I asked you to bring your own machine this week, and to do a pre-activity to set your computer up to run both Python and Komodo Edit. This was designed to make your life easier, so I hope you heeded that warning. If you have not done that and you have your computer with you, you can find the tutorial for setting up Python at the Programming Historian:

http://programminghistorian.org/lessons/introduction-and-installation

Choose either the Mac, Linux, or Windows lesson, depending on your machine. You can stop once you've successfully run a 'hello world' programme in Komodo Edit. If you see 'Hello World' written in your command output pane, then you have succeeded. This should take no more than 20 minutes total.

Workshop: Extracting Places from Free-Flowing Text

There are 6,690 Oxford graduates listed who began their studies during the reign of King James I (1603-1625). A copy of these entries can be downloaded from the module website under this week's workshop.

Most (but not all) of their bibliographic details provide enough information to tell us what county they came from. However, the information is not always available in the same format. Sometimes it's the first thing mentioned in an entry. Sometimes it's in the middle. Your challenge is to extract those counties of origin from within this messy text, and store it in a new column next to that person's entry.

This task is probably too big for you to do on your own in the time we have. So you will need to work together as a group. If all 9 of you are there, you could split the work up and do it manually. You'd have to do about 740 entries each. In an hour, that would be about one every 5 seconds.

But let's see if we can find another way so that we could do 6,000 or 60,000 without much extra effort. There are a million ways we could do this. Those of you who did the OpenRefine tutorial might think of turning to that tool as an option. We're going to take an approach that uses Python, which we played with a few weeks ago. This approach should give us more control than we had with Google Fusion, and we shouldn't have a 6% 'ambiguous' rate like we did with the automated tool. If you can master this ability then you can generalise the skill and use it for anything: places, dates, names, adjectives that rhyme with 'puppy', etc.

Step 1 – Building a Gazetteer (a list of keywords)

You'll need some tools to be able to pull this off. Firstly, you'll need to know what you're looking for. In this case, it's counties. Most of these people came from England or Wales. So, as a group you'll have to find a historic list of counties from the 17th century. I trust your Googling skills mean you won't find that too

difficult. Don't forget 'London' and 'Bristol', which aren't counties, but which appear a lot in the dataset.

Split up the task and work concurrently to fill in the shared Google Drive spreadsheet that I've set up:

https://docs.google.com/spreadsheets/d/1bspN2z7uKN5fqTfQugdP9GXh7n6CytEHm_ygU5SlZGU/edit?usp=sharing

This should only take you a couple of minutes if you spread the work around in a logical fashion. These will be our keywords for searching each of the entries.

Step 2 – Refining our Gazetteer

If you take a look at the entries, you'll notice that many of the county references are short-forms (see row 25, George Bell). We're going to be looking for matches between your list and the entries in the biographies. That means that we need to be able to match these short forms as well. As a group, on the spreadsheet, start filling in short forms for each county. You can use short forms you know, or ones you think might be logical. For example, I've included Herts, Hert, and Hertford for Hertfordshire. Most counties have some if not many possible variations. You can work collectively on this spreadsheet until you think you've got a good list.

My list had 157 entries on it, in order to get every mention of a county. So use that as a guideline, but don't worry if you can't think of them all right away. We'll have a chance to refine as we go.

Don't forget tricky ones like "Sarum" and "Salop".

Step 3 – Downloading the Python Programme

Now that you've got a list of the keywords we're after, we'll create a short Python programme that will search each entry for a match.

I don't expect you to be able to do this yourself. Instead, I've created one for you that I hope will work.

You can find it on Github, a website for sharing code:

https://gist.github.com/acrymble/b104e854248519ddf85e

Take a look at the comments (they start with #).

The programme I've written for you effectively does 6 things:

1) Loads a list of keywords that you've created
2) Loads your text, which you have to supply
3) Then for each biographical entry, it removes the unwanted punctuation
4) It then checks for the presence of one of the keywords from your list

5) If it finds a match, it stores it while it checks for other matches
6) Finally, it prints the results out for you.

If you're interested in learning more about Python, you can spend some time seeing if you can figure out the code line-by-line.

For now you need to know how to use it.

Step 4 – Loading in the Gazetteer and the Texts

4-a the Gazetteer:

For this Python programme to work, you need to give the programme a copy of your list of keywords and a copy of the texts you want it to search. To do this, create a folder on your desktop and give it a name without any spaces. "Mapping", for example.

Open a text editor, and cut and paste all of the keywords from the Google spreadsheet into your text editor so that each keyword is on its own line. You will have to do this one column at a time. There should be no blank lines. Save this file as 'keywords.txt' and put it in your new folder that you've created.

If you ever need to add to this set of keywords, you can open this file in your text editor and add new words, each on their own line. Komodo Edit is a good text editor for those of you who have installed it.

4-b the Texts:

You also need to give the programme a copy of the texts, again one on each line. The easiest way to do this is to open the spreadsheet containing the data and copying all of the text in the 'Details' column. Paste it into a new text document (not MS Word) and save it as texts.txt, also in your new folder.

