Community Garden Suitability Map

Individual Contributions

[redacted] Final Report, Presentation Slides, Helped with Euclidean Distance code
[redacted] Program Design, Project code, Editing Final Report

Problem Statement

Community gardens have a variety of health and social benefits. While there are a few community gardens throughout Wake County, NC, it could be interesting to explore how to create gardens that promote more community engagement, involve a wider audience, and make community gardening an important part of local culture. To create a community gardening culture, we think it would be important to make the gardens accessible to community members and integrate events taking place in the gardens with other community engagement spaces, such as schools and libraries. The main goal of this project is to create a suitability map for Wake County, North Carolina that shows the best spot for creating a community garden given a variety of factors including, available land, location of schools, parks, libraries, and other community gathering spots, and distance to water sources and roads. A secondary goal of this project would be to design the suitability map model in such a way so that it can be replicated for any county or area given the proper datasets.

Literature Review

Community gardens have been shown to have a great positive impact on the physical health of participants by increasing physical activity and by improving diet through a heightened awareness, positive perception of, and access to fruits and vegetables (Lovell et al. 2-3). Being centered on communal and collective activities, community gardens provide an opportunity for increased positive social interaction between community members, which promotes better overall well-being, both mentally and physically and strengthens social resilience (3-4). Community gardens also have the potential to address wider social needs by promoting community agency, including allowing communities to take food security issues into their own hands (Cumbers et al. 135). And indeed, community gardens can be used as an opportunity to "promote sustainable food production and knowledge...working with schools, housing associations and various parts of local government" (142). And finally, community gardens can become a center of "social capital, social support, and social connectedness" during times of crises, such as during the Covid-19 pandemic (Mejia et al. 17).

Data Sources

The project will focus on Wake County, North Carolina (Figures 1 and 2). The majority of the data that will be used in this project will be from the Wake County Open Data Portal (https://data.wakegov.com/). This includes data on zoning, irrigation (streams and bodies of

water), roads, schools, libraries, and parks. This data is readily available and easy to use for the purposes we intend it for. Each of these resources is reputable and reliable.

Methodology

Some of the Wake County datasets are separated by city, such as zoning data, so an initial step in processing the data will be creating one unified dataset for the whole of Wake County with a Spatial Join. Additionally, several geoprocessing tools will be used to do spatial analysis for this project, including the buffer tool. Finally, in order to create a suitability ranking, we plan to create a continuous (raster) dataset so that we can assign values to the raster and do calculations based on the different parameters (Map Algebra), and for this we need to transform my vector data to raster data.

Explanation of Site Selection

When it comes to selecting the best locations to place community gardens around Wake county the criteria we chose to focus on was the locations of pre existing community parks as well as the proximity to certain features like roads, rivers or streams, lakes or ponds, and schools or libraries. Each of these factors can affect the quality of the crops produced in these community gardens as well as the ease of access for the people who would be likely to use them.

In the process of selecting the most suitable locations for these gardens, proximity to roadways is important because in certain studies such as Agriculture and proximity to roads: How should farmers and retailers adapt? Examples from the Ile-de-France region by Caroline Petit, Christine Aubry, and Elisabeth Rémy-Hall it is shown that air pollution by road traffic can negatively impact the quality of agricultural production. For this reason we have created a buffer of 250 meters along roads as this 250 meter distance is the recommended separation from agricultural production and pollution produced by vehicles. Along with this buffer we used euclidean distance calculations to find the correct distance from the landmark in question.

With the same style of buffer and euclidean distance calculations we found the proximity to rivers and streams as well as lakes and ponds. This distance is important because the closer the community garden is to these natural water resources the easier it is to have some form of natural irrigation.

In the case of schools and libraries we found it to be important to place these community gardens in the closest proximity to these establishments with the purpose of increasing community involvement with the gardens. The article Agriculture-based community engagement in rural libraries by Vandana Singh, Bharat Mehra, and Everett Scott Sikes explains how the correlation between libraries and community gardens can be a very beneficial attribute for a community.

