MAT 175 Fall 2023
Exercise #9

In this exercise we are going to work on keeping code and data organized (chs. 39, 40, and 5) from the text).

40.1 Whenever starting something new in RStudio, instead of just creating a new script or markdown file, I prefer using a `Project`. A project is a way to keep all components of a data analysis project (data, scripts, markdown files, README files, etc.) in one folder. If you setup a project, by default any files you create or work on will be saved to that project folder.

For example, on my machine, I have a MAT175 folder that includes separate projects for your homework assignments, in-class exercises, etc.

Your text describes the procedure for starting a project in section 40.1. *Try setting up your own MAT175 project.*

If I were you I would move all of my MAT175 associated files into that project.

39 If I know that I am going to be working on a project that will be large, have many versions, involve collaborates, or I will want to share with potential employers, then I will set it up first as a GitHub repository.

Git is a version control system that you run on your own machine. GitHub is a Git repository hosting service. It's been said that GitHub is to Git what Facebook is to your actual face. In short, GitHub does the version control stuff of Git but also allows you to share projects with collaborators or employers.

So, let's start this process by having you *create a Github account*: here. It is a pretty straightforward process but is also described in your book: section 39.2.

The next step is linking your computer to your Github account. Your text recommends doing this by downloading Git and connecting directly through RStudio. However, I like using GitHub Desktop (download here).

GitHub desktop allows you to create Git repositories and link them with GitHub, and it allows you to do so with any IDE or programming language, not just RStudio. I like GitHub Desktop because it has broader use.

We will go through the steps of producing a repository that will ALSO be a folder for an RStudio project, and the process of "cloning" repositories, and "pushing" and "pulling" code updates between RStudio and GitHub. Your text goes into detail about this process in ch. 39. I highly suggest you read through the procedure.

Here is a (link)[https://docs.github.com/en/desktop] with more detail on GitHub Desktop.

Your goal at the end of this section will be to have 1) a GitHub account, 2) a repository/R project with a script or RMarkdown file in it, 3) practiced pushing and pulling updated repos.

This will be very useful if you choose to work with a partner on your midterm project!

Github Link: (https://github.com/arogyad/MAT175E9-test)[https://github.com/arogyad/MAT175E9-test]

5.1 You need to know where your datafiles are, and you need to know how to get to that location using RStudio. If you organize your RStudio usage into projects, like previously described, this should be quite straightforward. Unfortunately, this is not always the case and different computer systems have different ways to indicate a path, so you will have to know the system you are using.

The path of a file is a list of directory names that can be thought of as instructions on what folders to click on, and in what order, to find the file. If these instructions are for finding the file from the root directory

we refer to it as the full path. If the instructions are for finding the file starting in the working directory we refer to it as a relative path.

```
#path to dslabs package:
system.file(package = "dslabs")
```

```
## [1] "/Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library/dslabs"
```

This path is going to be unique to your computer. So, the book recommends that you use relative paths because that makes the code more portable.

For example, this is how I would use a full path to upload the `exams.rda` dataset into RStudio on my computer. If you try and run the code on your computer, you will get an error saying "No such file or directory".

```
load("C:/Users/ahoward1/Documents/MAT 175 - Intro to Data Science - Fall 2023/Exercises/exams.rda")
```

Instead, if I use the relative path to my working directory, and assuming the dataset is also in your working directory, the below code should work on both of our machines.

```
dir <- getwd()
load(file.path(dir,"exams.rda"))
```

You can change the working directory with the function 'setwd()'.

Move your 'exams.rda' dataset to some other location (drag and drop if need be) and try to load it to your session. Use relative paths if at all possible.

```
# Write your code here.
# Remember to use quote marks "_"
dir <- getwd()
load(file.path(dir, "../exams.rda"))
```

5.2 It's not likely that your datafile is already in R dataframe format. You will have to convert it to a form that R can work with. We will first assume that the datafile is a simple text file, not in a proprietary format like Microsoft Excel. In this case, there are several tools you can use: base R has `read.table`, `read.csv`, `read.delim` and so on. We will use the `readr` package, which is part of `tidyverse`. See section 5.2 for a list of functions that we can use. You will use the `read_csv` often, since comma-separated datafiles are pretty common.

