Project 1

Name: Wesley Newcomb Partner: Manuel Ricardo Vargas and Diva Medina Camp

2023-04-09

Background

The World Health Organization has recently employed a new data science initiative, *CSIT-165*, that uses data science to characterize pandemic diseases. *CSIT-165* disseminates data driven analyses to global decision makers.

CSIT-165 is a conglomerate comprised of two fabricated entities: *Global Health Union (GHU)* and *Private Diagnostic Laboratories (PDL)*. Your and your partner's role is to play a data scientist from one of these two entities.

Data

> 2019 Novel Coronavirus COVID-19 (2019-nCoV) Data Repository by John Hopkins CSSE Data for 2019 Novel Coronavirus is operated by the John Hopkins University Center for Systems Science and Engineering (JHU CSSE). Data includes daily time series CSV summary tables, including confirmations, recoveries, and deaths. Country/region are countries/regions hat conform to World Health Organization (WHO). Lat and Long refer to coordinates references for the user. Date fields are stored in MM/DD/YYYY format.

For this project, we will use global data sets for COVID-19 associated confirmations and deaths.

In order to download these files without cloning the whole repository, enter one of two commands in your terminal depending on your machine:

Windows wget https://address.to.data/goes/here.csv Mac curl https://address.to.data/goes/here.csv -O

Instructions

The World Health Organization has recently employed a new data science initiative, CSIT-165, that uses data science to characterize pandemic diseases. CSIT-165 disseminates data driven analyses to global decision makers.

CSIT-165 is a conglomerate comprised of two fabricated entities: Global Health Union (GHU) and Private Diagnostic Laboratories (PDL). Your and your partner's role is to play a data scientist from one of these two entities. Discuss with your partner to decide who will be part of GHU and PDL.

Getting Started

One project member per group must create a new repository on GitHub. Initialize this repository with a readme.md file that lists each member of the group. If your group decides to collaborate using a centralized workflow (recommended), then the project member that created the repository must declare their partners as

collaborators in GitHub. Each project member will clone this repository onto their machine using RStudio. In RStudio, create a project from version control with GitHub using the HTTP address of the repository created by project member.

All project members must first contribute to analyses by uploading data sets respective to the entity they belong to in the CSIT-165 initiative. GHU project members provide time series data counting COVID-19 related recoveries and and deaths. PDL project members provide time series data counting COVID-19 related confirmations.

Project Objectives

This project will encompass many of the lessons we have learned throughout the course. RMarkdown files must be written such that each time you render the document it will download the necessary data sets for analysis. Please render the RMarkdown file the day it is due to reflect the most recent data sets. With this added functionality, your code must be able to analyze the datasets regardless of the date you render your document.

Objective 1

CSIT-165's first objective is to determine where COVID-19 originated from. Predict where the origin started based on the area with the greatest number of confirmations and deaths on the first recorded day in the data set. Show this is the origin using an if statement.

```
# Notice: Copyright Wesley C Newcomb Inc.
#confirmed_df<-read.csv("data/time_series_covid19_confirmed_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
confirmed_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
#deaths_df<-read.csv("data/time_series_covid19_deaths_global.csv",
#                    header=TRUE, stringsAsFactors=FALSE)
deaths_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data/cs
#recovered_df<-read.csv("data/time_series_covid19_recovered_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
recovered_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data

## Confirmed Dataset
confirmed_ordered<-select(arrange(confirmed_df, -X1.22.20),
                          Province.State, Country.Region, X1.22.20)
cat("Confirmed Dataset ")
```

```
## Confirmed Dataset
```

```
head(confirmed_ordered)
```

```
##   Province.State Country.Region X1.22.20
## 1          Hubei          China      444
## 2      Guangdong          China       26
## 3        Beijing          China       14
## 4       Zhejiang          China       10
## 5       Shanghai          China        9
## 6      Chongqing          China        6
```

```
## Deaths Dataset
death_ordered<-select(arrange(deaths_df, -X1.22.20),
                      Province.State, Country.Region, X1.22.20)
cat("Deaths Dataset ")
```

```
## Deaths Dataset
```

```
head(death_ordered)
```

```
##   Province.State Country.Region X1.22.20
## 1          Hubei           China       17
## 2                     Afghanistan        0
## 3                         Albania        0
## 4                         Algeria        0
## 5                         Andorra        0
## 6                          Angola        0
```

```
## Recovered Dataset
recovered_ordered<-select(arrange(recovered_df, -X1.22.20),
                          Province.State, Country.Region, X1.22.20)
cat("Recovered Dataset \n")
```

```
## Recovered Dataset
```

```
head(recovered_ordered)
```

```
##   Province.State Country.Region X1.22.20
## 1          Hubei           China       28
## 2                       Thailand        2
## 3                     Afghanistan        0
## 4                         Albania        0
## 5                         Algeria        0
## 6                         Andorra        0
```

```
cat(confirmed_ordered[1,1], ", ", confirmed_ordered[1,2],
    " has the most confirmed cases on the first day. \n",
    death_ordered[1,1], ", ", death_ordered[1,2],
    " has the most deaths from the virus on the first day. \n",
    recovered_ordered[1,1], ", ", recovered_ordered[1,2],
    " has the most recovered cases from the virus on the first day. \n",
    sep="")
```

```
## Hubei, China has the most confirmed cases on the first day.
## Hubei, China has the most deaths from the virus on the first day.
## Hubei, China has the most recovered cases from the virus on the first day.
```

```
if (confirmed_ordered[1,1] == death_ordered[1,1] &&
    confirmed_ordered[1,1] == recovered_ordered[1,1] &&
    death_ordered[1,1] == recovered_ordered[1,1])
{
  cat(confirmed_ordered[1,1], ", ", confirmed_ordered[1,2],
    " is the most likely origin of the virus. \n", sep="")
}
```

