Title: Basic Python Scripting with ArcGIS Pro

Critical Resources: an internet connected computer, a zip tool, Lab 6 datasets, Notepad ++ (or some other text editor/Python tool like IDLE), ArcGIS Pro 2.x

Purpose: The purpose of this exercise will be for you gain experience and knowledge with basic GIS programming concepts and algorithm development.

Notes: This exercise assumes some rudimentary knowledge of Python programming. It is beyond the scope this book and exercise to teach basic Python programming concepts. See the basics of Python programming Youtube video that was provided with this week's materials for Python basics.

See: https://pro.arcgis.com/en/pro-app/arcpy/get-started/what-is-arcpy-.htm for an introduction to ArcPy.

Learning Objectives: After completing the exercise, you will know:

- How to write basic Python statements that use variables, import statements, for loop and the ArcPy library.
- How to run Python code written outside of ArcGIS Pro in the ArcGIS Pro Python window.

Deliverables:

1. A screen shot showing that your code ran in ArcGIS Pro correctly (see last page in these instructions for how to setup your screen shot.

2. Python source code file.

Upload your write-up to the lab 6 drop box on myCourses.

Overview:

You will write a simple computer program that creates three buffer output feature classes from a single input feature class using Python and the ArcPy library.

Task 1 – Download data sets

Step 1: Down the lab_6_datasets.zip from myCourses

Step 2: Unzip the datasets

Unzip the datasets to a location where you can find them easily, like the C:\temp folder.

Before proceeding with the next tasks, open ArcGIS Pro.

Task 2 – Add datasets

In this task, your will add all of the dataset that were downloaded in task 1 to your map.

Step 1. Create a new map-based .aprx

Step 2: Add major roads shape file to your map

Task 3 – Write Python Code

Step 1. Notepad ++ setup.

Open Notepad ++, create a new file (File Menu > New)

Save the new file as buffer_loop.py (File Menu > Save As > [save in a directory you can find easily, like C:\temp] > Save as type: Python file (*.py, *.pyw))

Step 2. Write code.

In this step, follow each instruction carefully.

Python code statements you should type into Notepad++ are highlighted in light gray.

1. Import the arcpy module – this will tell ArcGIS Pro to use this module

```
#import arcpy module

import arcpy
```

2. Declare a string variable for the buffer input feature class (major_roads) referenced from the .aprx created in step 1

```
#define buffer input feature class

fc = "major_roads"
```

3. Declare a list variable with three buffer distances of 100, 500, and 750 feet

```
#define distance to buffer the roads

distance_list = ["100 meters", "200 meters", "300 meters"]
```

Loop through the buffer distance list

```
#loop through each distance and buffer

for dist in distance_list:
```

For every buffer distance in the list:

5a. Declare an output feature class name string variable, assign this variable a value using (a) the word “roads_” and then (b) the first three characters of the current buffer distance variable in the loop using string slicing[1]. Example: “roads_100”

```
#append  feature class name with  first three characters of  dist value

outname = fc + "_" + dist[0:3]
```

5b. Give the end user of your program some feedback that are buffering is about to happen.

```
print "Now Buffering: " + outname
```

5c. Run the buffer command using the parameters (a) the buffer input feature class variable, (b) the output feature class name created by string slicing, and (c) the current distance in the loop from the list variable of three buffer distances.

