BERMMC010 Programming - Assignment 3

In this assignment we will be scraping the CIA World Factbook. Some code, including XPath query info, is provided to get you up to speed. The focus will be on generalizing your code and using functions with iteration.
In most cases you can answer the question by providing the code that will give the answer. If a textual answer is required, add it as a comment to the code. You will be graded on correctness as well as proper style (check https://style.tidyverse.org/).
Development hint: Try your functions on a selection of URLs first, instead of on all of them. Downloading the information for all countries will take some time and this will slow down your development process. Only test on all of them when it works for, say, 10 countries or so.

Part 1 – Scaffolding

1. Download the assignment-3.R file from Canvas, put it in the same project as before. If you have RStudio/Git problems, you are free to start a new project too.

2. Create a new branch called **assignment-3** specifically for this assignment and work on that branch for the remainder of the assignment. Commit and push the **assignment-3.R** file into this new branch.

3. While doing the assignment, try to commit every time you do a question from the assignment. It's not the end of the world if you forget sometimes, but it is good to practice with small and regular commits.

4. Each question will ask you to create a function. Add those functions to the ones already in the file.

5. Use RStudio's option to insert Roxygen comments to add documentation to each function, or update it when already provided.

Part 2 – Web Scraping

1. Create a function called **get_population_ranking** which scrapes the ranking from `fields/335rank.html` (add it to the given base_url).
    a. Extract the 4 elements for each country, given the 4 xpath queries that are provided to you
    b. Rename the "value" column to "population"
    c. Rename the "rank" column to "rank.population"
    d. Remove the "../" from all urls so they can be added to the base_url

2. Create a function called **get_land_area** with country_link as a parameter, which will visit each country_link (what you got from the previous question) and which will extract the land area element from the page using the given XPath query.

   a. You do not have to clean or transform the output (yet).
   b. Your function should work when country_link is a single url, but also when it is a character vector with multiple urls (iterate over the urls).
   c. The output should be a character vector of the same length as the country_link parameter.

3. Create a function called **get_population_density** that will use the land area and population information acquired using the previous two functions and compute the population density for each country.
   a. Retrieve the population ranking using **get_population_ranking**
   b. Retrieve for each country the `land_area` from the `country_link` url using **get_land_area**
   c. Transform population and land area into a number (be careful with Ethiopia)
   d. Add a column called `population_density` as `population / land_area`.

4. Create a function called **get_rankings** that will scrape the overview of all the available rankings.
   a. Use the two provided XPath queries to create a `characteristic` and a `characteristic_link` column.
   b. Remove the "../" from the `characteristic_link` column
   c. Remove the ":" from the `characteristic` column and put it in lowercase

Part 3 – Generalization

5. Create a **get_ranking** and **get_country_characteristic** function that are generalizations of the **get_population_ranking** and **get_land_area** functions.
   a. For **get_ranking**, renaming a column with a value from a variable can be done using special syntax, see: https://stackoverflow.com/questions/45472480/how-to-rename-a-column-to-a-variable-name-in-a-tidyverse-way#45472543
   b. Note that the extra parameters have default values such that calling them without specifying any of the extra parameters should result in the same output as the **get_population_ranking** and **get_land_area** functions, respectively.

6. Create a **combine_rankings** function which will use the generalized **get_ranking** function to download multiple rankings.
   a. Use as input the rankings from **get_rankings** (both url and characteristic name)
   b. Combine (or 'roll-up') all rankings into a single data frame using `full_join`

Part 4 – Submission

Find the URL of your final commit in the **assignment-3** branch on GitHub. Submit this URL on Canvas.