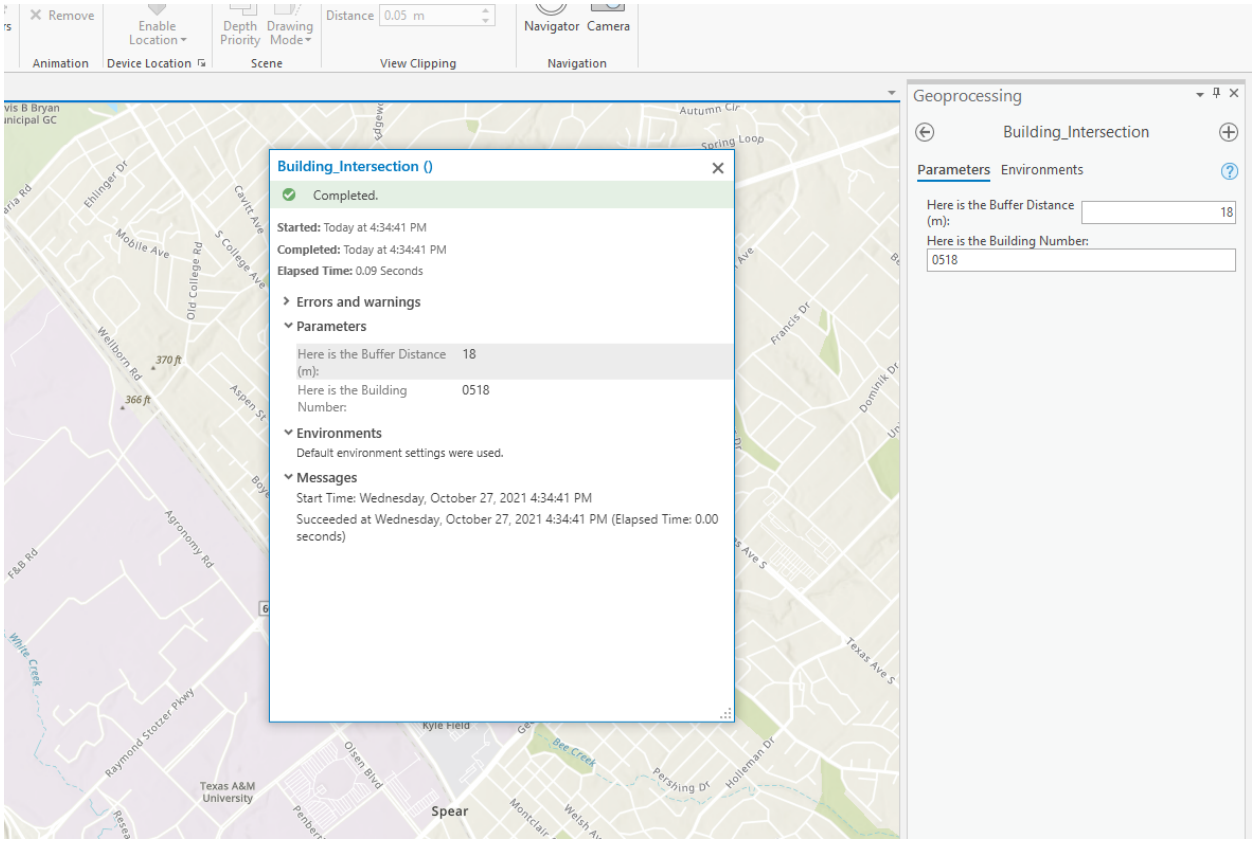
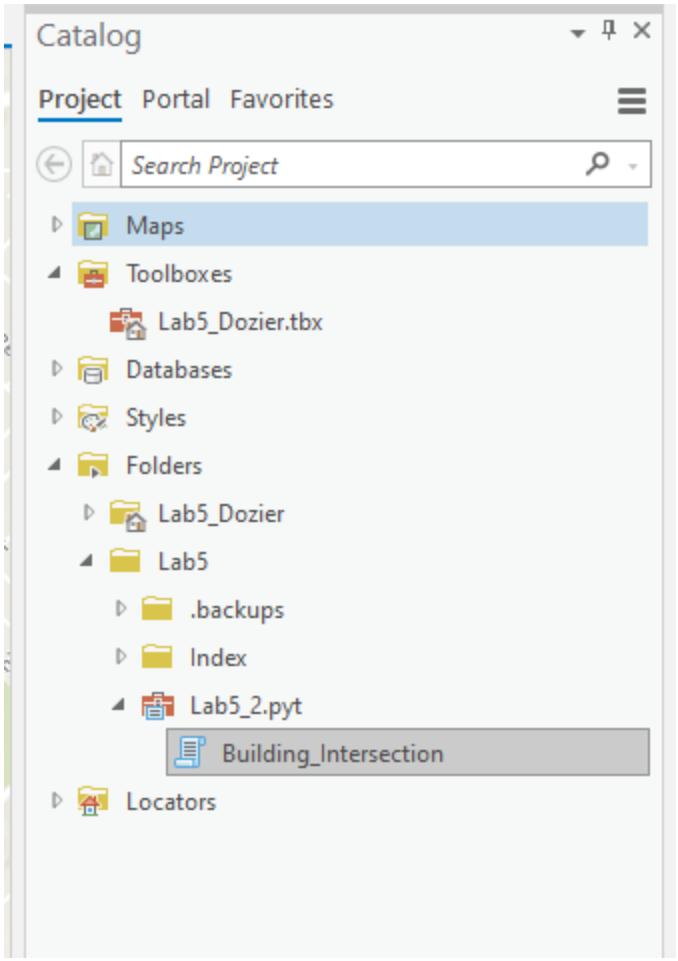
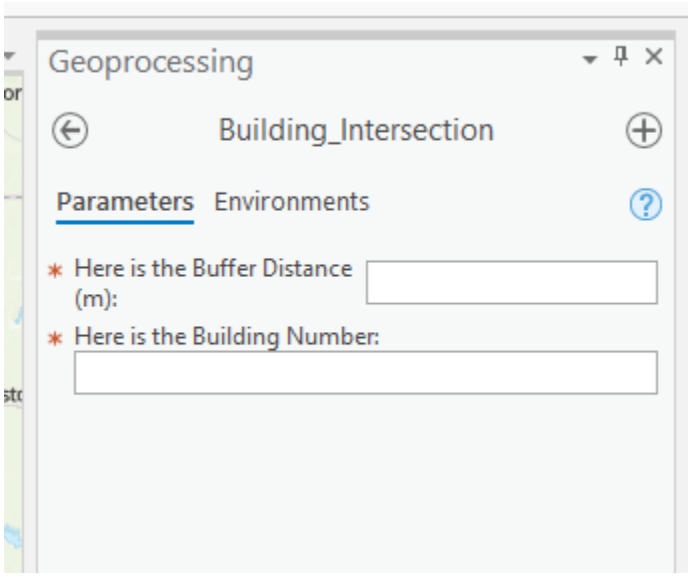
Garrett Dozier - GEOG 392 - Section: 501 - LAB 5

Screenshot of Tool successfully running in ArcGIS Pro:



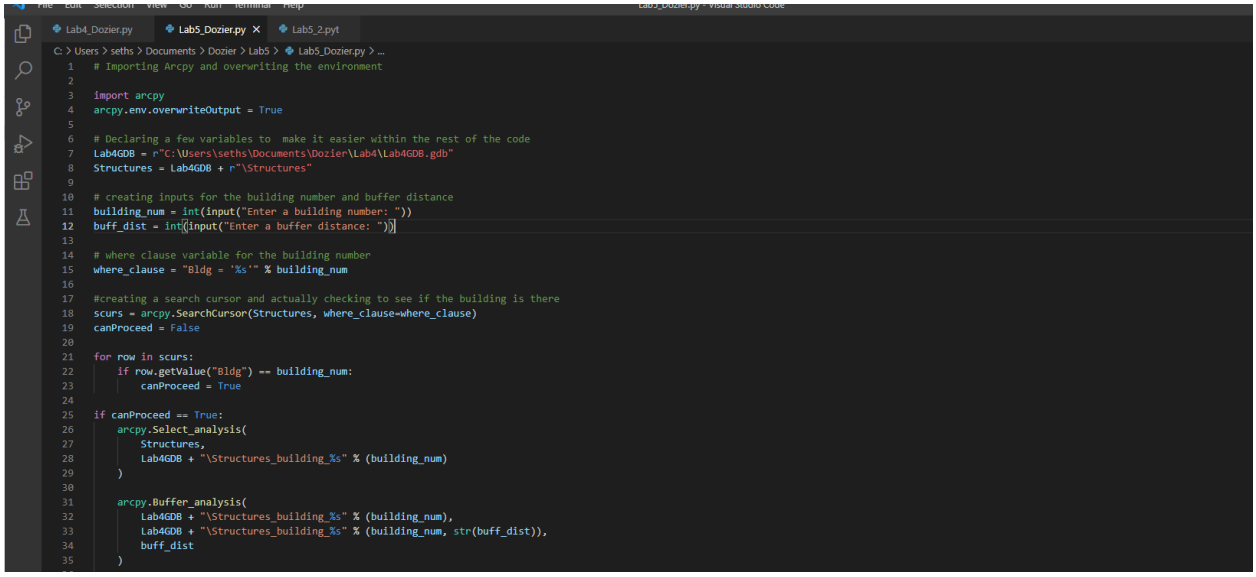
Screenshot of Toolbox in ArcGIS Pro:



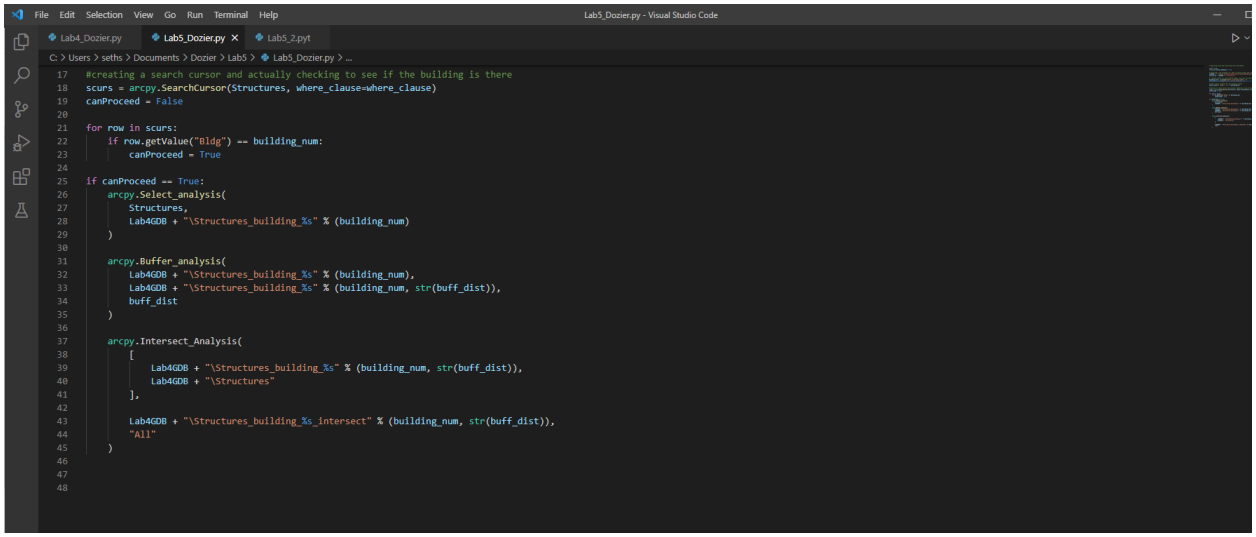


Screenshot of Tool code:

Garrett Dozier - GEOG 392 - Section: 501 - LAB 5



```python
# Importing Arcpy and overwriting the environment

import arcpy
arcpy.env.overwriteOutput = True

# Declaring a few variables to  make it easier within the rest of the code
Lab4GDB = r"C:\Users\seths\Documents\Dozier\Lab4\Lab4GDB.gdb"
Structures = Lab4GDB + r"\Structures"

# creating inputs for the building number and buffer distance
building_num = int(input("Enter a building number: "))
buff_dist = int(input("Enter a buffer distance: "))

# where clause variable for the building number
where_clause = "Bldg = '%s'" % building_num

#creating a search cursor and actually checking to see if the building is there