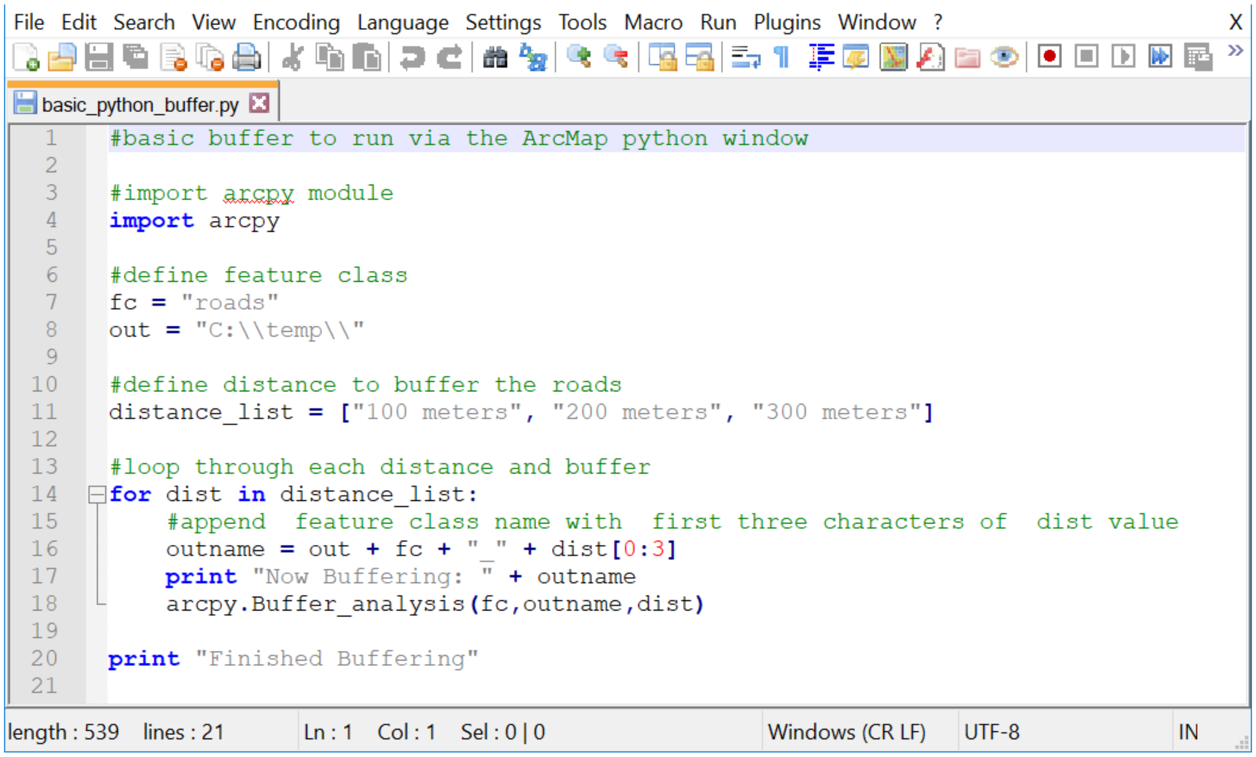
```
arcpy.Buffer_analysis(fc,outname,dist)
```

Give the end user of your program some feedback the program is finished.

```
print "Finished Buffering"
```

[1] http://pythoncentral.io/how-to-get-a-substring-from-a-string-in-python-slicing-strings/

Figure 1 shows what your final code should look like in Notepad++ if entered correctly.


File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
basic_python_buffer.py
length : 539 lines : 21 Ln : 1 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 IN

```
1  #basic buffer to run via the ArcMap python window
2  
3  #import arcpy module
4  import arcpy
5  
6  #define feature class
7  fc = "roads"
8  out = "C:\\temp\\"
9  
10 #define distance to buffer the roads
11 distance_list = ["100 meters", "200 meters", "300 meters"]
12 
13 #loop through each distance and buffer
14 for dist in distance_list:
15     #append  feature class name with  first three characters of  dist value
16     outname = out + fc + "_" + dist[0:3]
17     print "Now Buffering: " + outname
18     arcpy.Buffer_analysis(fc,outname,dist)
19 
20 print "Finished Buffering"
21 
```

Figure 1: The final Python code

Make note in particular in Figure 1 how the use of indents via tabs that are used for the for loop starting at line 14 as well as statements that beginning with # which comments like seen on line 1.

Task 4 - Run your script

In this final step, you will now run the code you create in Task 3 using the Python window of ArcGIS Pro.

If need be, re-open the .aprx file you created in Task 2.

To load the code you wrote Task 3 into the Python window of ArcGIS Pro, follow these steps:

Analysis Tab > Python > (open the Python window) > Command Prompt of Python window (bottom center where there is a flashing cursor) > Right click in the command prompt and select Load Code > Browse to where you saved buffer_loop.py created in Task 3 > Open.

The code you wrote in Task 3 should now be loaded into the Python window of ArcGIS.

If not already, place the cursor at the end of the last line of code (print ("Finished Buffering"))

Press the enter key twice to run the code.

If the code runs successfully, you should see:

1. Feedback printouts from processes that are running i.e.,

```
Now Buffering: major_roads_100

Now Buffering: major_roads_200

Now Buffering: major_roads_300

Finished Buffering
```

2. Three new feature classes added to your map that were created by your Python code.
3. Three new feature classes added to your default .aprx file geodatabase

Troubleshooting if the code does not work

1. If you run the code a second time and do not delete the three new feature classes that were added to your default .aprx file geodatabase, when you first ran the code, you will receive an error like:

```
arcgisscripting.ExecuteError: Failed to execute. Parameters are not valid.

ERROR 000725: Output Feature Class: Dataset C:\Users\....\major_roads_100 already

exists.

Failed to execute (Buffer).
```

As the code from this exercise does not check for the existence of the output feature classes first.

You will need to delete the output feature classes each time the code is run.

2. Bad indentation – Python is picky about items are indented. If you get: IndentationError: expected an indented block

Double check lines 13 to 17 in particular are formatted correctly.