In order to use `read_csv()` we need to load the `readr` package. The larger `tidyverse` package includes `readr`, so we can load that instead.

```
library(tidyverse)
```

The first thing to do is to look at the file. If necessary, you can edit the file; for example, if there are explanatory text that is not part of the data, you should edit it out. I suggest you look at the file using your computer's text editor, but you can also use the `read_lines` function to take a look.

```
# The following function shows the first 10 lines of the file.
# read_lines("path to the file", n_max=10)
read_lines("./AllCountries.csv", n_max=5)
```

```
## [1] "Country,LandArea,Population,Energy,Rural,Military,Health,HIV,Internet,Developed,BirthRate,Elderl
## [2] "Afghanistan,652230,29021099,,76.0,4.4,3.7,,1.7,,46.5,2.2,43.9"
## [3] "Albania,27400,3143291,2088,53.3,,8.2,,23.9,1,14.6,9.3,76.6"
## [4] "Algeria,2381740,34373426,37069,34.8,13,10.6,0.1,10.2,1,20.8,4.6,72.4"
## [5] "American Samoa,200,66107,,7.7,,,,,,,,"
```

There should be a `AllCountries.csv` file on Canvas. Download it, put it somewhere, take a look at it, and then use the `read_csv` function to load it into R.

```
# Write your commands here.
# You should give the file a name, like `countries`.
country <- read_csv(file.path(getwd(), "AllCountries.csv"))
```

```
## Rows: 213 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Country
## dbl (12): LandArea, Population, Energy, Rural, Military, Health, HIV, Intern...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Exercise: find the five most populous countries.

```
# Write your code here.
# Remember `arrange` and maybe `select` also.
country %>% slice_max(Population, n = 5)
```

```
## # A tibble: 5 x 13
##   Country       LandArea Population  Energy Rural Military Health   HIV Internet
##   <chr>            <dbl>      <dbl>   <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 China          9327480 1324655000 2116427  56.9     16.1   10.3  NA       22.5
## 2 India          2973190 1139964932  620973  70.5     14.7    4.4   0.3      4.5
## 3 United States  9147420  304375000 2283722  18.3     18.6   18.7   0.6     75.8
## 4 Indonesia      1811570  227345082  198679  48.5      5.3    6.2   0.2      7.9
## 5 Brazil         8459420  191971506  248528  14.4      5.9    6    NA       37.5
## # i 4 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>
```

If you look carefully at the original file, you should note that there are empty cells, indicating missing data. What happens to these empty cells after you import the data into R?

Ans: They change into NA values.

Beside plain text files, it is also very common to encounter data in the form of Microsoft Excel files. They are so common that there are packages built to read Excel files directly. (Excel can save files to `.csv` format, so even without these packages, we can still load Excel files into R.) Be warned: Excel files can contain complicated functions and formatting, so the import into R can be problematic.

We will use the `readxl` package, so we first load it.

```
library(readxl)
```

The book lists several functions to read Excel files. The most recent is `read_xlsx` which can be used to read the most recent type of Excel format.

Exercise: There should also be an Excel file called `AllCountries` in Canvas. Download that, put it somewhere on your computer, and then load it to your workspace using one of the read excel functions. Compare it with what you get when you load the `.csv` file. Are they the same?

Ans: Visually both the files look exactly the same.

```
# Write your code here.
country_xls <- read_xls(file.path(getwd(), "./AllCountries.xls"))
```

```
head(country_xls)
```

```
## # A tibble: 6 x 13
##   Country        LandArea Population Energy Rural Military Health   HIV Internet
##   <chr>             <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 Afghanistan      652230   29021099     NA  76        4.4    3.7  NA        1.7
## 2 Albania           27400    3143291   2088  53.3     NA      8.2  NA       23.9
## 3 Algeria         2381740   34373426  37069  34.8     13     10.6   0.1     10.2
## 4 American Samoa      200      66107     NA   7.7     NA     NA    NA       NA
## 5 Andorra             470      83810     NA  11.1     NA     21.3  NA       70.5
## 6 Angola          1246700   18020668  10972  43.3     NA      6.8   2        3.1
## # i 4 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>
```