scurs = arcpy.SearchCursor(Structures, where_clause=where_clause)
canProceed = False

for row in scurs:
    if row.getValue("Bldg") == building_num:
        canProceed = True

if canProceed == True:
    arcpy.Select_analysis(
        Structures,
        Lab4GDB + "\Structures_building_%s" % (building_num)
    )

    arcpy.Buffer_analysis(
        Lab4GDB + "\Structures_building_%s" % (building_num),
        Lab4GDB + "\Structures_building_%s" % (building_num, str(buff_dist)),
        buff_dist
    )
```

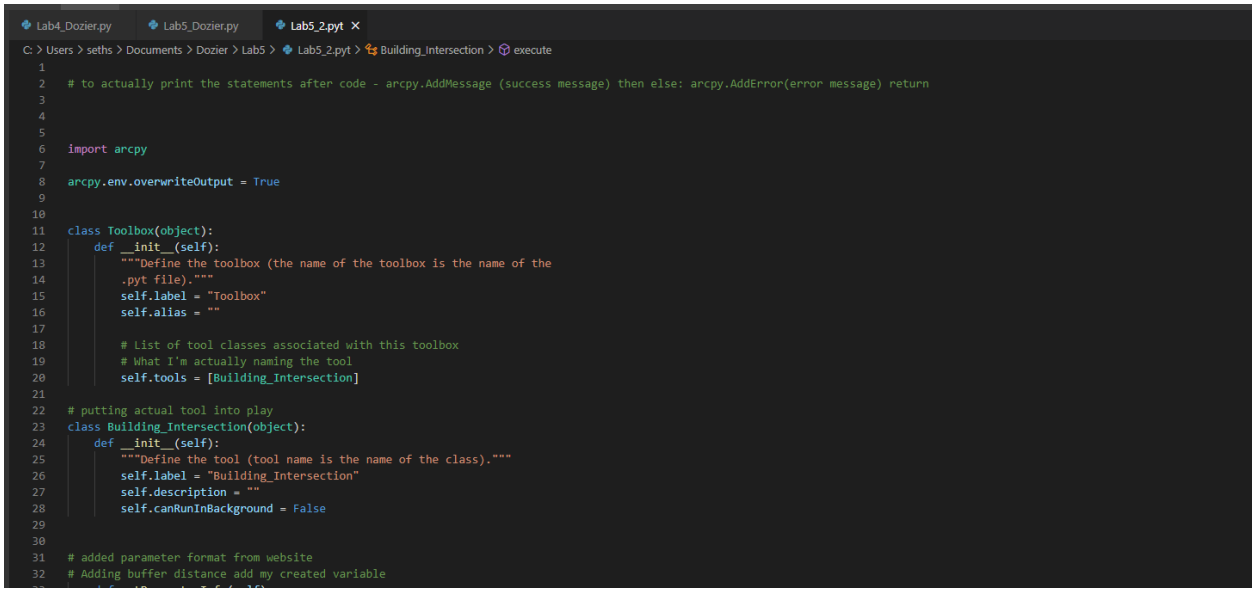


```python
#creating a search cursor and actually checking to see if the building is there
scurs = arcpy.SearchCursor(Structures, where_clause=where_clause)
canProceed = False

for row in scurs:
    if row.getValue("Bldg") == building_num:
        canProceed = True

if canProceed == True:
    arcpy.Select_analysis(
        Structures,
        Lab4GDB + "\Structures_building_%s" % (building_num)
    )

    arcpy.Buffer_analysis(
        Lab4GDB + "\Structures_building_%s" % (building_num),