Hubei, China is the most likely origin of the virus.

```
# Find greatest number of deaths on first day
for(i in deaths_df$X1.22.20) {
  if(i > 0) {
    print(i)
  }
}
```

```
## [1] 17
```

```
max(deaths_df$X1.22.20)
```

```
## [1] 17
```

```
first_city_deaths <- deaths_df$Province.State[which.max(deaths_df$X1.22.20)]

# Find greatest number of confirmed cases on first day
max(confirmed_df$X1.22.20)
```

```
## [1] 444
```

```
first_city_confirmations <- confirmed_df$Province.State[which.max(confirmed_df$X1.22.20)]

# Find and print origin city
if (first_city_deaths == first_city_confirmations) {
  covid_city_origin <- first_city_confirmations
}
print(covid_city_origin)
```

```
## [1] "Hubei"
```

```
# Notice: Copyright Diva Medina Camp Inc.
# Get csv files from CSSEGISandData/COVID-19 database
covid_confirmations <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid
covid_deaths <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
covid_recoveries <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19
# Find greatest number of deaths on first day
for(i in covid_deaths$X1.22.20) {
  if(i > 0) {
    print(i)
  }
}
```

```
## [1] 17
```

```
max(covid_deaths$X1.22.20)
```

```
## [1] 17
```

```
first_city_deaths <- covid_deaths$Province.State[which.max(covid_deaths$X1.22.20)]
# Find greatest number of confirmed cases on first day
max(covid_confirmations$X1.22.20)
```

```
## [1] 444
```

```
first_city_confirmation <- covid_confirmations$Province.State[which.max(covid_confirmations$X1.22.20)]
# Find and print city of origin
if (first_city_deaths == first_city_confirmation) {
  covid_city_origin <- first_city_confirmation
}
print(covid_city_origin)
```

```
## [1] "Hubei"
```

Based on the data above: Hubei, China is the most likely origin of the virus. This is based on Hubei, China having by far the most confirmed cases and is the only region with any recoveries or deaths on the first day of data avaliable. Also, all other regions that have confirmed cases on the first day are regions that are near Hubei. The conditional statement also proves that Hubei is the most likely origin of the virus because it shows that the place with the most deaths, recovered, and confirmed cases on the first day is Hubei.

