Spatial Programming Lab 7

Seth Opatz

(Python scripts for all 5 questions attached with assignment submission)

Question 1 script:

```
# Module 7 Question 1
import arcpy
arcpy.env.overwriteOutput = True
arcpy.env.workspace = 'C:/PythonPro/Exercise07'

prj = arcpy.mp.ArcGISProject("Exercise07.aprx")
maps = prj.listMaps()
for map in maps:
    print("Map: " + map.name)
    print("Units: " + map.mapUnits)
    print("Coordinate System: " + map.spatialReference.name + '\n')
```

Question 1 output:

```
========== RESTART: C:/PythonPro/Exercise07/q1.py ========
Map: Oregon
Units: Meter
Coordinate System: NAD_1983_Oregon_Statewide_Lambert

Map: Population
Units: Foot_US
Coordinate System: NAD_1983_StatePlane_Oregon_North_FIPS_3601_Feet

Map: Landcover
Units: Meter
Coordinate System: WGS_1984_Web_Mercator_Auxiliary_Sphere
```

Question 2 script:

```
# Module 7 Question 2
import arcpy
arcpy.env.overwriteOutput = True
arcpy.env.workspace = 'C:/PythonPro/Exercise07'

prj = arcpy.mp.ArcGISProject("Exercise07.aprx")
maps = prj.listMaps()
for map in maps:
    print("Map: " + map.name)
    layers = map.listLayers()
    for layer in layers:
        if layer.isFeatureLayer:
            print(layer.name + " is a feature layer")
        elif layer.isBasemapLayer:
            print(layer.name + " is a basemap layer")
        elif layer.isRasterLayer:
            print(layer.name + " is a raster layer")
        else: print(layer.name + "is none of the types accounted for.")
    print() #Skip a line for visual clarity
```

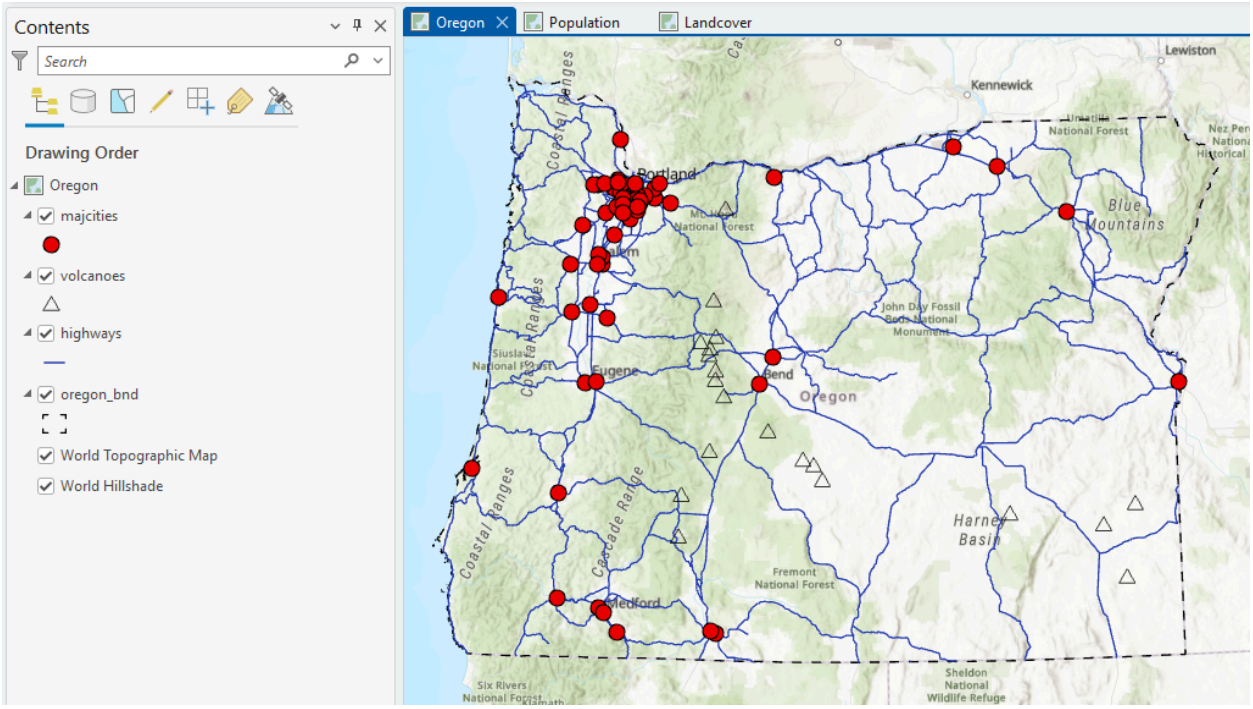
Question 2 output:

```
========== RESTART: C:/PythonPro/Exercise07/q2.py ==
Map: Oregon
volcanoes is a feature layer
rivers is a feature layer
oregon_bnd is a feature layer
National Geographic Style is a basemap layer
National Geographic Style Base is a basemap layer
World Hillshade is a basemap layer

Map: Population
counties is a feature layer
oregon_bnd is a feature layer
World Topographic Map is a basemap layer
World Hillshade is a basemap layer

Map: Landcover
cities is a feature layer
landcover is a raster layer
World Imagery is a basemap layer
```