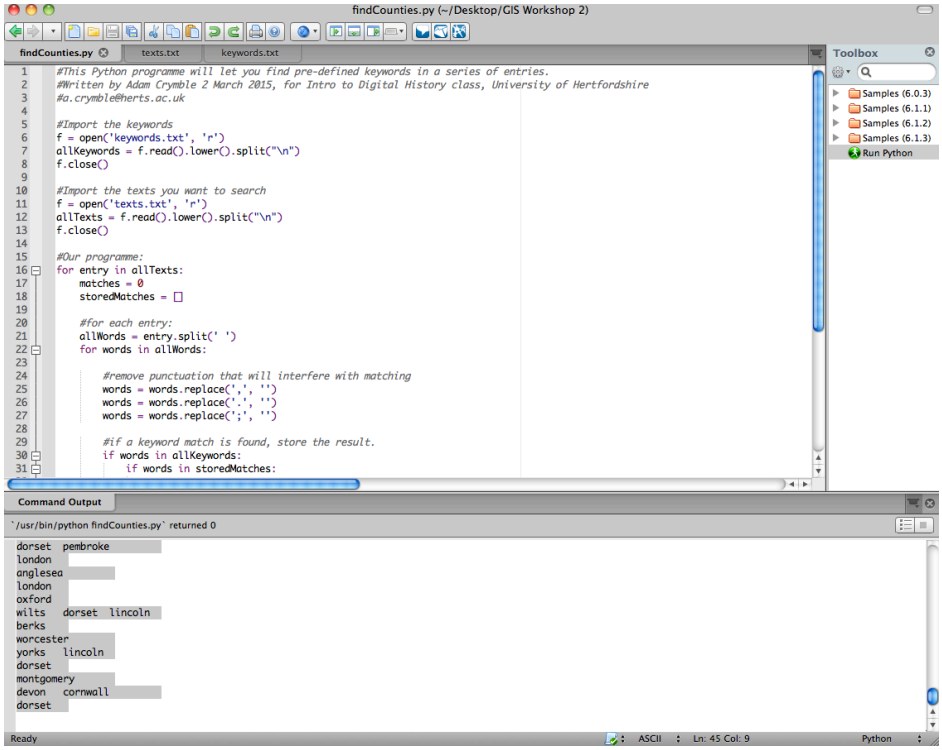
4-c the Programme:

Copy the programme from Github (https://gist.github.com/acrymble/b104e854248519ddf85e) and save it in a new text file (not MS Word) and save it to the same folder as findCounties.py

You now have all of the files you need. The next step is to run the programme.

Step 5 – Running the Programme

If you have set up your computer to use Komodo Edit, you can run the Python programme by opening it in Komodo Edit and clicking on the 'Run Python' button you created, which you can see here under the 'Toolbox' on the right.



If you cannot see either the Toolbox (right) or the Command Output pane (bottom), then you can open these via the 'View' -> 'Tabs & Sidebars' menu.

Double-clicking on the 'Run Python' button will run the code and give you the output in the Command Output pane.

Before you run the code, make sure that there are not any asterisks (*) showing on the tabs at the top of the screen (you may only have one tab, called findCounties.py). If there is an asterisk, that means the file has been changed since it has been saved, and the code will run on the OLD version. Meaning any changes you made will not take effect.

If you mess up the code, simply go back to Github and get a fresh copy and start over again.

Once the programme has finished running, copy all of the text from the Command Output pane and paste it into the original spreadsheet in the first cell of Column F. Double check that the first few entries have lined up properly. You should now have a lot of county names next to your entries!

If that worked, skip ahead to the next step.

If you do NOT have Komodo Edit on your machine, you will have to use the Python command line, which is probably called 'Idle'. It should be installed on the machines in our room.

Right click on the python file in the new folder that you created. There should be an option 'Open With'. Select 'IDLE'. This should open the programme in a very plain looking window. You can now run the code by selecting the 'Run' menu and choosing 'Run'. Or you can press F5. This should then open up another window and counties will start ticking by your screen.

It will take considerably longer to run this way than it does in Komodo Edit. But once it has finished running, select all of the text that has appeared in the output window, and copy and paste it into your spreadsheet of the original data, in Column F.

Make sure that the first few entries have lined up properly. You should now have a lot of county names next to your entries!

Step 6 – Refine your Gazetteer

You've probably noticed that a lot of entries got missed. Start going through the spreadsheet manually and look for spellings of county names or abbreviations that you've missed. Add these to the group's Google spreadsheet, and work together to build up a list of the missing entries.

It may help to know that you can find the next empty cell in a column in Excel by pressing CTRL + down arrow (CMD + down arrow on Mac).

You may also find it helpful to use the Sort feature in Excel to sort the whole spreadsheet by the F column, which will make it easier for you to find all of the entries that you've missed. If you do this, make sure you sort back by the 'Original Order' column before you paste in any refined results. Otherwise you will paste the wrong entry next to the wrong cells.

Once you've got some more entries for your gazetteer, remake your 'keywords.txt' file to include all of the new words. Remember, I got about 150.

Save the file and re-run your code to get the updated results. Keep updating the keywords list until you've all got as many entries as can possibly be extracted. There are a few that just have no place name in them, so you can ignore those.

Step 7 – Figure out which one is the place of origin

As I'm sure you've noticed, many entries have mention of more than one county. You can use the Sort features on Excel to sort the whole spreadsheet by Column F and then G (the first two columns of results). That should help you isolate all of the ones for which you only have 1 value (which we can presume are correct), leaving you with the remaining entries that have more than 1 value.

At this point, you might consider writing another Python programme to help you refine this further, but it's probably easiest to start in manually correcting these results, deleting matches that refer to something other than the place of origin. Have a go at getting this list refined.

A tip for getting started is to sort all of these multi-destination entries by the ‘details’ column. If you look at the entries beginning with ‘Of’, you’ll note that most of them refer to the place of origin. That should make it easy to strike off quite a large number of false positive matches so that you can focus on the tougher cases.

Not everything in digital history can yet be solved by computers, but if you’ve got this far, you’re a long way towards a nice clean set of geographic data. It’s not as instantaneous as the Google Fusion option, but it gives you control, which is important for a scholar.

Work with your classmates to see if you can find other patterns that will make it easier to refine the remaining entries.

Good luck!