Objective 2

Where is the most recent area to have a first confirmed case? To do this, you will need to use a for loop, if statement, and subsets.

```
# Notice: Copyright Wesley C Newcomb Inc.
#confirmed_df<-read.csv("data/time_series_covid19_confirmed_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
confirmed_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
recent_df<-arrange(confirmed_df[confirmed_df[,ncol(confirmed_df)-1] == 0
                                & confirmed_df[,ncol(confirmed_df)] > 0,])

i<-0
# If there are no new cases today loop back to find most recent region
# to have new cases
if (nrow(recent_df) == 0) {
  while (nrow(recent_df) == 0) {
    i<-i+1
    recent_df<-arrange(confirmed_df[confirmed_df[,ncol(confirmed_df)-1-i] == 0
                                & confirmed_df[,ncol(confirmed_df)-i] > 0,])
  }
}

head(select(recent_df, Province.State, Country.Region, ncol(confirmed_df)-1-i,
            ncol(confirmed_df)-i))
```

```
##     Province.State Country.Region X7.19.22 X7.20.22
## 1 Pitcairn Islands United Kingdom        0        4
```

```
# Vector is small enough that loop is reasonable
for(i in 1:nrow(recent_df))
{
  if (recent_df[i,1] == "") {
    cat(recent_df[i,2], "has recently had their first confirmed case. \n")
  } else {
    if (recent_df[i,2] == "") {
      cat(recent_df[i,1], "has recently had their first confirmed case. \n")
    } else {
      cat(recent_df[i,1], ", ", recent_df[i,2],
          " has recently had their first confirmed case. \n", sep="")
    }
  }
}
```

```
## Pitcairn Islands, United Kingdom has recently had their first confirmed case.
```

```
# Notice: Copyright Diva Medina Camp Inc.
## Most recent area to have a confirmed case
## Use a for loop, if statement, and subsets.
recent_confirmations <- c()
for (i in 1:nrow(covid_confirmations)) {
  if (covid_confirmations$X3.9.23[i] > covid_confirmations$X3.8.23[i]) {
    recent_confirmations <- c(recent_confirmations,covid_confirmations$Country.Region[i])
   }
}
print(recent_confirmations)
```

```
##  [1] "Albania"              "Algeria"              "Australia"
##  [4] "Australia"            "Australia"            "Australia"
##  [7] "Australia"            "Australia"            "Austria"
## [10] "Azerbaijan"           "Bahrain"              "Barbados"
## [13] "Belgium"              "Bolivia"              "Bulgaria"
## [16] "Burma"                "Canada"               "Canada"
## [19] "Canada"               "Chile"                "Cote d'Ivoire"
## [22] "Cuba"                 "Czechia"              "Denmark"
## [25] "Ethiopia"             "Fiji"                 "Finland"
## [28] "France"               "Germany"              "Guatemala"
## [31] "India"                "Indonesia"            "Iran"
## [34] "Israel"               "Jamaica"              "Japan"
## [37] "Korea, South"         "Kosovo"               "Latvia"
## [40] "Lebanon"              "Malaysia"             "Mali"
## [43] "Mexico"               "Montenegro"           "Nepal"
## [46] "New Zealand"          "Norway"               "Pakistan"
## [49] "Philippines"          "Poland"               "Qatar"
## [52] "Russia"               "Saudi Arabia"         "Serbia"
## [55] "Slovakia"             "Slovenia"             "South Africa"
## [58] "Sri Lanka"            "Sweden"               "US"
## [61] "Ukraine"              "United Arab Emirates" "United Kingdom"
## [64] "United Kingdom"       "Venezuela"
```

```
# R Solution
# recent_confirmations <- paste(covid_confirmations$Province.State[which(covid_confirmations$X3.9.23 >
#                               covid_confirmations$Country.Region[which(covid_confirmations$X3.9.23 > c
```

Most recent territories were found by going through the data and selecting the countries that did not have cases before yesterday and had their first cases today. If there are no regions meeting this check, then each previous day is looked at until there are regions found with new cases.