Question 3 script:

```
import arcpy
arcpy.env.overwriteOutput = True
arcpy.env.workspace = 'C:/PythonPro/Exercise07'

prj = arcpy.mp.ArcGISProject("Exercise07.aprx")
oregon_map = prj.listMaps("Oregon")[0]
oregon_map.addBasemap("Topographic")

boundary = oregon_map.listLayers("oregon_bnd")[0]
sym = boundary.symbology

if boundary.isFeatureLayer and hasattr(sym, "renderer"):
    sym.updateRenderer('SimpleRenderer')
    symbol = sym.renderer.symbol
    symbol.applySymbolFromGallery("Dashed Black Outline (1pt)")
    # Apply changes
    sym.renderer.symbol = symbol
    boundary.symbology = sym

oregon_map.addDataFromPath('C:/PythonPro/Exercise07/Exercise07.gdb/majcities')
oregon_map.addDataFromPath('C:/PythonPro/Exercise07/Exercise07.gdb/highways')

# Apply symbology to majcities
majcities_layer = oregon_map.listLayers("majcities")[0]
sym = majcities_layer.symbology

if majcities_layer.isFeatureLayer and hasattr(sym, "renderer"):
    sym.updateRenderer("SimpleRenderer")
    symbol = sym.renderer.symbol
    symbol.applySymbolFromGallery("Circle 3", 1)
    symbol.color = {'RGB': [230, 0, 0, 100]}  # Poinsettia red
    sym.renderer.symbol = symbol
    majcities_layer.symbology = sym

rivers = oregon_map.listLayers("rivers")[0]
oregon_map.removeLayer(rivers)

volcanoes_layer = oregon_map.listLayers("volcanoes")[0]
sym = volcanoes_layer.symbology

if volcanoes_layer.isFeatureLayer and hasattr(sym, "renderer"):
    sym.updateRenderer("SimpleRenderer")
    symbol = sym.renderer.symbol
    symbol.applySymbolFromGallery("Triangle 3")

    #Color as the instructions specify
    #(Don't know why we're putting in color but making it transparent too)
    symbol.color = {'RGB': [56, 168, 0, 0]}
    sym.renderer.symbol = symbol
    volcanoes_layer.symbology = sym

prj.saveACopy("Exercise07_q3.aprx")
del prj
```