```
print(head(country))
```

```
## # A tibble: 6 x 13
##   Country        LandArea Population Energy Rural Military Health   HIV Internet
##   <chr>             <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 Afghanistan      652230   29021099     NA  76        4.4    3.7  NA        1.7
## 2 Albania           27400    3143291   2088  53.3     NA      8.2  NA       23.9
## 3 Algeria         2381740   34373426  37069  34.8     13     10.6   0.1     10.2
## 4 American Samoa      200      66107     NA   7.7     NA     NA    NA       NA
## 5 Andorra             470      83810     NA  11.1     NA     21.3  NA       70.5
## 6 Angola          1246700   18020668  10972  43.3     NA      6.8   2        3.1
## # i 4 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>
```

We should be able to export R data to a plain text format so that others who don't use R can still have access to the data. (Even plain text can have complications: there are ASCII, UTF-8, UTF-16, and other text encodings.) The function to do this is `write_csv()` (or `write_delim`, or `write_tsv`, etc). The basic usage is `write_csv(x,path)` where `x` is the name of an R dataframe (or tibble) and `path` is your intended location of the exported file, including its name. The default path is your working directory.

Try out the `write_csv` function on an R dataset, for example, on the `exams` dataframe. Don't forget to `load` the exams dataset into your workspace first. Check that the function works.

```
# Write your code here.
write_csv(exams, "exams.csv")
```

Here are some more exercises about manipulating dataframes. You should have the ***AllCountries*** *dataframe in your R session by now. Try doing the following things (these are separate tasks):*

- add a new column ***density*** *which shows the number of people per sq kilometer. Then find the top 10 densest countries. Also the 10 least dense countries.*

```
country$density <- (country$Population / country$LandArea)
country %>% top_n(n = 10, density)
```

```
## # A tibble: 10 x 14
##    Country       LandArea Population Energy Rural Military Health   HIV Internet
##    <chr>            <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
##  1 Bahrain            760     775585   9226  11.5     15.6   10.3  NA       51.9
##  2 Bangladesh      130170  160000128  27944  72.9     10.8    7.4   0.1      0.3
##  3 Bermuda             50      64200     NA   0       NA     NA    NA       79.4
##  4 Gibraltar           10      31032    159   0       NA     NA    NA       58
##  5 Hong Kong SA~     1042    6977700  14138   0       NA     NA    NA       59.1
##  6 Macao SAR, C~       28     526178     NA   0       NA     NA    NA       49.2
##  7 Maldives           300     305027     NA  62.1     NA     13.8   0.1     23.5
##  8 Malta              320     411950    819   5.7      1.5   12.3   0.1     49.5
##  9 Monaco               2      32715     NA   0       NA     15.8  NA       NA
## 10 Singapore          700    4839400  18523   0       26.7    7.8   0.1     69.6
## # i 5 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>, density <dbl>
```

```
country %>% top_n(n = 10, -density)
```

```
## # A tibble: 10 x 14
##    Country    LandArea Population Energy Rural Military Health   HIV Internet
##    <chr>         <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
##  1 Australia   7682300   21431800 130113  11.3      7.7   17.1   0.1     70.8
##  2 Botswana     566730    1921122   2117  40.4     NA     16.6  24.9      6.2
##  3 Canada      9093510   33311400 266771  19.6      7.4   17.2   0.2     75.3
##  4 Greenland    410450      56328     NA  16.5     NA     NA    NA       63.9
##  5 Iceland      100250     317414   5255   7.7      0.1   13.1   0.3     90.5
##  6 Libya       1759540    6294181  18221  22.5     NA      5.5  NA        5.1
##  7 Mauritania  1030700    3215043     NA  59       NA      4.9   0.7      1.9
##  8 Mongolia    1553560    2641216   3152  42.8     NA      7.5   0.1     12.5
##  9 Namibia      823290    2129854   1752  63.2     NA     12.1  13.7      5.3
## 10 Suriname     156000     515124     NA  25.1     NA     13.6   1       21.1
## # i 5 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>, density <dbl>
```

- construct a new dataframe that consists only of health-related data. (Of course, you should include the name of the countries too.)