Objective 3

How far away are the areas from objective 2 from where the first confirmed case(s) occurred? Please provide answer(s) in terms of miles. Use the function distm from the R package geosphere to calculate the distance between two coordinates in meters (geosphere::distm). You will need to convert the value returned by distm from meters to miles (this conversion is simple and can be found online). Please use a table or printed statement to describe what Province/State and Country/Region first confirmed cases occurred as well as the distance (in miles) away from the origin. Please print the following: {recent region} is {distance in miles} away from {origin city, origin country}.

```
# Notice: Copyright Wesley C Newcomb Inc.
origin_df<-arrange(confirmed_df, -X1.22.20)[1,]
#confirmed_df<-read.csv("data/time_series_covid19_confirmed_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
confirmed_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
recent_df<-arrange(confirmed_df[confirmed_df[,ncol(confirmed_df)-1] == 0
                                & confirmed_df[,ncol(confirmed_df)] > 0,])
i<-0

# If there are no new cases today loop back to find most recent region
# to have new cases
if (nrow(recent_df) == 0) {
  while (nrow(recent_df) == 0) {
    i<-i+1
    recent_df<-arrange(confirmed_df[confirmed_df[,ncol(confirmed_df)-1-i] == 0
                                & confirmed_df[,ncol(confirmed_df)-i] > 0,])
  }
}

# Compute distances from origin
distances<-distm(select(recent_df, Long, Lat), select(origin_df, Long, Lat))

# Convert from m to miles
distances<-distances * 0.00062137

# Add distance from origin to dataframe and sort by distance
recent_df$distance<-distances[,1]
recent_df<-arrange(recent_df, distance)

head(select(recent_df, Province.State, Country.Region, Lat, Long, distance))
```

```
##     Province.State Country.Region      Lat      Long distance
## 1 Pitcairn Islands United Kingdom -24.3768 -128.3242  8746.96
```

```
# Vector is small enough that loop is reasonable
for (i in 1:nrow(recent_df)) {
  city<-recent_df[i, "Province.State"]

  # If there is no city use country
  if (city == "") {
    city<-recent_df[i, "Country.Region"]
  }

  cat(city, "is", recent_df[i, "distance"],
      "miles away from the virus origin in",
      paste0(origin_df[1, "Province.State"], ","),
      paste0(origin_df[1, "Country.Region"], "."), "\n")
}
```

```
## Pitcairn Islands is 8746.96 miles away from the virus origin in Hubei, China.
```

```
# Notice: Copyright Diva Medina Camp Inc.
library ("geosphere")
# How far away are the areas from objective 2 from where the first confirmed case(s) occurred?
first_city_confirmation_index <- which.max(covid_confirmations$X1.22.20)
area_df <- data.frame(Long = covid_confirmations$Long[which(covid_confirmations$X3.9.23 > covid_confirma
                      ,Lat = covid_confirmations$Lat[which(covid_confirmations$X3.9.23 > covid_confirma
area_df_matrix <- as.matrix(area_df)
# First City Coordinates
First_City_Coords <- data.frame(Long = covid_confirmations$Long[first_city_confirmation_index]
                          ,Lat = covid_confirmations$Lat[first_city_confirmation_index])
First_City_Coords_matrix <- as.matrix(First_City_Coords)
# Find distance between cities
# Use function distm
# distm(x, y)
dist <- distm(area_df_matrix, First_City_Coords_matrix)
# Convert from meters to miles
dist_miles <- round(dist/1609.34, 2)
# Please print the following:
# {recent region} is {distance in miles} away from {origin city, origin country}
distance_from_origin <- data.frame(City = covid_confirmations$Province.State[which(covid_confirmations$
                                   ,Country = covid_confirmations$Country.Region[which(covid_confirmatio
                                   ,Distance = dist_miles)
for(i in 1:length(distance_from_origin$City)) {
  print(paste(distance_from_origin$City[i], distance_from_origin$Country[i], " is ",
              distance_from_origin$Distance[i], "miles away from",
              covid_confirmations$Province.State[first_city_confirmation_index],
              covid_confirmations$Country.Region[first_city_confirmation_index]))
}
```

```
## [1] " Albania  is  4960.44 miles away from Hubei China"
## [1] " Algeria  is  6327.14 miles away from Hubei China"
## [1] "Australian Capital Territory Australia  is  5156.88 miles away from Hubei China"
## [1] "New South Wales Australia  is  5132.81 miles away from Hubei China"
## [1] "Northern Territory Australia  is  3231.78 miles away from Hubei China"
## [1] "South Australia Australia  is  4846.19 miles away from Hubei China"
## [1] "Tasmania Australia  is  5547.63 miles away from Hubei China"
```

```
## [1] "Victoria Australia  is  5181.72 miles away from Hubei China"
## [1] " Austria  is  5015.91 miles away from Hubei China"
## [1] " Azerbaijan  is  3618.98 miles away from Hubei China"
## [1] " Bahrain  is  3722.86 miles away from Hubei China"
## [1] " Barbados  is  9345.19 miles away from Hubei China"
## [1] " Belgium  is  5297.24 miles away from Hubei China"
## [1] " Bolivia  is  11387.03 miles away from Hubei China"
## [1] " Bulgaria  is  4665.79 miles away from Hubei China"
## [1] " Burma  is  1185.83 miles away from Hubei China"
## [1] "British Columbia Canada  is  5592.11 miles away from Hubei China"
## [1] "Ontario Canada  is  6669.8 miles away from Hubei China"
## [1] "Quebec Canada  is  6642.87 miles away from Hubei China"
## [1] " Chile  is  12042.48 miles away from Hubei China"
## [1] " Cote d'Ivoire  is  7554.68 miles away from Hubei China"
## [1] " Cuba  is  8759.71 miles away from Hubei China"
## [1] " Czechia  is  4907.37 miles away from Hubei China"
## [1] " Denmark  is  4928.15 miles away from Hubei China"
## [1] " Ethiopia  is  4822.45 miles away from Hubei China"
## [1] " Fiji  is  5503.47 miles away from Hubei China"
## [1] " Finland  is  4251.24 miles away from Hubei China"
## [1] " France  is  5561.51 miles away from Hubei China"
## [1] " Germany  is  5061.55 miles away from Hubei China"
## [1] " Guatemala  is  8887.15 miles away from Hubei China"
## [1] " India  is  2185.45 miles away from Hubei China"
## [1] " Indonesia  is  2187.16 miles away from Hubei China"
## [1] " Iran  is  3407.35 miles away from Hubei China"
## [1] " Israel  is  4487.71 miles away from Hubei China"
## [1] " Jamaica  is  8996.19 miles away from Hubei China"
## [1] " Japan  is  1536.31 miles away from Hubei China"
## [1] " Korea, South  is  956.32 miles away from Hubei China"
## [1] " Kosovo  is  4883.7 miles away from Hubei China"
## [1] " Latvia  is  4379.19 miles away from Hubei China"
## [1] " Lebanon  is  4361.62 miles away from Hubei China"
## [1] " Malaysia  is  1959.34 miles away from Hubei China"
## [1] " Mali  is  7051.8 miles away from Hubei China"
## [1] " Mexico  is  8027.44 miles away from Hubei China"
## [1] " Montenegro  is  4951.13 miles away from Hubei China"
## [1] " Nepal  is  1693.71 miles away from Hubei China"
## [1] "Niue New Zealand  is  6205.4 miles away from Hubei China"
## [1] " Norway  is  4825.84 miles away from Hubei China"
## [1] " Pakistan  is  2540.06 miles away from Hubei China"
## [1] " Philippines  is  1384.87 miles away from Hubei China"
## [1] " Poland  is  4701.39 miles away from Hubei China"
## [1] " Qatar  is  3702.32 miles away from Hubei China"
## [1] " Russia  is  2133.37 miles away from Hubei China"
## [1] " Saudi Arabia  is  4096.29 miles away from Hubei China"
## [1] " Serbia  is  4838.64 miles away from Hubei China"
## [1] " Slovakia  is  4766.42 miles away from Hubei China"
## [1] " Slovenia  is  5040.05 miles away from Hubei China"
## [1] " South Africa  is  7226.35 miles away from Hubei China"
## [1] " Sri Lanka  is  2580.62 miles away from Hubei China"
## [1] " Sweden  is  4515.04 miles away from Hubei China"
## [1] " US  is  7129.09 miles away from Hubei China"
## [1] " Ukraine  is  4277.99 miles away from Hubei China"
```

```
## [1] " United Arab Emirates  is  3594.43 miles away from Hubei China"
## [1] "Turks and Caicos Islands United Kingdom  is  8798.46 miles away from Hubei China"
## [1] " United Kingdom  is  5383.81 miles away from Hubei China"
## [1] " Venezuela  is  9857.19 miles away from Hubei China"
```