        Lab4GDB + "\Structures_building_%s" % (building_num, str(buff_dist)),
        buff_dist
    )

    arcpy.Intersect_Analysis(
        [
            Lab4GDB + "\Structures_building_%s" % (building_num, str(buff_dist)),
            Lab4GDB + "\Structures"
        ],

        Lab4GDB + "\Structures_building_%s_intersect" % (building_num, str(buff_dist)),
        "All"
    )
```

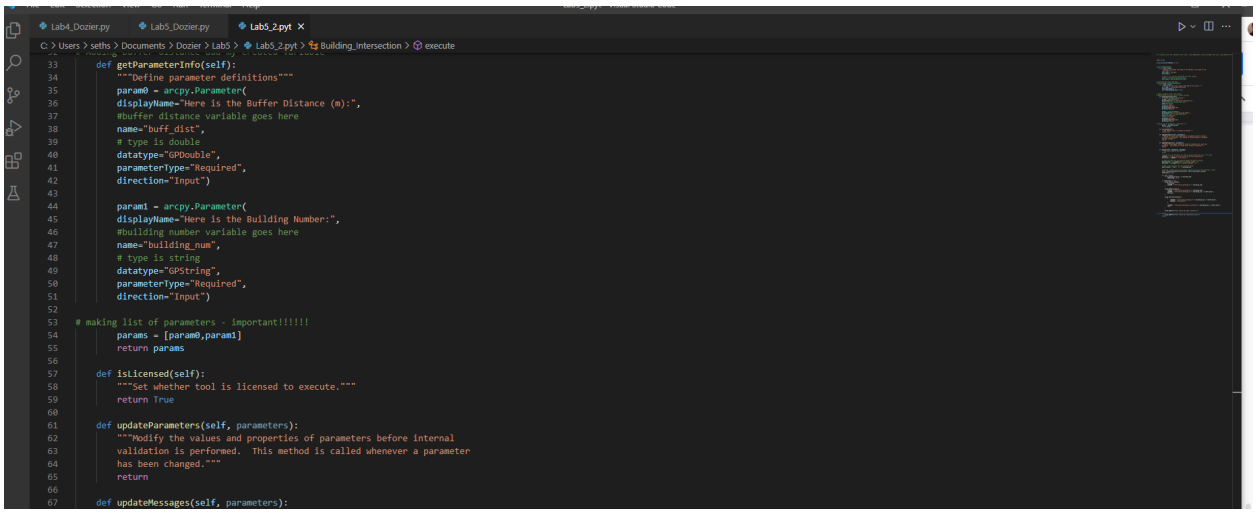
```
PS C:\Users\seths> & "C:/Program Files/ArcGIS/Pro/bin/Python/envs/arcgispro-py3/python.exe" c:/Users/seths/Documents/Dozier/Lab5/Lab5_Dozier.py
Enter a building number: 2
Enter a buffer distance: 3
```

Screenshot of Toolbox code:

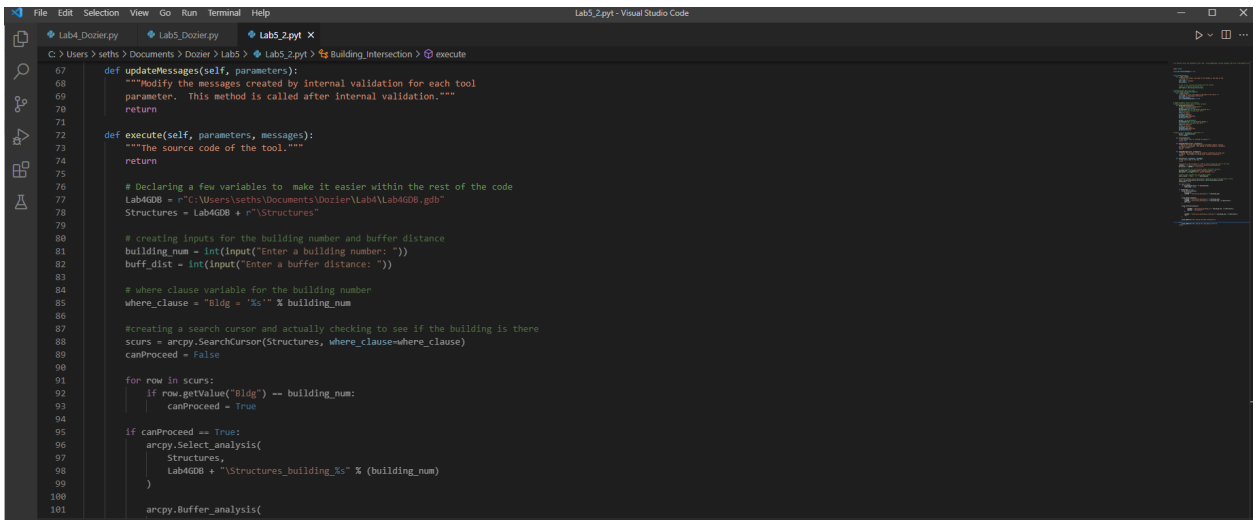
Garrett Dozier - GEOG 392 - Section: 501 - LAB 5



```python
# to actually print the statements after code - arcpy.AddMessage (success message) then else: arcpy.AddError(error message) return



import arcpy

arcpy.env.overwriteOutput = True


class Toolbox(object):
    def __init__(self):
        """Define the toolbox (the name of the toolbox is the name of the