```
country_health <- data.frame(with(country, cbind(Country, Population, Health, HIV, BirthRate, LifeExpec
country_health
```

```
##                          Country Population Health  HIV BirthRate
## 1                    Afghanistan   29021099    3.7 <NA>      46.5
## 2                        Albania    3143291    8.2 <NA>      14.6
## 3                        Algeria   34373426   10.6  0.1      20.8
## 4                 American Samoa      66107   <NA> <NA>      <NA>
## 5                        Andorra      83810   21.3 <NA>      10.4
## 6                         Angola   18020668    6.8    2      42.9
## 7            Antigua and Barbuda      86634     11 <NA>      <NA>
## 8                      Argentina   39882980   13.7  0.5      17.3
## 9                        Armenia    3077087    7.2  0.1      15.3
## 10                         Aruba     105455   <NA> <NA>      11.7
## 11                     Australia   21431800   17.1  0.1      13.8
## 12                       Austria    8336926   15.8  0.3       9.3
## 13                    Azerbaijan    8680100    2.5  0.1      17.8
## 14                Bahamas, The     337668   13.1  3.1      16.7
## 15                       Bahrain     775585   10.3 <NA>        18
## 16                    Bangladesh  160000128    7.4  0.1      21.4
## 17                      Barbados     255203   10.8  1.3      11.2
## 18                       Belarus    9680850    8.2  0.2      11.1
## 19                       Belgium   10708433   14.8  0.2      11.7
## 20                        Belize     322100   10.8  2.4      24.7
## 21                         Benin    8662086    8.8  1.2      39.4
## 22                       Bermuda      64200   <NA> <NA>      12.5
## 23                        Bhutan     686789     13  0.2      21.5
## 24                       Bolivia    9694113    8.2  0.2      27.1
## 25         Bosnia and Herzegovina    3773100     14 <NA>       9.1
## 26                      Botswana    1921122   16.6 24.9      24.5
## 27                        Brazil  191971506      6 <NA>      16.2
## 28            Brunei Darussalam     392280      7 <NA>      19.8
## 29                      Bulgaria    7623395   11.2  0.1      10.2
## 30                  Burkina Faso   15233884   16.3  1.2      47.2
## 31                       Burundi    8074254   11.8  3.5      34.5
## 32                      Cambodia   14562008      9  0.6      24.7
## 33                      Cameroon   19088385    6.1  5.3      36.9
## 34                        Canada   33311400   17.2  0.2      11.3
## 35                    Cape Verde     498672   10.1 <NA>      24.1
## 36                 Cayman Islands      54248   <NA> <NA>      <NA>
## 37       Central African Republic    4339263     11  5.1      35.4
## 38                          Chad   10913667   13.8  3.4      45.7
## 39                Channel Islands     149581   <NA> <NA>       9.3
## 40                         Chile   16803952   15.6  0.4      14.9
## 41                         China 1324655000   10.3 <NA>      12.1
## 42                      Colombia   45012096   18.3  0.5      20.4
## 43                       Comoros     643571      8  0.1      32.4
## 44               Congo, Dem. Rep.   64256635   17.5 <NA>      44.9
## 45                    Congo, Rep.    3615152    5.3  3.5      34.5
## 46                    Costa Rica    4519126   26.1  0.3      16.7
## 47                 Cote d'Ivoire   20591302    4.6  3.7        35
## 48                       Croatia    4434000   17.6  0.1       9.9
## 49                          Cuba   11204735   15.5  0.1      10.5
```