Objective 4

CSIT-165 characterizes diseases using risk scores. Risk scores are calculated as the ratio of deaths to confirmations, that is Riskscore=100×deathsconfirmations . Risk scores equal to 100 indicate the highest risk while risk scores equal to 0 indicate the lowest risk. Areas are characterized as being especially vulnerable to loss if they have higher risk scores. For this assignment, exclude cruise ships (hint: they have lat and long coordinates of 0 or NA in this data set, filter this out before calculating risk scores).

Which area of the world currently has the lowest risk score (if more than one, display the one with the most confirmations)? Which area of the world currently has the highest risk score (if more than one, display the one with the most confirmations)? How do risk scores in these areas compare to global risk score? Why might it be helpful to calculate metrics like risk scores for different areas of the world and what would their limitations be (what assumptions does risk score make and what important variables might be left out)?

```
# Notice: Copyright Wesley C Newcomb Inc.
# Datasets respresent a cumlative sum by date, so last column represents
# summation for region
#confirmed_df<-read.csv("data/time_series_covid19_confirmed_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
confirmed_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
confirmed_df<-select(confirmed_df, Province.State,
                     Country.Region, ncol(confirmed_df))
names(confirmed_df)[3] <- "confirmed"
#deaths_df<-read.csv("data/time_series_covid19_deaths_global.csv",
#                    header=TRUE, stringsAsFactors=FALSE)
deaths_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data/cs
deaths_df<-select(deaths_df, Province.State,
                  Country.Region, ncol(deaths_df))
names(deaths_df)[3] <- "deaths"
#recovered_df<-read.csv("data/time_series_covid19_recovered_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
recovered_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
recovered_df<-select(recovered_df, Province.State,
                     Country.Region, ncol(recovered_df))
names(recovered_df)[3] <- "recovered"

# Combine the datasets into one and fill NA with 0
combined_df<-full_join(confirmed_df, recovered_df,
                       by=c("Province.State", "Country.Region"))
combined_df<-full_join(combined_df, deaths_df,
                       by=c("Province.State", "Country.Region"))
combined_df[is.na(combined_df)] <- 0

# Assignment is unclear if we are to consider state and region or
# just region. Based on how data is formatted, I think it is cleaner
# and makes more sence to use region only.  For instance, in confirmed
```

```
# dataset, Canada is broken up by region, but in recovered dataset it
# uses Canada as a whole.  There are numerous examples of this in
# the data
grouped_df<-as.data.frame(summarise_each(group_by(
  select(combined_df, -Province.State), Country.Region), sum))
```