Technical Variations

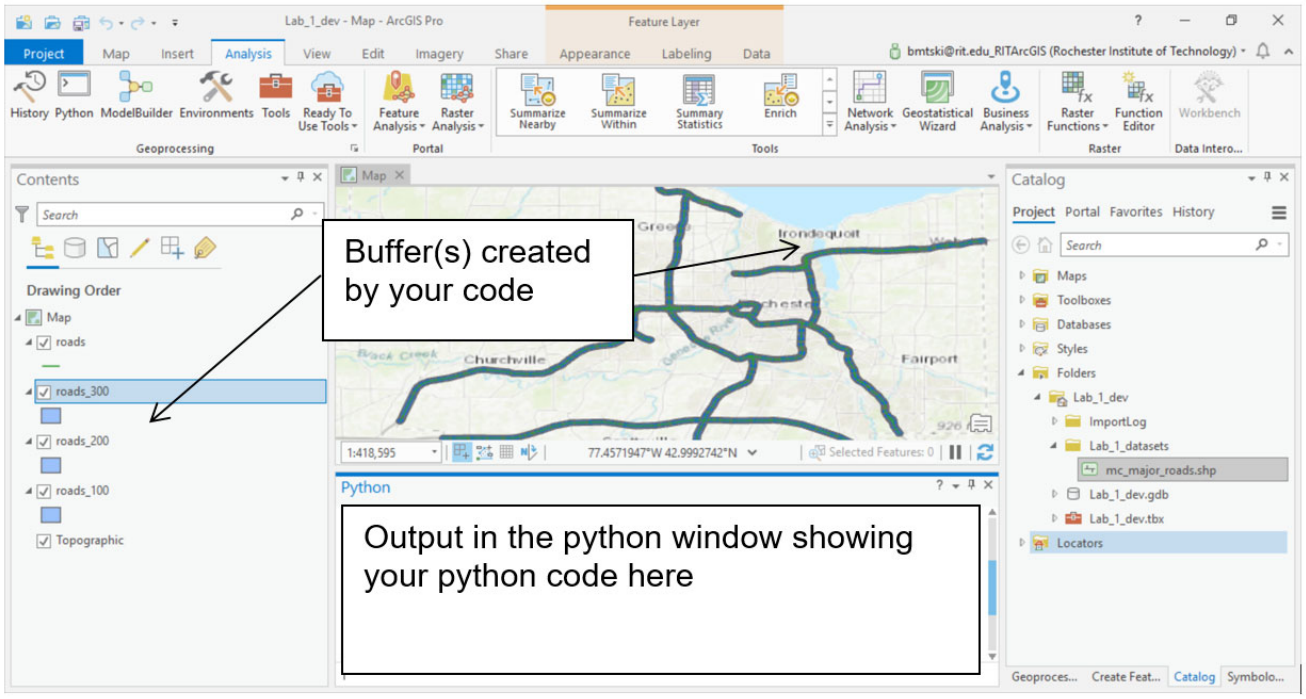
Implement one of the following technical variations based on the previous steps to expand upon the basic skills and ideas presented. Document with comments in your code which option you have selected. Specific instructions are not provided for self-learning.

1. Check for the existence of the output feature classes before the buffer command runs.

   1a. Delete existing output feature classes if they are found to exist before proceeding to generate new features classes via the buffer command.
2. Specify an output directory/file geodatabase in the code so the output do not automaticaly go to the default .aprx geodatabase.
3. Add dissolve options to the buffer command parameters.

Deliverables

How to create your screen shot deliverable:


Lab_1_dev - Map - ArcGIS Pro
Feature Layer
Project
Map
Insert
Analysis
View
Edit
Imagery
Share
Appearance
Labeling
Data
bmtski@rit.edu_RITArcGIS (Rochester Institute of Technology)
History
Python
ModelBuilder
Environments
Tools
Ready To Use Tools
Feature Analysis
Raster Analysis
Summarize Nearby
Summarize Within
Summary Statistics
Enrich
Network Analysis
Geostatistical Wizard
Business Analysis
Raster Functions
Function Editor
Workbench
Geoprocessing
Portal
Tools
Raster
Data Intero...
Contents
Search
Drawing Order
Map
roads
roads_300
roads_200
roads_100
Topographic
Map
Buffer(s) created by your code
Greece
Irondequoit
Rochester
Black Creek
Churchville
Fairport
1:418,595
77.4571947°W 42.9992742°N
Selected Features: 0
Python
Output in the python window showing your python code here
Catalog
Project
Portal
Favorites
History
Search
Maps
Toolboxes
Databases
Styles
Folders
Lab_1_dev
ImportLog
Lab_1_datasets
mc_major_roads.shp
Lab_1_dev.gdb
Lab_1_dev.tbx
Locators
Geoproces...
Create Feat...
Catalog
Symbolo...