**NJ/PA Town Populations and Incomes**

Methodology

**Purpose:**

Retrieve raw economic and demographic data from public sources for visualising the business environment around Washington, NJ.

Notes:

Retrieved data will be sortable by county to enable a focused analysis

Requires the use of geopy for plotting on a map.

There are eight parts to this task:

1. Geo data and incomes for NJ cities
2. Geo data and incomes for NJ towns
3. Geo data and incomes for PA cities
4. Geo data and incomes for PA towns
5. Geo data and populations for NJ cities
6. Geo data and populations from NJ towns
7. Geo data and populations for PA cities
8. Geo data and populations for PA towns

NOTE: income and population source data is different so geo data is not sharable between population and income categories

**Steps**

Copy raw data from Wikipedia and paste into Excel

Use Excel Right() and Left() functions to separate town names from index and incomes

Use Excel Right() function to separate incomes from index and town names

Standardise data – make column heading names and order consistent; make data formats consistent

NOTE: County names are not included in raw data for smaller towns. County data is needed for geocoding reliability (and getting the latitude and longitude data for plotting the circles

Transform small town data ready for processing:

1. For retrieval of County
2. For performing geocoding
   1. Includes steps such as copying/pasting-as-values-only, concatenation to append State to end of name

Paste-as-values transformed small town, state data into csv files (for reading into Python pandas DataFrame for geocoding):

* Small NJ towns incomes
* Small PA towns populations
* Small PA towns incomes

NOTE: population, income data not required for county-finding step

Python Coding:

Import relevant libraries

Import CSV file into pandas DataFrame

Iterate through town names to get geocode for each town – assign to list ‘locations’

NOTE: geocode retrieval is prone to timing out on big queries. It may take numerous repeats to get through the whole list.

* My code includes a 2 second delay in each query in an effort to reduce time-outs.
* My code includes a print statement in each step to track where the time-out occurred and where to restart from.

Add locations as column ‘Location’ to DataFrame

Create new DF by dropping NAs from old DataFrame

Split Locations geopy objects into lists

Obtain County by using “if “County” in…” (NOTE: some flitting back and forth between csv may be needed in order to find gaps and ensure correct list lengths)

Export new DF as csv file