Objective 4.1

```
## Warning: ‘summarise_each_()‘ was deprecated in dplyr 0.7.0.
## i Please use ‘across()‘ instead.
## i The deprecated feature was likely used in the dplyr package.
##   Please report the issue at <https://github.com/tidyverse/dplyr/issues>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
# Compute risk and burden by region
grouped_df$risk<-grouped_df$deaths / grouped_df$recovered
grouped_df$burden<-grouped_df$confirmed * grouped_df$risk

cat("Highest risk scores ")
```

```
## Highest risk scores
```

```
head(arrange(grouped_df, -risk, -confirmed))
```

```
##   Country.Region confirmed recovered  deaths risk burden
## 1             US 103802702         0 1123836  Inf    Inf
## 2          India  44690738         0  530779  Inf    Inf
## 3         France  39866718         0  166176  Inf    Inf
## 4        Germany  38249060         0  168935  Inf    Inf
## 5         Brazil  37076053         0  699276  Inf    Inf
## 6          Japan  33320438         0   72997  Inf    Inf
```

```
cat("Highest risk scores, that are not infinite ")
```

```
## Highest risk scores, that are not infinite
```

```
head(arrange(grouped_df[grouped_df$risk != Inf,], -risk, -confirmed))
```

```
##      Country.Region confirmed recovered deaths risk burden
## NA             <NA>        NA        NA     NA   NA     NA
## NA.1           <NA>        NA        NA     NA   NA     NA
## NA.2           <NA>        NA        NA     NA   NA     NA
## NA.3           <NA>        NA        NA     NA   NA     NA
## NA.4           <NA>        NA        NA     NA   NA     NA
```

```
cat("Lowest Risk Scores ")
```

```
## Lowest Risk Scores
```

```
head(arrange(grouped_df, risk, confirmed))
```

```
##     Country.Region confirmed recovered deaths risk burden
## 1     Korea, North         1         0      6  Inf    Inf
## 2       MS Zaandam         9         0      2  Inf    Inf
## 3 Diamond Princess       712         0     13  Inf    Inf
## 4         Kiribati      5014         0     18  Inf    Inf
## 5            Nauru      5247         0      1  Inf    Inf
## 6            Palau      5991         0      9  Inf    Inf
```

```
cat("Lowest risk scores, that are not zero ")
```

```
## Lowest risk scores, that are not zero
```

```
head(arrange(grouped_df[grouped_df$risk != 0,], risk, confirmed))
```

```
##       Country.Region confirmed recovered deaths risk burden
## 94      Korea, North         1         0      6  Inf    Inf
## 108       MS Zaandam         9         0      2  Inf    Inf
## 50  Diamond Princess       712         0     13  Inf    Inf
## 93          Kiribati      5014         0     18  Inf    Inf
## 127            Nauru      5247         0      1  Inf    Inf
## 138            Palau      5991         0      9  Inf    Inf
```

```
global_confirmed<-sum(grouped_df$confirmed)
global_deaths<-sum(grouped_df$deaths)
global_recovered<-sum(grouped_df$recovered)
global_risk<-global_deaths / global_recovered
global_burden<-global_confirmed * global_risk

cat("Global Data\n",
    "Confirmed:", global_confirmed, "\n",
    "Deaths:   ", global_deaths, "\n",
    "Recovered:", global_recovered, "\n",
    "Risk:     ", global_risk, "\n",
    "Burden:   ", global_burden, "\n")
```

```
## Global Data
##  Confirmed: 676570149
##  Deaths:    6881802
##  Recovered: 0
##  Risk:      Inf
##  Burden:    Inf
```

```
# Notice: Copyright Manuel Ricardo Vargas Inc.
# Import the data for global confirmed cases
confirmed_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/cs
confirmed_data <- read.csv(confirmed_url, header = TRUE, stringsAsFactors = FALSE)

# Import the data for global deaths
```

```r
deaths_url <- "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse
deaths_data <- read.csv(deaths_url, header = TRUE, stringsAsFactors = FALSE)

# Remove cruise ships from the confirmed cases data
confirmed_data <- confirmed_data[!(confirmed_data$Lat == 0 | is.na(confirmed_data$Lat)), ]
confirmed_data <- confirmed_data[!(confirmed_data$Long == 0 | is.na(confirmed_data$Long)), ]

# Remove cruise ships from the deaths data
deaths_data <- deaths_data[!(deaths_data$Lat == 0 | is.na(deaths_data$Lat)), ]
deaths_data <- deaths_data[!(deaths_data$Long == 0 | is.na(deaths_data$Long)), ]

# Filter both confirmed and deaths data to only include common countries
common_countries <- intersect(confirmed_data$Country.Region, deaths_data$Country.Region)
confirmed_cases <- confirmed_data[confirmed_data$Country.Region %in% common_countries, c("Country.Regio
deaths <- deaths_data[deaths_data$Country.Region %in% common_countries, c("Country.Region", tail(names(

# Calculate risk scores
risk_scores <- round(100 * deaths[, 2] / confirmed_cases[, 2], 2)

# Create data frame with country names and risk scores
risk_scores_df <- data.frame(Country = confirmed_cases$Country.Region, RiskScore = risk_scores)

# Sort risk scores by ascending order and select first row
lowest_risk <- risk_scores_df[order(risk_scores_df$RiskScore), ][1, ]

# If there are multiple areas with lowest risk, select the one with most confirmations
if (sum(!is.na(lowest_risk$Confirmations)) > 0) {
  lowest_risk <- lowest_risk[which(lowest_risk$Confirmations == max(lowest_risk$Confirmations, na.rm = 
}

# Display the result
cat("The area with the lowest risk score is", lowest_risk$Country, "with a risk score of", lowest_risk$
```

```
## The area with the lowest risk score is Antarctica with a risk score of 0 and confirmations.
```

```r
# Sort risk scores by descending order and select first row
highest_risk <- risk_scores_df[order(-risk_scores_df$RiskScore), ][1, ]

# If there are multiple areas with highest risk, select the one with most confirmations
if (sum(!is.na(highest_risk$Confirmations)) > 0) {
  highest_risk <- highest_risk[which(highest_risk$Confirmations == max(highest_risk$Confirmations, na.r
}

# Display the result
cat("The area with the highest risk score is", highest_risk$Country, "with a risk score of", highest_ris
```

```
## The area with the highest risk score is Korea, North with a risk score of 600 and confirmations.
```

