NR 426 – Programming for GIS I

Lab 3 – Iterating with spatial data

Warner College of Natural Resources | Colorado State University | Instructor: Elizabeth Tulanowski

Learning objectives:

1. Understand the list methods in arcpy
2. Loop through a list of spatial objects in Python
3. Implement Python code to batch process ArcGIS tools

What to submit:

For each part, submit the code (as a .py file) from the demos and self-directed activities.

Lab 3A: The arcpy List methods

Lab 3A In-class follow-along demo (done as a group):

Work as a class to use the help and documentation to complete the code for these four tasks.

1. List all the polygon feature classes in the workspace:

   ```
   gdbList = arcpy.List_________ (_____)
   ```

2. List all the integer fields that start with "N":

   ```
   fieldList = arcpy.List_____("Roads", _____)
   ```

3. List all the rasters that are TIFs:

4. List all the tools from the analysis toolbox:

   ```
   toolList = arcpy.
   ```

Submit the above as a .py, .txt, Word doc or screenshot to Lab 3A

Use Lesson 3A ListDemoStarterCode.py to finish this demo. **Submit the completed .py as part of Lab 3A.**

Lab 3A Self-directed activities (done on your own):

- Write the code to loop through only the *polygon* feature classes in the San Antonio geodatabase and report some details about each:
  1. Print out the name of each polygon feature class, the number of features in each polygon feature class, (like we did in Demo) and their attribute field names (Refer to Lesson 3 List Methods slide).
  2. Format the output for readability (Remember \n and \t in your print statements)
  3. Format the code so it could easily be reused with any workspace or folder.

    a. Ie, Use variables, check for existence of the inputs
    b. Try running it on another folder or GDB to test it. How easy is it to run on something else?
4. Include code to report if there are *no* polygon feature classes
5. Grade will be based on *function* (does it run and accomplish the task?) as well as *style* (comments, readability, formatting of messaging).
6. OPTIONAL CHALLENGES:
    a. Format the messaging to be grammatically correct – ie, There *is* 1 polygon feature classes vs. There *are* 2 polygon features classes (this will need an if statement)
    b. Try arcpy.da.Walk to return the above info for all the GDBs in the Lab 3 data folder
    c. Write out your little report to a text file using .open and .write or sys.stdout
        i. Some samples of writing to a text file are in with Lesson 1 and easily found online

Hints for a workflow:

- **Start simple, then build complexity**
- Loop through all feature classes first and print out just the name
- Then modify to loop through only polygons and print out the name
- Include print statements so you can see what's happening
- Search for/ find the syntax for the tool you need to get the count of the number of features in a feature class. Use the Help and code sample as a guide.
    - Incorporate this into the loop
- Search for/ find the syntax for the method needed to get the fields for each feature class. You'll need to loop through each field of each feature class (a loop in a loop!)

Your final, polished result should look *something* like this, but feel free to add your own flair or wording:

```
There are 6 polygon feature classes in F:\NR426-427\NR426 Materials\Lesson 3 - Iterating spatial data\Lesson3LabData\CityOfSanAntonio.gdb
CityBoundaries contains 1 features and has these fields:
     OBJECTID_1
     Shape
     OBJECTID
     TYPE
     STATUS
     NAME
     ORDINANCEN
     TREEORDINA
     MSLINK
     STATE
     COUNTY
     FRANCHISEC
     TAXJURISDI
     PERMITLOCA
     INCORPORAT
     ZIPCODE
     Shape_Length
     Shape_Area

CrimesBySchoolDistrict contains 1 features and has these fields:
     OBJECTID_1
     Shape
     FID_1
     OBJECTID
     COLOR
     NAME
     NAME2
     DISTRICT_N
     DISTRICT
     Shape_Length
     Shape_Area

Texas_Counties_LowRes contains 1 features and has these fields:
     OBJECTID
     Shape
     NAME
     STATE_NAME
     STATE_FIPS
     CNTY_FIPS
     FIPS
     AREA
     POP1990
     POP2000
     POP90_SQMI
     Shape_Length
     Shape_Area

EdgewoodSD contains 1 features and has these fields:
     OBJECTID_1
     Shape
     OBJECTID
     COLOR
     NAME
     NAME2
     DISTRICT_N
     DISTRICT
     DISTRICT_C
     Shape_Length
     Shape_Area

Script completed. Pat yourself on the back!

Process finished with exit code 0
```

Lab 3B: Putting it all Together

Lab 3B Follow-along demo (Done as a group)

Let's work together with the Lesson 3B Describe Demo Starter Code.py to figure out how to solve these tasks, using the ArcGIS arcpy help:

1. If you start with this line of code, referring to our Lesson 3 Union City GDB:

```
dsc = arcpy.Describe("UnionCity.gdb")
```

a. How can you determine the *type* of geodatabase?

b. How can you retrieve the *path* to this geodatabase?

