Rebecca Perlin
GIS 610, Exercise 3

1. Explain in a few sentences the differences between properties and methods in Python. Properties are for computed attributes, they set the parameters for a class or object but no doesn’t have any attributes passing through. While methods are functions that preform some type of work that includes factors feeding through the method to produce something to be incorporated into a larger program.

2. Look at the following statements below and indicate if each one is a property or method and why.
   a. arcpy.env.overwriteOutput = True – Property, this performs a specific task, doesn’t incorporate any attributes.
   b. arcpy.SearchCursor(“roads”, “TYPE” <> 4’) – Method, this is a function that will incorporate attributes to do some work, as indicated by the roads, type, and 4 parameters.
   c. row.setValue(‘distance’,100) – Method, this is a function that will incorporate attributes to do some work, as indicated by the distance and 100 parameters.
   d. ArcGISProject.dateSaved – Property, this performs a specific task, doesn’t incorporate any attributes.
   e. Table.isBroken – Property, this performs a specific task, doesn’t incorporate any attributes.

3. Review the following function and explain what you think is happening. Are parameters being passed into the function? If so, what’re their data types? Write what you think the output of the function would be if it were invoked/called.

```
def letterFunc (wordParam1, wordParam2):
if (wordParam1[0].lower() == wordParam2[0].lower()):
return True
else:
return False
```

   This function looks at the first character of each string data attribute, if the first character of each attribute match it prints True, if they do not match it prints False.

4. Write a function definition which satisfies the following requirements:
   a. Accepts a list of names as a parameter
   b. Prints ‘Happy Birthday’ to each person

```
NamesList = ["Sarah", "John"]
for names in NamesList:
   print("Happy Birthday " + names)
```

```
Happy Birthday Sarah
Happy Birthday John
Press any key to continue . . .
```

5.

```
import arcpy

from arcpy import env
arcpy.CreateFileGDB_management(r'C:\gisclass', 'classHW.gdb')

current_workspace = r'C:\gisclass\classHW.gdb'
geometry_type = "POLYGON"
spatial_reference = arcpy.SpatialReference(102100)
fetureClassNamesList = ['CapitalCities', 'Landmarks', 'HistoricPlaces', 'StateNames',
'Nationalities', 'Rivers']
arcpy.env.workspace = current_workspace
arcpy.env.overwriteOutput = True

def createFeatureClass(in_fc_name):
    arcpy.CreateFeatureclass_management(current_workspace, in_fc_name, geometry_type,
"", "DISABLED", "DISABLED", spatial_reference)
    print('Feature Class ' + in_fc_name + ' was sucessfull created.')

createFC = [createFeatureClass(fc) for fc in fetureClassNamesList]

print('All Done')
```

6.

```
import arcpy

arcpy.env.workspace = r'C:\Users\rvanclea\Desktop\Exercise 3.gdb\Exercise 3.gdb'
inFeatures = 'CallsforService'
fieldname = 'Crime_Explanation1'
field_type = 'text'

arcpy.AddField_management(inFeatures,fieldname,"TEXT")

featureClass = r'C:\Users\rvanclea\Desktop\Exercise 3.gdb\Exercise
3.gdb\CallsforService'
FieldNames = ['CFSType','Crime_Explanation1']

with arcpy.da.UpdateCursor(featureClass, FieldNames) as cursor:
    for x in cursor:
            if x[0] == ('Burglary Call'):
                    x[1] = 'This is a burglary'
                    cursor.updateRow(x)

                    print('row updated')
```

7.

```
import arcpy
arcpy.env.workspace = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb'
arcpy.env.overwriteOutput = True

inFeatures = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb\CallsforService'
outLocation = 'FeaturetoFeature'
outFeatureClass = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb\General_Offense'

arcpy.MakeFeatureLayer_management(inFeatures, 'CallsforService_lyr')
arcpy.SelectLayerByAttribute_management('CallsforService_lyr','NEW_SELECTION','x_rand
>10')
```

```
arcpy.CopyFeatures_management('CallsforService_lyr',r'C:\Users\rvanclea\Desktop\Exercise
3\Exercise 3.gdb\FtoFCopy')
```

8.

```
    import arcpy
    arcpy.env.workspace = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb'
    featureClass = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb\CallsforService'
    result = arcpy.GetCount_management(featureClass)
    print(' {} has {} records'. format(featureClass, result[0]))
```

9.

```
    import arcpy

    current_workspace = r'C:\gisclass\classHW.gdb'
    geometry_type = "POLYGON"
    spatial_reference = arcpy.SpatialReference(102100)
    featureClassNamesList = ['NewFC']
    arcpy.env.workspace = current_workspace
    arcpy.env.overwriteOutput = True

    def createFeatureClass(in_fc_name):
        arcpy.CreateFeatureclass_management(current_workspace, in_fc_name, geometry_type,
    "", "DISABLED", "DISABLED", spatial_reference)
        print('Feature Class ' + in_fc_name + ' was sucessfully created.')

    createFC = [createFeatureClass(fc) for fc in featureClassNamesList]

    print('All Done')

    inFeatures = 'NewFC'
    fieldname = 'NewField'
    field_type = 'text'

    arcpy.AddField_management(inFeatures, fieldname, "TEXT")

    featureClass = r'C:\gisclass\classHW.gdb\NewFC'

    print('field created')

    domName = "NewDomain"
    gdb = current_workspace
    inFeatures = featureClass
    inField = fieldname

    arcpy.CreateDomain_management(gdb, domName, "This is the stuff",
                                  "TEXT", "CODED")

    domDict = {"1":"stuff", "2": "stuff2", "3": "stuff3",
               "4": "stuff4", "5": "Boom!"}

    for code in domDict:
        arcpy.AddCodedValueToDomain_management(gdb, domName, code, domDict[code])

    arcpy.AssignDomainToField_management(inFeatures, inField, domName)

    print('Success')
```

10.

```
import arcpy
from arcpy import env
env.overwriteOutput = True

target_features = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb\Tracts
join_features = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise 3.gdb\General_Offense'
out_feature_class = r'C:\Users\rvanclea\Desktop\Exercise 3\Exercise
3.gdb\OffenseJoinTracts'

arcpy.SpatialJoin_analysis(target_features, join_features, out_feature_class)
```

11.

https://github.com/RPerlin/Exercise3.git