```
## 50                       Cyprus     862434    5.8 <NA>      11.5
## 51               Czech Republic   10424336   13.3  0.1      11.5
## 52                      Denmark    5493621   15.3  0.2      11.8
## 53                     Djibouti     849245   15.3  2.6      28.4
## 54                     Dominica      73193   11.8 <NA>      <NA>
## 55           Dominican Republic    9952711   10.4  0.9      22.5
## 56                      Ecuador   13481424    6.9  0.4      20.8
## 57             Egypt, Arab Rep.   81527172    5.9  0.1      24.7
## 58                  El Salvador    6133910   11.9  0.8      20.2
## 59            Equatorial Guinea     659197      7  4.7        38
## 60                      Eritrea    4926877      3  0.8        37
## 61                      Estonia    1340675   11.9  1.2        12
## 62                     Ethiopia   80713434   11.5 <NA>      38.2
## 63                Faeroe Islands      48511   <NA> <NA>      <NA>
## 64                         Fiji     844046   10.2  0.1      20.9
## 65                      Finland    5313399   12.6  0.1      11.2
## 66                       France   62277432     16  0.4      12.9
## 67               French Polynesia     265702   <NA> <NA>        18
## 68                        Gabon    1448159    6.6  5.3      27.3
## 69                  Gambia, The    1660200   11.6  1.7      36.8
## 70                      Georgia    4307011    7.3  0.1      12.1
## 71                      Germany   82110097     18  0.1       8.3
## 72                        Ghana   23350927    8.5  1.8      32.4
## 73                    Gibraltar      31032   <NA> <NA>      <NA>
## 74                       Greece   11237094     13  0.1      10.5
## 75                    Greenland      56328   <NA> <NA>      14.8
## 76                      Grenada     103538   11.3 <NA>      19.4
## 77                         Guam     175552   <NA> <NA>      18.3
## 78                    Guatemala   13686128   15.9  0.8        33
## 79                       Guinea    9833055    4.3  1.4      39.6
## 80                Guinea-Bissau    1575446      4  2.5      41.2
## 81                       Guyana     763437   14.5  1.2      17.9
## 82                        Haiti    9876402    9.5    2      27.6
## 83                     Honduras    7318789   15.5  0.8      27.5
## 84           Hong Kong SAR, China    6977700   <NA> <NA>      11.3
## 85                      Hungary   10038188   10.2  0.1       9.9
## 86                      Iceland     317414   13.1  0.3      15.2
## 87                        India 1139964932    4.4  0.3      22.8
## 88                    Indonesia  227345082    6.2  0.2      18.6
## 89             Iran, Islamic Rep.   71956322    8.7  0.2      18.9
## 90                         Iraq   30711152    3.1 <NA>      31.2
## 91                      Ireland    4425675     16  0.2        17
## 92                    Isle of Man      80543   <NA> <NA>      <NA>
## 93                       Israel    7308800     10  0.2      21.5
## 94                        Italy   59832179   13.6  0.3       9.6
## 95                      Jamaica    2687200    5.7  1.7      16.7
## 96                        Japan  127704000   17.9  0.1       8.7
## 97                       Jordan    5812000   16.3 <NA>      25.7
## 98                   Kazakhstan   15674000    8.3  0.1      22.8
## 99                        Kenya   38765312    5.8  6.3      38.8
## 100                    Kiribati      96558   16.8 <NA>      <NA>
## 101              Korea, Dem. Rep.   23818753   <NA> <NA>      13.7
## 102                   Korea, Rep.   48607000   12.3  0.1       9.4
## 103                       Kosovo    1795000   <NA> <NA>        19
```

```
## 104                      Kuwait    2728040    6.1 <NA>      17.7
## 105           Kyrgyz Republic    5277900   11.5  0.2      24.4
## 106                   Lao PDR    6205341    3.7  0.2      27.3
## 107                    Latvia    2266094   10.2  0.6      10.6
## 108                   Lebanon    4193758   12.3  0.1      15.7
## 109                   Lesotho    2049429    8.2 23.6      28.9
## 110                   Liberia    3793400   17.2  1.6      38.3
## 111                     Libya    6294181    5.5 <NA>      23.3
## 112             Liechtenstein      35629   <NA> <NA>       9.9
## 113                 Lithuania    3358115   12.8  0.1      10.4
## 114                Luxembourg     488650   13.7  0.3      11.5
## 115          Macao SAR, China     526178   <NA> <NA>       8.2
## 116            Macedonia, FYR    2041342   13.6 <NA>      10.9
## 117                Madagascar   19110941   14.6  0.2      35.9
## 118                    Malawi   14846182   12.1 11.2      40.2
## 119                  Malaysia   27014337    6.9  0.5      20.4
## 120                  Maldives     305027   13.8  0.1      18.7
## 121                      Mali   12705736   11.1    1      42.6
## 122                     Malta     411950   12.3  0.1        10
## 123          Marshall Islands      59667   14.6 <NA>      <NA>
## 124                Mauritania    3215043    4.9  0.7      33.6
## 125                 Mauritius    1268854    8.3  0.9      12.9
## 126                   Mayotte     191187   <NA> <NA>      25.1
## 127                    Mexico  106350434     15  0.3      18.3
## 128      Micronesia, Fed. Sts.     110414   18.9 <NA>      25.3
## 129                   Moldova    3633369     13  0.4      12.3
## 130                    Monaco      32715   15.8 <NA>      <NA>
## 131                  Mongolia    2641216    7.5  0.1      18.8
## 132                Montenegro     622344   13.6 <NA>      12.1
## 133                   Morocco   31605616    6.6  0.1      20.4
## 134                Mozambique   22382533   12.6 11.4      39.2
## 135                   Myanmar   49563019    0.7  0.6      20.5
## 136                   Namibia    2129854   12.1 13.7      27.6
## 137                     Nepal   28809526   11.3  0.4      25.4
## 138               Netherlands   16445593   16.2  0.2      11.2
## 139       Netherlands Antilles     195253   <NA> <NA>      13.3
## 140              New Caledonia     246705   <NA> <NA>      16.2
## 141                New Zealand    4268900   18.3  0.1      15.2
## 142                  Nicaragua    5667325   18.7  0.2      24.6
## 143                      Niger   14704318   14.8  0.8      53.5
## 144                    Nigeria  151212254    6.4  3.6      39.8
## 145  Northern Mariana Islands      85364   <NA> <NA>      <NA>
## 146                     Norway    4768212   16.7  0.1      12.7
## 147                       Oman    2785361    4.9  0.1        22
## 148                   Pakistan  166111487    3.1  0.1      30.1
## 149                      Palau      20279   16.6 <NA>      <NA>
## 150                     Panama    3398823   13.5  0.9      20.6
## 151          Papua New Guinea    6576822    7.4  0.9      31.4
## 152                   Paraguay    6237855   12.3  0.3      24.6
## 153                       Peru   28836700   15.6  0.4      21.1
## 154                Philippines   90348437    6.1  0.1      24.7
## 155                     Poland   38125759   10.9  0.1      10.9
## 156                   Portugal   10622413   15.4  0.5       9.8
## 157                Puerto Rico    3954553   <NA> <NA>      11.8
```