        .pyt file)."""
        self.label = "Toolbox"
        self.alias = ""

        # List of tool classes associated with this toolbox
        # What I'm actually naming the tool
        self.tools = [Building_Intersection]

# putting actual tool into play
class Building_Intersection(object):
    def __init__(self):
        """Define the tool (tool name is the name of the class)."""
        self.label = "Building_Intersection"
        self.description = ""
        self.canRunInBackground = False


# added parameter format from website
# Adding buffer distance add my created variable
```



```python
    def getParameterInfo(self):
        """Define parameter definitions"""
        param0 = arcpy.Parameter(
        displayName="Here is the Buffer Distance (m):",
        #buffer distance variable goes here
        name="buff_dist",
        # type is double
        datatype="GPDouble",
        parameterType="Required",
        direction="Input")

        param1 = arcpy.Parameter(
        displayName="Here is the Building Number:",
        #building number variable goes here
        name="building_num",
        # type is string
        datatype="GPString",
        parameterType="Required",
        direction="Input")

    # making list of parameters - important!!!!!!
        params = [param0,param1]
        return params

    def isLicensed(self):
        """Set whether tool is licensed to execute."""
        return True

    def updateParameters(self, parameters):
        """Modify the values and properties of parameters before internal
        validation is performed.  This method is called whenever a parameter
        has been changed."""
        return

    def updateMessages(self, parameters):
```



```python
    def updateMessages(self, parameters):
        """Modify the messages created by internal validation for each tool
        parameter.  This method is called after internal validation."""
        return

    def execute(self, parameters, messages):
        """The source code of the tool."""
        return

        # Declaring a few variables to  make it easier within the rest of the code
        Lab4GDB = r"C:\Users\seths\Documents\Dozier\Lab4\Lab4GDB.gdb"
        Structures = Lab4GDB + r"\Structures"

        # creating inputs for the building number and buffer distance
        building_num = int(input("Enter a building number: "))
        buff_dist = int(input("Enter a buffer distance: "))

        # where clause variable for the building number
        where_clause = "Bldg = '%s'" % building_num

        #creating a search cursor and actually checking to see if the building is there
        scurs = arcpy.SearchCursor(Structures, where_clause=where_clause)
        canProceed = False

        for row in scurs:
            if row.getValue("Bldg") == building_num:
                canProceed = True

        if canProceed == True:
            arcpy.Select_analysis(
                Structures,
                Lab4GDB + "\Structures_building_%s" % (building_num)
            )

            arcpy.Buffer_analysis(
```

Garrett Dozier - GEOG 392 - Section: 501 - LAB 5