As part of an earlier course on "Algorithmic Thinking", we created infrastructure that allowed students to plot cancer-risk due to air toxics on a county-level map of the United States. The practice projects for the last two weeks of the previous course in this specialization focused on reading and processing the cancer-risk data given in a CSV file. The four practice projects in this course will focus on the task of plotting this cancer-risk data on a county-level map of the USA.

The four practice projects corresponding to this task are as follows:

- Install the popular Python drawing package matplotlib (and its dependent packages). Then draw a county-level image of USA using matplotlib.
- Parse an SVG image of the USA and its county boundaries and compute the center of each county with respect to its position in the image. Output a unique identifier for each county (a FIPS code) and its corresponding center to a CSV file.
- Reconcile the FIPS codes and county centers in this CSV file with the FIPS codes and cancer-risk data in the CSV file created in the previous course.
- Draw the cancer-risk data for each county on the USA map using the computed county centers and create an image similar to the one linked here.

Practice Project: Drawing a USA Map in matplotlib

For the practice projects in this final course, we will work with the Python plotting package matplotlib to create a plot of our cancer-risk data on a map of the USA. matplotlib is widely recognized as one of the most popular and powerful drawing packages in Python. However, matplotlib has two drawbacks for novice Python scripters. First, installing matplotlib is complicated by the fact that it depends on several other Python packages that must also be installed. Second, matplotlib is more complex that most of plotting packages making it somewhat intimidating for novices. As a result, we have opted to use pygal for the required projects in this class and matplotlib for the practice projects. Having experience with both packages will help you begin to the understand the pros and cons of the many Python packages available for plotting.

Install matplotlib and its dependent packages

Your first task for this project is to install **`matplotlib`** and the five packages upon which it depends: **`numpy`**, **`python-dateutil`**, **`pytz`**, **`pyparsing`**, and **`six`**. All six of these packages are available in the Python Package Index. We recommend installing these packages from this index via one of two options:

- For those of you using Thonny, we recommend using the "Manage Packages" option in the "Tools" menu. This functionality is discussed in detail in this week's programming tip. Note that these

packages are installed in Thonny's own private instance of Python and are not available in other Python environments that you may have installed.

- If you are not using Thonny, we recommend installing these packages via the command line using pip. You are welcome to skip ahead to next week's programming tip which goes over this process in detail.

Draw a county-level map of the USA in matplotlib

Our next task is to locate a map that includes the boundaries of the counties of the USA. Based on some Google searching, this map from wikimedia.org is a suitable candidate. The original map is stored in SVG format (scalable vector graphics) in which each line in the map represented in terms of precise mathematical coordinates. The advantage of this vector representation is that the map can be drawn as a bit-mapped (pixel-based) image of any resolution without a loss of accuracy.

We recommend that you download the 555x352 resolution SVG version of this image as well as the 555x352 resolution and 1000x634 resolution PNG (bitmap) versions of this image. Your matplotlib code will draw these PNG images since matplotlib does not support drawing of SVG images. **Note that it is critical that you use our versions of these assets** (saved from 2014 when we developed this project for our "Algorithmic Thinking" course) since Wikimedia updates these assets fairly frequently. In particular, the current version of the SVG map has a slightly different format that the provided 2014 version.

Next, review the documentation for the pyplot module in matplotlib and experiment with some of its example code. Then, starting from this provided code, add **`matplotlib`** code to draw the PNG version of the USA map. Our solution includes calls to the following **`pyplot`** methods: **`imread()`**, **`imshow()`**, and **`show()`**.

Experiment with drawing scatter points in the map's coordinate system

Once you have successfully drawn the PNG version of the USA map in **`matplotlib`**, we suggest that you conclude this week's activities by experimenting with drawing scattered data (like the county-level cancer risk data) overlaid on top of the USA map. In **`pyplot`**, one simple technique for drawing scattered points is the **`scatter()`** method. We suggest that you experiment with drawing scattered points of varying position, size, and colors on the maps of varying resolution.

Our solution code for this project includes code that draws a scatter point at the center of the map as well as over RIce University in Houston, Texas. One interesting feature of our solution is that the coordinates of RIce on the 555×352 resolution map are automatically rescaled to the appropriate location on the 1000×634 resolution map.

Mark as completed