Based on how the equation is written, any region which has had at least one person recover and no deaths will have a risk score of zero. Examples of this can be seen in the "Lowest Risk Scores" table above. When filtering out risk scores of 0, the regions of lowest risk can be seen in "Lowest risk scores, that are not zero" table above. Any region that has no recoveries and yet at least one death will have infinite risk. Examples

of this can be seen in the "Highest risk scores" table above. If filtering out regions that have infinite risk, we see that the regions in the "Highest risk scores, that are not infinite" table above. When looking at the global score, it seems like the risk is high when considering it represents the people that have recovered versus those who have died. This value seems especially high when looking at the regions in the "Lowest risk scores, that are not zero" table, but when compared to the "Highest risk scores, that are not infinite" table where the risk scores are extremely high, the global risk seems less significant. This wide range in risk numbers indicates that while the risk in some regions is extremely high, for the most part, the risk is rather low globally.

Risk assessments like this are important because they are good indicators of where danger is located or help is needed that can be used across many industries. For example, the travel industry may wish to impose bans on travelling to and from locations of high risk. The medical field can use these values to determine locations that are in the most need for medical support. Research fields may also use this data to help identify trends. For example, if a region has a high amount of recoveries and almost no deaths, i.e. low risk score, it may be worth looking into what kind of treatment they are using in that region and if it could be used in other locations throughout the world. The thing to be careful though is that risk scores may be a little misleading. For instance, several regions have almost no cases, but one death and no recoveries causing a massive risk score. Even though these regions have pretty much no cases, they are still seen as extremely risky. This is why it could be beneficial to filter out the extremes before considering the data as valid.

Objective 4.2 You are asked to make two tables with the top 10 countries that have the most COVID-19 related confirmations and and deaths. Make sure to include all of the counts for the country, not just the counts for one area in the country. To do this we will need to sum all of the values for each country, create new data frames from these values, and use the package kable to convert those data frames into tables.

Hint: Sum each country's counts by subsetting the data frame using a list of countries available in the data set. Use a for loop to iterate through the data frame using the list of countries. For each country, calculate the count sum and assign this value to a list.

```
# Notice: Copyright Wesley C Newcomb Inc.
# Datasets respresent a cumlative sum by date, so last column represents
# sumation for region
#confirmed_df<-read.csv("data/time_series_covid19_confirmed_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
confirmed_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
confirmed_df<-select(confirmed_df, Province.State,
                     Country.Region, ncol(confirmed_df))
names(confirmed_df)[3] <- "confirmed"
#deaths_df<-read.csv("data/time_series_covid19_deaths_global.csv",
#                    header=TRUE, stringsAsFactors=FALSE)
deaths_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data/cs
deaths_df<-select(deaths_df, Province.State,
                  Country.Region, ncol(deaths_df))
names(deaths_df)[3] <- "deaths"
#recovered_df<-read.csv("data/time_series_covid19_recovered_global.csv",
#                       header=TRUE, stringsAsFactors=FALSE)
recovered_df <- read.csv("https://raw.githubusercontent.com/Chato1969/COVID-19/master/csse_covid_19_data
recovered_df<-select(recovered_df, Province.State,
                     Country.Region, ncol(recovered_df))
names(recovered_df)[3] <- "recovered"

# Combine the datasets into one and fill NA with 0
combined_df<-full_join(confirmed_df, recovered_df,
                       by=c("Province.State", "Country.Region"))
```

```
combined_df<-full_join(combined_df, deaths_df,
                       by=c("Province.State", "Country.Region"))
combined_df[is.na(combined_df)] <- 0

# Group and combine data by region
grouped_df<-as.data.frame(summarise_each(group_by(
  select(combined_df, -Province.State), Country.Region), sum))

# Compute risk and burden by region
grouped_df$risk<-grouped_df$deaths / grouped_df$recovered
grouped_df$burden<-grouped_df$confirmed * grouped_df$risk

confirmed_tb = kable(arrange(grouped_df, -confirmed)[1:5,])
deaths_tb = kable(arrange(grouped_df, -deaths)[1:5,])
recovered_tb = kable(arrange(grouped_df, -recovered)[1:5,])

cat("Top 5 confirmed regions ")
```

```
## Top 5 confirmed regions
```

```
confirmed_tb
```

Country.Region	confirmed	recovered	deaths	risk	burden
US	103802702	0	1123836	Inf	Inf
India	44690738	0	530779	Inf	Inf
France	39866718	0	166176	Inf	Inf
Germany	38249060	0	168935	Inf	Inf
Brazil	37076053	0	699276	Inf	Inf

```
cat("Top 5 deaths regions ")
```

```
## Top 5 deaths regions
```

```
deaths_tb
```