```
## 158                         Qatar    1280862    6.8  0.1      12.1
## 159                       Romania   21513622   11.8  0.1      10.3
## 160            Russian Federation  141950000    9.2    1      12.1
## 161                        Rwanda    9720694   18.2  2.9      41.1
## 162                         Samoa     178869   14.9 <NA>      23.5
## 163                    San Marino      31031   13.6 <NA>        11
## 164          Sao Tome and Principe     160174   13.2 <NA>      32.1
## 165                  Saudi Arabia   24807000    8.4 <NA>      23.4
## 166                       Senegal   12211181   11.9  0.8      38.4
## 167                        Serbia    7350221   14.1  0.1       9.4
## 168                    Seychelles      86956   10.1 <NA>      17.8
## 169                  Sierra Leone    5559853    4.2  1.6      40.3
## 170                     Singapore    4839400    7.8  0.1      10.2
## 171                Slovak Republic    5406626   15.4  0.1      10.6
## 172                      Slovenia    2021316   12.9  0.1      10.8
## 173                Solomon Islands     510672   14.4 <NA>      30.4
## 174                       Somalia    8926326   <NA>  0.6      44.1
## 175                  South Africa   48793022   10.4 17.9        22
## 176                         Spain   45555716   15.2  0.4      11.4
## 177                     Sri Lanka   20156204    7.9  0.1      18.8
## 178           St. Kitts and Nevis      49189      8 <NA>      <NA>
## 179                     St. Lucia     170204   10.8 <NA>      <NA>
## 180 St. Vincent and the Grenadines     109117    9.5 <NA>      17.6
## 181                         Sudan   41347723    9.8    1      31.3
## 182                      Suriname     515124   13.6    1        19
## 183                     Swaziland    1167834    8.5 25.9      29.9
## 184                        Sweden    9219637   13.8  0.1      11.9
## 185                   Switzerland    7647675   19.9  0.4        10
## 186          Syrian Arab Republic   20581289    4.6 <NA>        28
## 187                    Tajikistan    6836083      5  0.2      28.1
## 188                      Tanzania   42483923     18  5.8      41.5
## 189                      Thailand   67386383   14.2  1.3      14.5
## 190                   Timor-Leste    1098386   11.9 <NA>        40
## 191                          Togo    6458605      8  3.2      32.9
## 192                         Tonga     103566   14.2 <NA>      27.7
## 193           Trinidad and Tobago    1333388    8.8  1.5      14.8
## 194                      Tunisia   10327800   10.4  0.1      17.7
## 195                       Turkey   73914260   12.8  0.1      18.2
## 196                   Turkmenistan    5043618      7 <NA>      21.9
## 197      Turks and Caicos Islands      32664   <NA> <NA>      <NA>
## 198                       Tuvalu       <NA>      8 <NA>      <NA>
## 199                       Uganda   31656865   10.5  6.4      46.2
## 200                      Ukraine   46258200    8.6  1.1        11
## 201          United Arab Emirates    4484935    8.9 <NA>        14
## 202                United Kingdom   61406928   15.1  0.2      12.9
## 203                 United States  304375000   18.7  0.6      14.3
## 204                       Uruguay    3334052   13.8  0.5      14.6
## 205                    Uzbekistan   27313700    8.6  0.1      21.7
## 206                       Vanuatu     233866   13.5 <NA>      30.2
## 207                  Venezuela, RB   27935000    8.4 <NA>      21.2
## 208                       Vietnam   86210781    9.3  0.4      17.2
## 209         Virgin Islands (U.S.)     109840   <NA> <NA>      12.3
## 210            West Bank and Gaza    3937309   <NA> <NA>      35.5
## 211                   Yemen, Rep.   22917485    4.3 <NA>      36.8
```

```
## 212                      Zambia   12620219   15.3 13.6      42.9
## 213                    Zimbabwe   12462879   <NA> 15.1      29.9
##     LifeExpectancy
## 1             43.9
## 2             76.6
## 3             72.4
## 4             <NA>
## 5             <NA>
## 6               47
## 7             <NA>
## 8             75.3
## 9             73.5
## 10            74.7
## 11            81.4
## 12            80.2
## 13            70.2
## 14            73.5
## 15            75.9
## 16            66.1
## 17              77
## 18            70.5
## 19            80.4
## 20            76.3
## 21            61.4
## 22              79
## 23            66.1
## 24            65.7
## 25            75.1
## 26            54.2
## 27            72.4
## 28            77.4
## 29              73
## 30              53
## 31            50.4
## 32              61
## 33            51.1
## 34              81
## 35              71
## 36            <NA>
## 37              47
## 38            48.7
## 39            79.1
## 40            78.6
## 41            73.1
## 42              73
## 43            65.3
## 44            47.6
## 45            53.6
## 46            78.9
## 47            57.4
## 48            75.9
## 49            78.7
## 50            79.7
## 51              77
```

```
## 52           78.4
## 53           55.4
## 54           <NA>
## 55           72.6
## 56           75.1
## 57           70.1
## 58           71.3
## 59           50.2
## 60           59.5
## 61           73.8
## 62           55.2
## 63           <NA>
## 64           68.9
## 65           79.6
## 66           80.9
## 67           74.5
## 68           60.4
## 69           55.9
## 70           71.5
## 71           79.7
## 72           56.6
## 73           <NA>
## 74           79.9
## 75           68.4
## 76           75.3
## 77           75.6
## 78           70.3
## 79           57.8
## 80           47.8
## 81           67.1
## 82           61.2
## 83           72.2
## 84           82.3
## 85           73.7
## 86           81.3
## 87           63.7
## 88           70.8
## 89           71.4
## 90           67.9
## 91           79.1
## 92           <NA>
## 93             81
## 94           81.2
## 95           71.8
## 96           82.6
## 97           72.7
## 98             67
## 99           54.2
## 100          <NA>
## 101          67.2
## 102          79.8
## 103          69.4
## 104            78
## 105          67.4
```

```
## 106            65
## 107          72.4
## 108            72
## 109            45
## 110          58.3
## 111          74.3
## 112          82.6
## 113          71.8
## 114          80.1
## 115          80.7
## 116          74.2
## 117          60.3
## 118          53.1
## 119          74.4
## 120          71.6
## 121          48.4
## 122          79.4
## 123          <NA>
## 124          56.7
## 125          72.6
## 126          76.1
## 127          75.1
## 128          68.6
## 129          68.4
## 130          <NA>
## 131          66.6
## 132          74.1
## 133          71.3
## 134          47.9
## 135          61.6
## 136            61
## 137          66.7
## 138          80.3
## 139            76
## 140          76.1
## 141          80.4
## 142          73.1
## 143          51.4
## 144          47.9
## 145          <NA>
## 146          80.6
## 147          75.9
## 148          66.5
## 149          <NA>
## 150          75.7
## 151          61.1
## 152          71.9
## 153          73.3
## 154          71.8
## 155          75.5
## 156          78.5
## 157          78.6
## 158          75.9
## 159          72.6
```

