# Project 10: Geography, and the World Wide Web

## Clarifications/Corrections

- Apr 22: minor typos in q37 and q38
- Apr 23: for those of you getting the dreaded SSLCertVerificationError, see the fix here: https://piazza.com/class/jqa9nnsyjx778j?cid=682
- Apr 23: `get_json` directions now says "Python structure" instead of "Python dict"
- Apr 24: relax how close answers need to be to expected to count them as correct
- Apr 24: fix wording on q21 (should be "least first", not "most first")
- Apr 24: q39 refers to q38

## Introduction

For your final CS 301 project, you're going to analyze the whole world!

Specifically, you're going to study various statistics for 175 countries, answering questions such as: *what is the correlation between a country's literacy rate and GDP?*

To start, download `test.py` and `expected.html`. Do not download any data files manually (you must write Python code to download these automatically). You'll do all your work in a `main.ipynb`.

# Data

For this project, you'll be using one large JSON file with statistics about 175 countries, and 9 small JSON files, each with details about the capitals for a subset of those 175 countries. The data was adapted from
here and
here.

You should start by looking at these two web resources:

- https://tyler.caraza-harter.com/cs301/spring19/data/countries.json
- https://tyler.caraza-harter.com/cs301/spring19/data/capitals/manifest.txt

The second file looks like this:

```
Brazil_Peru.json
ElSalvador_SaintHelena.json
Ghana_Belize.json
Palau_Nigeria.json
Portugal_Somalia.json
Seychelles_Cuba.json
Slovenia_Estonia.json
Tanzania_Swaziland.json
Vanuatu_Tajikistan.json
```

These are file names you can use to construct URLs for capital data files (e.g., the first capital JSON file is here: https://tyler.caraza-harter.com/cs301/spring19/data/capitals/Brazil_Peru.json).

Some of the columns require a little extra explanation:

- area: measured in square miles
- coastline: ratio of coast to area
- birth-rate: births per 1000 people per year
- death-rate: deaths per 1000 people per year
- infant-mortality: per 1000 births
- literacy: (out of 100%)
- phones: number of phone per 1000 people

# Testing

For answers involving a DataFrame, `test.py` compares your tables to those in `expected.html`, so take a moment to open that file in your browser.

`test.py` doesn't care if you have extra rows or columns, and it doesn't care about the order of the rows or columns. However, you must have the correct values at each index/column location shown in `expected.html`.

For P10, `test.py` is pickier than it has been. In addition to checking for incorrect answers, it will also check for a few common kinds of bad coding style. You should look for these at the bottom of the output:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
AUTO-GENERATED CODING SUGGESTIONS FOR YOUR CONSIDERATION:
q3: The function g has more than one definition. Avoid reusing function names
q39: Do not use 'list' for a variable name; that clobbers the Python type.
q40: Consider using more descriptive variable names. You used : 'm'.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

In some cases, `test.py` will deduct points for bad style. In other cases, you should consider whether the auto-generated tips apply to you or not. For example, consider this one:

```
q40: Consider using more descriptive variable names. You used : 'm'.
```

It's not good to have too many one-letter variables (it often makes it hard to read the code), but some are OK.  It's on you to look at your code and decide whether or not it's best to use a longer variable name in such cases.

## The Stages

- Stage 1: scrape some data files and answer some geography questions
- Stage 2: query a DB and generate some plots