**GEOG 489 – Lesson 1: Part 2 Python coding and profiling**

**By Mark A. Prettyman**

(NOTE: MAP in file names are my initials and not in reference to a map/document)

Deliverables:

* *Modify the code to handle a parameterized output path (still using unique output filenames for each shapefile)* 
  + See modified Python code. (scripttool\_MAP.py & multicode\_MAP.py)
* *Create a tool for running the modified code* 
  + See ArcGIS Pro Toolbox. (ScriptTool “489 Lesson1 ScriptTool MAP Parameterized Output” in 489\_Lesson1HW.tbx)
* *Implement code profiling and run simple (using the time function) and also the more advanced module (no visual or line profiling needed) and report on any bottlenecks in the code. For above and beyond points modify the code to make it more efficient.* 
  + After modifying the two scripts to incorporate a parameterized output path and creating a tool to run it in ArcGIS Pro, I first added simple code profiling by adding the time function to the functional script tool. In addition to simply having the tool output specify how long it took to complete the script tool, I also added a few lines of code after some of the existing AddMessage statements to help gauge what part of the script took the longest (which I have removed in the final scripts). The overall script to clip and create the 59 shapefiles took about 18 to 19 seconds. However, just after the script counts the number of processor cores and just before it begins performing the clipping for each of the 59 shapefiles, I added another line of code to return the time, which was about 0.04 seconds. This alone leads me to believe that the steps in the doWork function (which materially clips and exports the shapefiles) are what is taking the longest to complete the script/tool.
  + In running the script through the profiler, it once again completes in about 18-19 seconds, and creates the 59 shapefiles. To me, nothing looks like a bottleneck. As I alluded to in my post on Piazza, if there is a potential bottleneck that I should be able to see based upon the times, I am not seeing it. In an effort to edit the script to be more efficient, I commented out all of the AddMessage and print statements except for the very last one to specify the overall time, which reduced the overall time to about 16-17 seconds. All of the functions and modules seem to run in milliseconds or microseconds, although I did find a line <built-in method \_imp.create.dynamic> under “\_call\_with\_frames\_removed” under “\_find\_and\_load\_unlocked” “under “\_find\_and\_load” which had a Local Time of 2.41 seconds (which I believe is the longest time listed in Local Time). I do see that the main > multi function was called at 16.10 seconds in the Total Time, vs a time of 4.19 seconds for the \_find\_and\_load function, which leads me to believe that the bulk of the time for the script was spent prior to calling the multi function. The final scripts (referenced above) do not have any additional print statements, and the original print (or AddMessage) statements have all been commented out.
* *Expand the code so that it can handle multiple input shapefiles to be clipped (still using a single polygon clipping shapefile) and run them in multiprocessing (do this after the profiling stage - and you do not need to rerun the profiling).* 
  + See modified Python Code (multi\_file\_scripttool\_MAP.py & multi\_file\_multicode\_MAP.py) and ArcGIS Pro Toolbox. (ScriptTool “489 Lesson1 ScriptTool MAP Multi Input” in 489\_Lesson1HW.tbx)
* *Above and beyond points may be earned by uploading your write-up and/or profiling results only (not the code) to github, excellent documentation, in-tool documentation or other enhancements as you see fit.*
  + This file shall be uploaded to my GitHub account. (<https://github.com/map579/GEOG489>)
  + Each script tool includes documentation (included via editing metatdata).
  + Multi input script(s) include code to add a folder for each set of clipped shapefiles to be exported to a subfolder (to the primary output folder chose) with the name of the shapefile being clipped.
* *A 300 word write up of what you have learned during this exercise, reflect on and briefly discuss any changes to the processing workflow and/or the code that might be necessary if our input data were in geodatabases or if we wanted to write our output data to geodatabases and briefly comment on possible issues (using pseudocode or a simple flowchart if you wish).*
  + In this exercise I have learned that syntax will once again be a very challenging aspect of Python coding for myself. While I can understand what is being discussed in the Lesson plans, ensuring that just the right syntax is being used in a script is proving a little difficult. For example, I struggled with determining exactly where I should put the “for loop” in the code modifying the code for multiple input files. I believe I was over-thinking the problem, and thankfully Jim Detwiler helped to put me on the correct path, in that it could simply be incorporated around the “entire” multiprocessing code in the scripttool file, instead of trying to interject the for loop into just one part of the overall multiprocessing code of the multicode file, as I was trying to do. I’m sure that with time and experience, I will get better at this and that it will become less of a challenge. Another part of the Python coding which I struggled with was the usage of the variables which represented the input files. In my trials, I would get errors stating that objects could not be pickled, which I tried to get around by ensuring that the shapefile names and file paths were strings instead of GIS objects. This is one way which I believe the code and script tools could be used for processing data in geodatabases. By first declaring the environment to be the geodatabase of interest, subsequent file name variables could be ensured that they were just string variables. When calling the Clip\_analysis tool, the tool would recognize that the variables represent names of the files based upon the geodatabase environment which is initially declared. While I have not tried this, I believe it would work based upon what was learned about working with geodatabases andsetting the working environment in GEOG 485.