2. What code would you type to describe these 2 properties for a Watersheds feature class?

```
dsc = arcpy.Describe("Waterhseds")
```

a. Write the code to determine the *geometry* type
b. Write the code to return some *spatial reference* properties (type, name, unit)

3. What happens when you describe the spatial reference? Let's explore.

a. Spatial Reference is an object – so we can't just print it. We have to call out one of its properties, then print that. Refer to the help for more info.
b. Write a few lines of code to describe a dataset and return:
    i. spatialReference.name
    ii. spatialReference.type
    iii. spatialReference.domain

Lab 3B Self-directed activities: (Do on your own)

In a new script file (not the one from the demo): Write the code to use the list methods, loops, and appropriate tools on the datasets in RoadkillDataLab3.gdb, report some data properties and run tools:.

You will be graded on both function and style, so it has to perform the necessary operations, and also follow good coding practices and include basic error checking, like data existence, and handling empty lists. Do not manually type out the names of the layers and run tools brute force.

1. Print the number of tables (It will be 0)
2. **Rasters.** Print info about, and process, the raster:
    a. Report the number of rasters (It will be 1)
    b. Raster name, coordinate system, cellsize (with unit), and extent for the raster
    c. Correct projection, if needed: If the raster is not in UTM Zone 18 NAD83 (EPSG/Factory code: 26918), project it into UTM (Project Raster tool)
        i. Name the output descriptively (ie, NLCD_UTM)
3. **Feature classes**. Print info about, and process, the feature classes:

a. Report the number of feature classes (It will be 3)
b. Print out the feature class name, number of features, coordinate system, and extent for each feature class
c. Clip layers that are not the HRVCounties layer, to the HRVCounties layer
d. Correct projection, if needed: If a feature class is not in UTM Zone 18 NAD83 (EPSG/Factory code: 26918), project it into UTM (Project tool)
    i. *Roadkill points should be in GCS, so you should need to project it*
    ii. Name the output descriptively (ie Rdkill_UTM)

4. Use variables, loops, methods, or functions to perform the operations - Automate this process with a list methods and loops, don't run each tool manually, or manually type layer names.
    a. Use an if statement and exists to make the logic work better so you don't "accumulate" outputs each time you run the script. If do you accumulate outputs, just delete in ArcGIS before running again, and try to fix the code.
    b. Check that your input env.workspace exists before calling list methods. Include code to handle if there are no items of a certain type
5. Report a list of the feature classes and rasters in the GDB **after** you've run the above operations

6. *Advanced option #1: (optional)*
    a. Create the output projected and clipped layers in a different location (ie, make a Results folder or GDB).
7. *Advanced option #2: (optional)*
    a. Send the final output/messaging to a text file, instead of just the console window.

Your final polished output will look *something* like this, but feel free to add your own flair or wording:

```
*** Creating report for D:\NR426-427\NR426 Materials\NR426Data\WPSkiRuns.gdb: ***
D:\NR426-427\NR426 Materials\NR426Data\WPSkiRuns.gdb contains:
0 Tables

2 Rasters
       WP_Elevation is stored in NAD_1983_UTM_Zone_13N
       WPSlope_deg is stored in NAD_1983_UTM_Zone_13N

1 Feature Classes
 Report on feature classes and project if needed....
       WPSkiRuns is stored in NAD_1983_StatePlane_Colorado_North_FIPS_0501_Feet

Running project tool on WPSkiRuns
WPSkiRuns was projected successfully.

Clip the rasters to the WPSkiRuns layer using Extract by Mask....
Masking WP_Elevation.......
Masked WP_Elevation successfully

Masking WPSlope_deg.......
Masked WPSlope_deg successfully

**The contents of the D:\NR426-427\NR426 Materials\NR426Data\WPSkiRuns.gdb are now: **
Rasters:
       WP_Elevation
```

```
        WPSlope_deg
        WP_Elevation_clp
        WPSlope_deg_clp
Feature Classes:
        WPSkiRuns
        WPSkiRuns_UTM

 ***  Script completed. ***
```