the lowest carbs. Present your output as a 10x3 tibble, where the column names are "Recipe" (the title of the recipe), "ID" (the ID of the recipe), and "Carbs" (the carb content).

Make API Request

```
api_url <- paste(
  "https://api.spoonacular.com/recipes/complexSearch",
  "?apiKey=", api_key,
  sep = ""
)

url <- paste(
  api_url,
  "&cuisine=", "italian",
  "&includeNutrition=", "true",
  "&maxCarbs=", "30",
  "&sort=", "carbs",
  "&sortDirection=", "asc",
  sep = ""
)

# using api request to get json data
json_result <- url %>%
  curl() %>%
  readLines(warn=F)
```

Convert to 10x3 tibble

```
# convert json data to 10x3 tibble
result_t <- json_result %>%
  fromJSON() %>%
  .[[1]] %>%
  as_tibble() %>%
  mutate(
    Recipe = title,
    ID = id,
    Carbs = bind_rows(.$nutrition$nutrients)$amount) %>%
  select(Recipe, ID, Carbs)
result_t
```

```
## # A tibble: 10 x 3
##    Recipe                                           ID Carbs
##    <chr>                                         <int> <dbl>
##  1 Tiramisu Overnight Oats                     1697783  1.80
##  2 Mini eggplant pizza                          651956  2.11
##  3 Baked Ziti with Ricotta and Italian Sausage  1697599  2.78
##  4 Cast Iron Shrimp Pizza with Pecan Basil Pesto 1697557  2.81
##  5 Italian Caprese Sliders                      648084  2.90
##  6 Easy Shrimp Scampi                           642096  3.00
##  7 Just Another Tiramisu                        648660  3.30
##  8 Vegan Pea and Mint Pesto Bruschetta          664470  4.75
##  9 Shrimp Fettuccine Alfredo, Mamma Mia that's good 1697675  5.22
## 10 Easy Italian Meatballs                      1504227  5.23
```

3. Find 10 types of Riesling wines whose price do not exceed $50 and present your results as a 10x3 tibble, where the columns represent the title of the wine, its ID and its price.

API Request

```
api_url <- paste(
  "https://api.spoonacular.com/food/wine/recommendation",
  "?apiKey=", api_key,
  sep=""
)
url <- paste(
  api_url,
  "&wine=", "riesling",
  '&maxPrice=', "50",
  "&number=", "10",
  sep = ""
)

# using api request to get json data
json_result <- url %>%
  curl() %>%
  readLines(warn=F)
```

Convert to 10x3 tibble

```
# convert json data to 10x3 tibble
result_t <- json_result %>%
  fromJSON() %>%
  .[[1]] %>%
  as_tibble() %>%
  mutate(
    Title = title,
    ID = id,
    Price = price
  ) %>%
  select(Title, ID, Price) %>%
  mutate(Price = as.double(str_replace_all(.$Price, "[$]", "")))

result_t
```

```
## # A tibble: 10 x 3
##    Title                                                           ID Price
##    <chr>                                                        <int> <dbl>
##  1 Domaine LeSeurre Dry Cuvee Classique Riesling               4.80e5 23.0
##  2 Chateau Ste. Michelle Riesling                              4.77e5  9.99
##  3 J.J. Prum Graacher Himmelreich Kabinett Riesling            4.38e5 22.0
##  4 Maximin Grunhaus Herrenberg Riesling Spatlese               2.05e6 40.0
##  5 Weingut Darting Durkheimer Nonnengarten Riesling Kabinett (1 Li~ 2.04e6 18.0
##  6 Weingut Schneider Niederhauser Hermannshohle Riesling Trocken    5.00e5 30.0
##  7 Gunderloch Estate Riesling Dry                              4.48e5 15.0
##  8 Forge Cellars Classique Riesling                            4.60e5 21.0
##  9 Funf Riesling                                               4.93e5  8.99
## 10 Selbach Oster Zeltinger Himmelreich Riesling Kabinett Halbtrock~ 2.04e6 21.0
```