```
## 160          67.8
## 161          50.1
## 162          71.8
## 163          82.8
## 164          65.5
## 165          73.1
## 166          55.6
## 167          73.6
## 168          73.2
## 169          47.6
## 170          80.7
## 171          74.7
## 172          78.8
## 173          66.3
## 174          49.8
## 175          51.5
## 176          81.9
## 177          74.1
## 178          <NA>
## 179          <NA>
## 180          71.7
## 181          58.1
## 182            69
## 183          45.8
## 184          81.1
## 185            82
## 186          74.2
## 187          66.7
## 188          55.6
## 189          68.9
## 190          61.1
## 191          62.5
## 192          71.9
## 193          69.3
## 194          74.3
## 195          71.9
## 196          64.8
## 197          <NA>
## 198          <NA>
## 199          52.7
## 200          68.3
## 201          77.7
## 202          79.6
## 203          78.4
## 204            76
## 205          67.8
## 206          70.3
## 207          73.5
## 208          74.4
## 209          78.9
## 210          73.5
## 211          62.9
## 212          45.4
## 213          44.2
```

- some of the data are missing. For example, the variable **`Developed`** *classifies countries into categories 1, 2, or 3. (What are they?) But not all countries are included. Dominica, for example, does not have a classification. Construct a dataframe that includes only countries that are classified and also where we know the percentage of the elderly population.* Ans: *Developed* ranks the countries based on their development from 1 to 3 where 1 is low development and 3 is developed country.

```
developed_elderly <- filter(country, !is.na(country$Developed) & !is.na(country$ElderlyPop))
head(developed_elderly)
```

```
## # A tibble: 6 x 14
##   Country   LandArea Population Energy Rural Military Health   HIV Internet
##   <chr>        <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
## 1 Albania      27400    3143291   2088  53.3     NA      8.2  NA       23.9
## 2 Algeria    2381740   34373426  37069  34.8     13     10.6   0.1     10.2
## 3 Angola     1246700   18020668  10972  43.3     NA      6.8   2        3.1
## 4 Argentina  2736690   39882980  76359   8       NA     13.7   0.5     28.1
## 5 Armenia      28480    3077087   2997  36.1     16.1    7.2   0.1      6.2
## 6 Australia  7682300   21431800 130113  11.3      7.7   17.1   0.1     70.8
## # i 5 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>, density <dbl>
```

- continuation of the previous exercise. Compare the average percent of elderly population among the three different types of countries.

```
developed_elderly %>% group_by(Developed) %>% summarise(mean=mean(ElderlyPop))
```

```
## # A tibble: 3 x 2
##   Developed  mean
##       <dbl> <dbl>
## 1         1  5.53
## 2         2 11.4
## 3         3 12.7
```

- add a new column which essentially replaces **`1`** *in the* **`Developed`** *column by "least developed",* **`2`** *by "developing', and* **`3`** *by"developed".*

```
developed_elderly_n <- developed_elderly %>% mutate(dev = case_when(
  Developed == 1 ~ "least developed",
  Developed == 2 ~ "developing",
  Developed == 3 ~ "developed",
))
developed_elderly_n
```

```
## # A tibble: 134 x 15
##    Country    LandArea Population Energy Rural Military Health   HIV Internet
##    <chr>         <dbl>      <dbl>  <dbl> <dbl>    <dbl>  <dbl> <dbl>    <dbl>
##  1 Albania       27400    3143291   2088  53.3     NA      8.2  NA       23.9
##  2 Algeria     2381740   34373426  37069  34.8     13     10.6   0.1     10.2
##  3 Angola      1246700   18020668  10972  43.3     NA      6.8   2        3.1
##  4 Argentina   2736690   39882980  76359   8       NA     13.7   0.5     28.1
##  5 Armenia       28480    3077087   2997  36.1     16.1    7.2   0.1      6.2
##  6 Australia   7682300   21431800 130113  11.3      7.7   17.1   0.1     70.8
```

```
##  7 Austria       82450    8336926  33246  32.8      2.4   15.8   0.3     72.9
##  8 Azerbaijan    82620    8680100  13367  48.1     22.9    2.5   0.1     28.2
##  9 Bahrain         760     775585   9226  11.5     15.6   10.3  NA       51.9
## 10 Bangladesh   130170  160000128  27944  72.9     10.8    7.4   0.1      0.3
## # i 124 more rows
## # i 6 more variables: Developed <dbl>, BirthRate <dbl>, ElderlyPop <dbl>,
## #   LifeExpectancy <dbl>, density <dbl>, dev <chr>
```