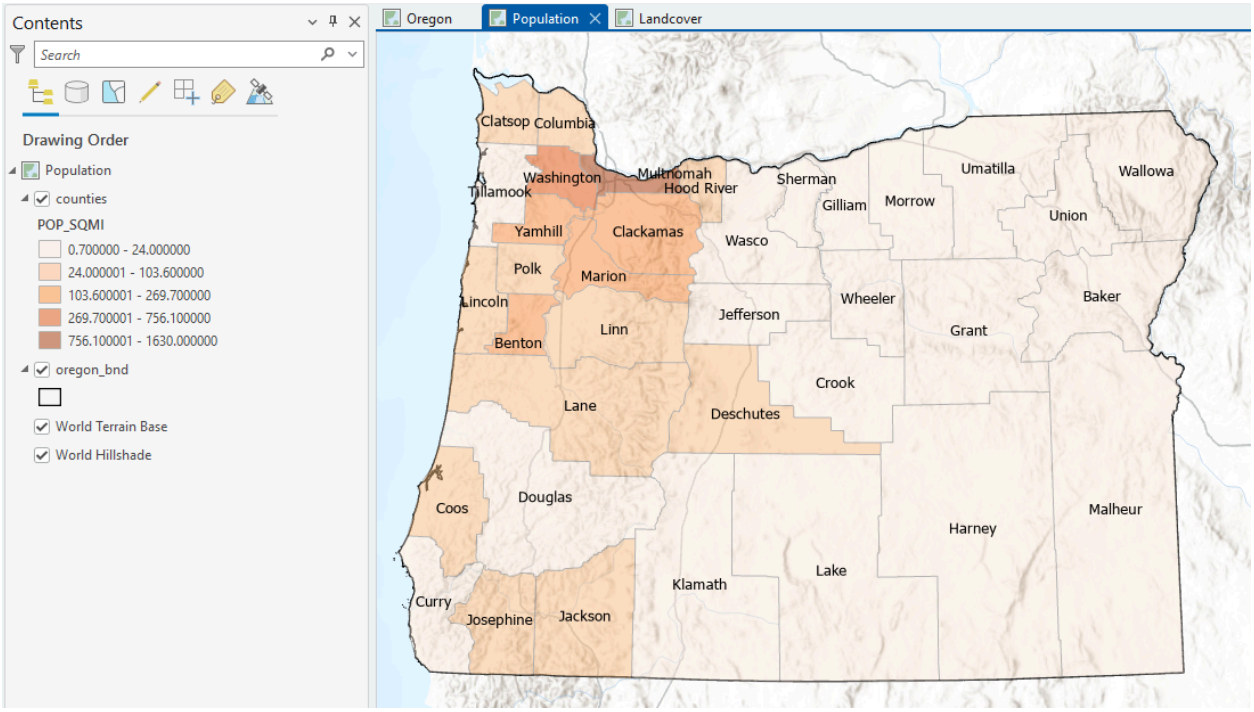
Question 3 output:



Question 4 script:

```
# Lab 7 Question 4
import arcpy
arcpy.env.overwriteOutput = True
arcpy.env.workspace = 'C:/PythonPro/Exercise07'

prj = arcpy.mp.ArcGISProject("Exercise07_q3.aprx")
pop_map = prj.listMaps("Population")[0]
pop_map.addBasemap("Terrain with Labels")

#Remove labels from basemap
labels = pop_map.listLayers("World Terrain Reference")[0]
pop_map.removeLayer(labels)

boundary = pop_map.listLayers("oregon_bnd")[0]
sym = boundary.symbology
if boundary.isFeatureLayer and hasattr(sym, "renderer"):
    sym.updateRenderer('SimpleRenderer')
    symbol = sym.renderer.symbol
    symbol.applySymbolFromGallery("Black Outline (1pt)")
    sym.renderer.symbol = symbol
    boundary.symbology = sym

counties = pop_map.listLayers("counties")[0]
sym = counties.symbology
if counties.isFeatureLayer and hasattr(sym, "renderer"):
    sym.updateRenderer("GraduatedColorsRenderer")
    sym.renderer.classificationField = "POP_SQMI"
    sym.renderer.classificationMethod = "NaturalBreaks"
    sym.renderer.breakCount = 5
    sym.renderer.colorRamp = prj.listColorRamps("Oranges (5 Classes)")[0]
    counties.symbology = sym

# Set 50% transparency
counties.transparency = 50

# Label county names
label_classes = counties.listLabelClasses()
if label_classes:
    label_class = label_classes[0]
    label_class.expression = "$feature.NAME"
    label_class.visible = True
    counties.showLabels = True

prj.saveACopy("Exercise07_q4.aprx")
del prj
```

Question 4 output:



Question 5 script:

```
# Lab 7 Question 5
import arcpy
arcpy.env.overwriteOutput = True
arcpy.env.workspace = 'C:/PythonPro/Exercise07'

prj = arcpy.mp.ArcGISProject("Exercise07_q4.aprx")
lc_map = prj.listMaps("Landcover")[0]

cities = lc_map.listLayers("cities")[0]
lc_map.removeLayer(cities)
wi = lc_map.listLayers("World Imagery")[0]
lc_map.removeLayer(wi)

lc_remap = arcpy.sa.RemapValue([
    #Forest: (includes evergreen, deciduous, mixed, transitional) green
    [29, 1], [31, 1], [39, 1], [71, 1], [32, 1], [33, 1], [34, 1], [35, 1], [36, 1], [37, 1],
    [38, 1], [41, 1], [42, 1], [44, 1], [45, 1], [46, 1], [47, 1], [48, 1], [49, 1], [50, 1],
    [51, 1], [52, 1], [53, 1], [54, 1], [55, 1], [56, 1], [57, 1], [58, 1], [59, 1], [60, 1],
    [61, 1], [62, 1], [63, 1], [64, 1], [65, 1], [66, 1], [67, 1], [30, 1], [40, 1], [68, 1],
    [69, 1], [73, 1], [110, 1], [113, 1], [123, 1], [124, 1], [126, 1], [127, 1], [129, 1],
    [130, 1], [131, 1], [133, 1], [150, 1], [118, 1], [119, 1], [120, 1], [121, 1],

    #Barren: (includes ash, dune, rock) purple
    [26, 2], [115, 2], [116, 2], [117, 2], [21, 2], [22, 2], [27, 2], [15, 2], [18, 2], [28, 2],
    [78, 2], [122, 2], [12, 2], [13, 2], [14, 2], [16, 2], [17, 2], [19, 2], [20, 2], [24, 2],
    [25, 2], [72, 2], [107, 2],

    #Water: Blue
    [1, 3], [136, 3],

    #Wetlands: dark green
    [137, 4], [138, 4], [139, 4], [142, 4], [144, 4], [147, 4], [149, 4], [114, 4], [125, 4], [132, 4],
    [134, 4], [128, 4], [140, 4], [141, 4],

    #Built-up: (urban, developed, residential) red
    [5, 5], [4, 5], [3, 5], [6, 5],

    #Agriculture: (agriculture, crops, pasture) yellow
    [10, 6], [2, 6], [8, 6], [11, 6], [9, 6],

    #Brush: (grasslands, shrublands, steppe) tan
    [111, 7], [112, 7], [84, 7], [89, 7], [98, 7], [104, 7], [23, 7], [75, 7], [86, 7], [94, 7],
    [95, 7], [96, 7], [97, 7], [99, 7], [100, 7], [101, 7], [102, 7], [103, 7], [105, 7], [106, 7],
    [108, 7], [135, 7], [145, 7], [146, 7], [77, 7], [74, 7], [76, 7], [90, 7], [91, 7], [92, 7],
    [148, 7], [43, 7], [70, 7], [79, 7], [80, 7], [81, 7], [82, 7], [83, 7], [85, 7], [87, 7],
    [88, 7], [93, 7]

])

out_recl = arcpy.sa.Reclassify('Exercise07.gdb/landcover', 'Value', lc_remap, 'NODATA')
out_recl.save("Exercise07.gdb/lc_reclass")
lc_map.addDataFromPath('C:/PythonPro/Exercise07/Exercise07.gdb/lc_reclass')

lc = lc_map.listLayers("lc_reclass")[0]
sym = lc.symbology
sym.updateColorizer('RasterUniqueValueColorizer')
sym.colorizer.field = "Value"
```

```
label_dict = {
    '1': 'Forest',
    '2': 'Barren',
    '3': 'Water',
    '4': 'Wetlands',
    '5': 'Urban',
    '6': 'Cropland',
    '7': 'Grassland'
}

for group in sym.colorizer.groups:
    for item in group.items:
        val = str(item.values[0])
        if val in label_dict:
            item.label = label_dict[val]
            if val == '1':
                item.color = {'RGB': [34, 139, 34, 100]}
            elif val == '2':
                item.color = {'RGB': [160, 32, 240, 100]}
            elif val == '3':
                item.color = {'RGB': [0, 0, 255, 100]}
            elif val == '4':
                item.color = {'RGB': [0, 100, 0, 100]}
            elif val == '5':
                item.color = {'RGB': [255, 0, 0, 100]}
            elif val == '6':
                item.color = {'RGB': [255, 255, 0, 100]}
            elif val == '7':
                item.color = {'RGB': [210, 180, 140, 100]}
lc.symbology = sym

prj.saveACopy("Exercise07_q5.aprx")
del prj
```

Question 5 output: