CMU MSP 36602 R Web Scraping Homework Due 10 PM Sunday Feb 3

1) **Function UNGDP() from template file UNGDP.R**

The goal of this function is to extract GDP data for one or several countries for one or several years. Follow all instruction in the template. Remember to use a standard style and appropriate documentation (you will need little beyond what is in the template).

Work alone and ask Howard for help as needed.

Requirements:

a) You must use standard R **without any special scraping packages such as "rvest"**. You may use dplyr, etc., but they are not needed. The only package you need is one for reading Excel files (see below).

b) We assume that the web page https://unstats.un.org/unsd/snaama/Downloads exists and has a link named "All countries for all years - sorted alphabetically" beneath the text "GDP and its breakdown at current prices in US Dollars", and that the link holds an Excel file with columns including "Country", "IndicatorName", and "Xyyyy" for each year "yyyy". We also assume that one of the IndicatorName values is "Gross Domestic Product (GDP)".

   We do **not** assume that the name of the Excel file is fixed; instead scrape it from the above URL. Note that there are several links labeled "All countries for all years - sorted alphabetically", so be sure to pull of the one under "current prices in US Dollars". Arrange so that your function stops with a friendly message if the web page has changed in a way that you can no longer scrape it, e.g., the link text is changed.

c) The function should attempt to load any required packages and `stop()` with a friendly message if any are not installed. Use `require()` for this.

d) The function should give a friendly `stop()` message if the input is not appropriate or if there are no country/year combinations matching the input. Do not assume that the years currently found in the Excel file will always be the years available.

e) The result must be a plain `data.frame` with columns named "Country", "Year", and "GDP", where the GDP is rounded to millions of US dollars.

f) You will need to load the Excel file (and give it a .xlsx extension) from the web into a temporary local file. I suggest `download.file()`, which is in the pre-loaded "utils" package. You must use a temporary file name and remove the file when you are done with it. See `tempfile()` and `file.remove()`.

g) There are several mechanisms for reading an Excel file. I suggest `read.xls()` in the commonly used "gdata" library. Note that this will require you to load Perl, if it is not already on your computer, but the installation is easy. The use of the "XLConnect" or "xlsx" packages are no longer suggested because they require Java (not JavaScript). If you are a tidyverse person, you can use the "readxl" package. Be sure to manually examine the Excel file before trying to read it, and note that there are two lines before the real data, one of which is blank.

h) Once you have pared the data.frame down to the correct rows and columns, you need to transform it from a wide format to a tall format. I suggest the built-in function `reshape()`. It will be easier to use `reshape()` if you rename your columns from, e.g., "X2001" and "X2002", to "GDP.2001" and "GDP.2002". If you want to use the tidyverse to reshape, that is fine, but the final return value must be a plain data.frame, not a tibble.

2) Function pieFaq() from the template file pieFaq.R

The goal of this function is to extract FAQ's for the Pie For Breakfast restaurant **using the "rvest" package**. Follow all instructions the template. Remember to use a standard style and appropriate documentation (you will need little beyond what is in the template).

Work alone and ask Howard for help as needed.

Requirements:

a) We assume that the http://www.pieforbreakfastpittsburgh.com/ website has the same structure it had on 1/24/2019.

b) Be sure to read the website only once and store the scraped info in a local file called FAQ.csv. Read that local file for all subsequent calls to the function.

c) Assume the hours of operation are text inside an "h2" tag (not the only one), and that it contains the word "open".

d) Note the `sapply()` and `lapply()` work on nodesets, e.g., `sapply(myNodeset, html_text)`.

e) It is fairly easy to pull out a nodeset that contains one FAQ per node plus some extraneous nodes. The next step, to restrict the nodeset to FAQs only, is not obvious. One possible step is to use `html_children()` on each node and check the `html_name()` of the children. The real FAQ nodes have two children, and the first is a "br" tag (to separate the question from the answer on the web page).

f) Unfortunately, the structure of the FAQs and the available functions in rvest do easily and robustly pull out the questions vs. the answers. You can just use html_text() to get the questions and answers pasted together and assume that the first question mark defines the end of the text and the beginning of the answer.