Country.Region	confirmed	recovered	deaths	risk	burden
US	103802702	0	1123836	Inf	Inf
Brazil	37076053	0	699276	Inf	Inf
India	44690738	0	530779	Inf	Inf
Russia	22075858	0	388478	Inf	Inf
Mexico	7483444	0	333188	Inf	Inf

```
cat("Top 5 recovered regions")
```

```
## Top 5 recovered regions
```

```
recovered_tb
```

Country.Region	confirmed	recovered	deaths	risk	burden
Afghanistan	209451	0	7896	Inf	Inf
Albania	334457	0	3598	Inf	Inf
Algeria	271496	0	6881	Inf	Inf
Andorra	47890	0	165	Inf	Inf
Angola	105288	0	1933	Inf	Inf

```r
# Notice: Copyright Manuel Ricardo Vargas Inc.
# Load the CSV files for COVID-19 data
confirmed_cases <- read.csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covi
deaths <- read.csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/

# Extract the country names from the data frames
countries <- unique(confirmed_cases$Country.Region)

# Initialize empty lists for the sums of confirmed cases and deaths
confirmed_sums <- list()
death_sums <- list()

# Calculate the sums of confirmed cases and deaths for each country
for (country in countries) {
  confirmed_sums[[country]] <- sum(confirmed_cases[confirmed_cases$Country.Region == country, -(1:4)])
  death_sums[[country]] <- sum(deaths[deaths$Country.Region == country, -(1:4)])
}

# Convert the sums into data frames
confirmed_df <- data.frame(Country = names(confirmed_sums), Confirmed = unlist(confirmed_sums))
death_df <- data.frame(Country = names(death_sums), Deaths = unlist(death_sums))

# Sort the data frames by the values of confirmed cases and deaths, respectively
confirmed_df <- confirmed_df[order(-confirmed_df$Confirmed),]
death_df <- death_df[order(-death_df$Deaths),]

# Select the top 5 countries with the most confirmed cases and deaths
top_5_confirmed <- head(confirmed_df, 5)
top_5_deaths <- head(death_df, 5)

# Convert the data frames into tables using the kable function
library(knitr)
kable(top_5_confirmed)
```

	Country	Confirmed
US	US	53813184406
India	India	29131119694
Brazil	Brazil	21182690594
France	France	16105911886
Germany	Germany	13686043720

GitHub Log

#{bash gitlog} # By Wesley: git log --pretty=format:"%nSubject: %s%nAuthor: %aN%nDate: %aD%nBody: %b" # By Diva: Biscuit:Project_1_Wesv4 miuixtli$ git log --pretty=format:"%nSubject: %s%nAuthor: %aN%nDate: %aD%nBody: %b" #

```
##
##  Subject: Updated responses to be more generic and automatic such that they are created based on data
##  pulled automatically from the table. Fixed typos and minor bugs.
##  Author: Wesley Newcomb
##  Date: Wed, 5 Apr 2023 18:06:15 -0700
```

```
##  Body:
##
##  Subject: Added first rev of readme.md
##  Author: Wesley Newcomb
##  Date: Wed, 5 Apr 2023 23:28:47 -0700
##  Body:
##
##  Subject:  Added first rev of index.md
##  Author: Wesley Newcomb
##  Date: Wed, 5 Apr 2023 23:25:33 -0700
##  Body:
##
##  Subject: Update README.md
##  Author: Wesley Newcomb
##  Date: Thu, 6 Apr 2023 10:20:17 -0700
##  Body: Added names to the README file
##
##  Subject: Initial commit
##  Author: Wesley Newcomb
##  Date: Fri, 7 Apr 2023 19:05:40 -0700
##  Body:
##
##  Subject: Add files via upload
##  Author: Wesley Newcomb
##  Date: Fri, 7 Apr 2023 19:14:55 -0700
##  Body:
##
##  Subject:  Update README.md
##  Author: Wesley Newcomb
##  Date: Fri, 7 Apr 2023 19:20:29 -0700
##  Body:
##
##  Subject: Merge branch 'new-chart'
##  Author: Wesley Newcomb
##
##  Subject: Completed Objective_1.
##  Author: Diva
##  Date: Sat, 8 Apr 2023 13:50:00 -0700
##  Body:
##
##  Subject: Complete Objective 2 program.
##  Author: Diva
##  Date: Sat, 8 Apr 2023 14:08:58 -0700
##  Body:
##
##  Subject: Completed Objective 3.
##  Author: Diva
##  Date: Sat, 8 Apr 2023 14:10:48 -0700
##  Body:
##
##  Subject: Corrected misspelling.
##  Author: Diva
##  Date: Sat, 8 Apr 2023 14:31:29 -0700
##  Body:
```

```
##
##  Subject Created file.
##  Author: Diva
##  Date: Sat, 8 Apr 2023 19:39:43 -0700
##  Body:
##
##  Subject: Merge pull request #1 from Chato1969/Diva_Branch
##  Author: Wesley Newcomb
##  Date: Sat, 8 Apr 2023 18:18:03 -0700
##  Body: Diva branch
```