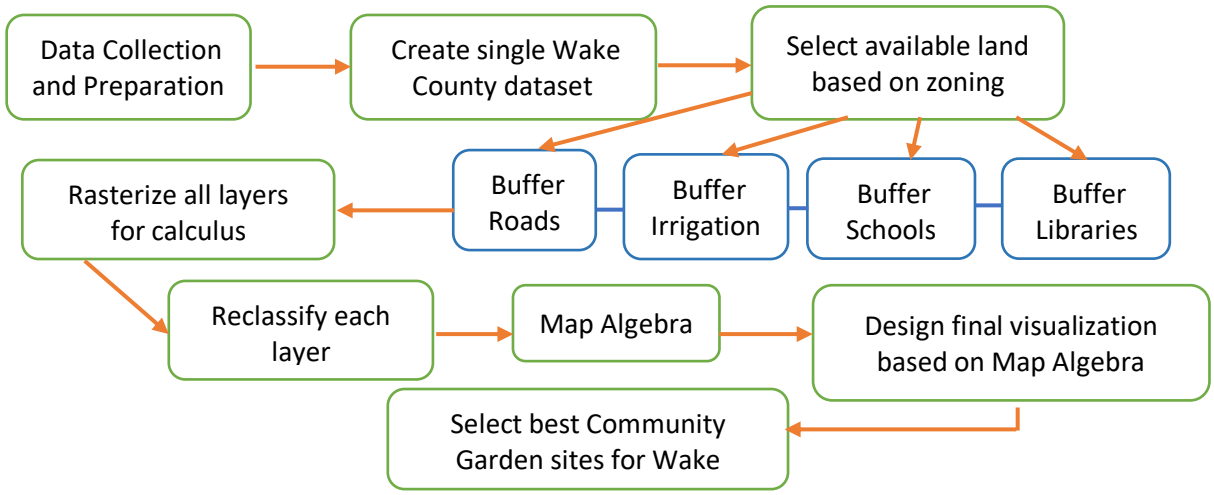
Putting all of these factors together allows us to create a map of Wake county that highlights the most suitable locations for a community garden. This is done by creating a composite score based on the location and landmarks around it including libraries, roads, rivers, lakes, and

streams. The final outcome is to produce a visual representation of the most likely candidates for a hopeful garden and therefore an improvement to the overall community wellbeing.

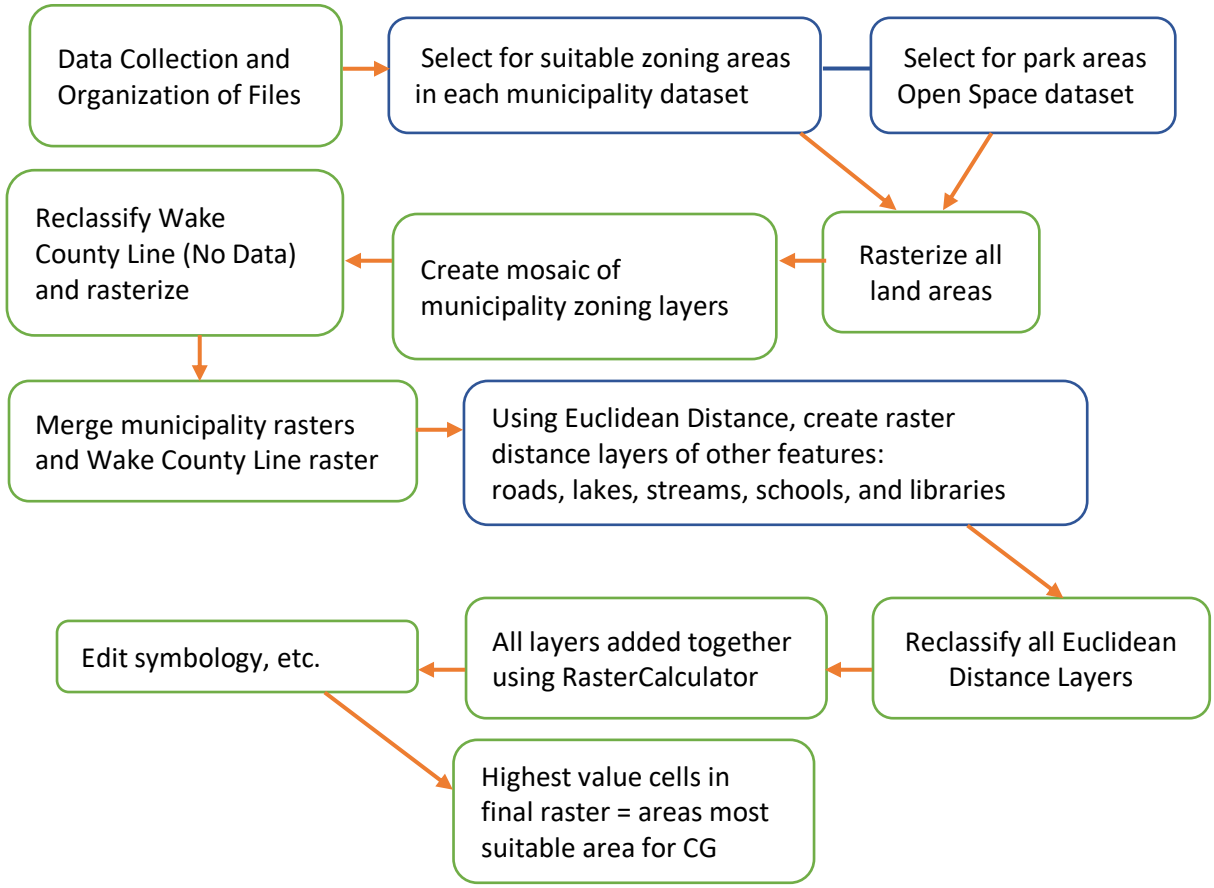
Program Design

Flow Charts

The original program design is laid out in the flow chart below:



However, after looking at the datasets and beginning to code the program, the program design was altered slightly, which is reflected in the flow chart below.



Explanation of Code

After collecting the data and organizing the data files, each individual county file was processed using search and insert cursors to create a new column ("CGZONE") with binary notation denoting whether a given observation had the correct value in the zoning "CLASS" field. The Open Space dataset that contained information on parks in Wake County was processed and classified in the same manner.

After this classification of the zoning and park areas, each municipality zoning area and the park area datasets were rasterized individually by utilizing a loop to run through each dataset. The Wake County Line dataset, which contains all of the area in Wake County and is used for no data areas, is similarly rasterized. Following rasterization, the individual municipality rasters are joined into one new raster, using the MosaicToNewRaster tool, which results in a raster showing the suitable and unsuitable zoning areas for community gardens based on the "CGZONE" column. Next, the Wake County Line dataset is reclassified and given a no data value (99999), followed by which it is merged to the municipality mosaic raster, resulting in a single raster that reflects the community garden-relevant zoning for all of wake county.

The other layers for the project (roads, lakes and streams, schools, and libraries) are created using the Euclidean Distance tool. The raster resulting for each feature set is then reclassified using the Reclassify tool to reflect distance values decided upon based on reviewing the literature.

Subsequently, all of the layers (zoning, parks, roads, lakes, streams, schools, and libraries) are added together using the RasterCalculator tool. In all of the raster layers that were created, higher value cells reflected favorable conditions, while lower value cells reflected unfavorable conditions. The exception to this is the no data value of (99999). The highest possible value a cell could have is 13. Thus, in the final suitability map, the highest value cells are the most suitable areas for community gardens in Wake County.

Results Interpretation

The final map (Figure 3) shows the most and least suitable sites for community gardens in Wake County, with the lowest suitability rating being 6 and the highest 13. In Wake County the most suitable areas for community gardening are located in the west and northwest. There are also small pockets of highly suitable land across the entire county, meaning there is potential for building a community garden in many of the various municipalities and communities across the county.

Based on the suitable areas demarcated on the map, users can use it to pick a site for creating a community garden that is located in a favorable natural environment, will engage the local population, and will integrate community gardening into other aspects of community life, such as school life and library activities. Additionally, the model used to create this map could be easily replicated for any other county in North Carolina or other geographic regions given the proper data.

Limitations and Future Extension

While many of the datasets, such as rivers and roads, were easy to locate and manipulate in the program, the zoning datasets were both difficult to obtain and manipulate. Thus, while locating or creating a unified dataset of rivers, roads, lakes and ponds, libraries, and schools for the entire state of North Carolina, or even the US could be relatively easy, obtaining or creating one unified dataset of the zoning areas for North Carolina, let alone the US, would be very difficult. Furthermore, there does not appear to be a unified system for classifying and encoding zoning types; even within the Wake County area large variation was observed. To this end we conclude that a lack of available zoning data, lack of consistency in zoning classification, and similar barriers to creating a unified zoning dataset make automating this community garden suitability assessment process or converting it into a tool inconceivable at present. Further research would need to be done into easily acquiring and compiling zoning data or alternative datasets that provide the same or similar function as zoning data could be explored

References

Cumbers, Andrew, et al. “The Work of Community Gardens: Reclaiming Place for Community in the City.” *Work, Employment, and Society*, vol. 31, no. 1, 2017, pp. 133-149, doi.org/10.1177/0950017017695042.

Lovell, Rebecca, et al. “What are the health and well-being impacts of community gardening for adults and children: a mixed method systematic review protocol.” *Environmental Evidence*, vol. 3, no. 20, 2014. doi.org/10.1186/2047-2382-3-20.

Mejia, Angie, et al. “Community Gardening During Times of Crisis: Recommendations for Community-Engaged Dialogue, Research, and Praxis”. *Journal of Agriculture, Food Systems, and Community Development*, vol. 10, no. 1, 2020, pp. 13–19, doi.org/10.5304/jafscd.2020.101.006.

Petit, Caroline, et al. “Agriculture and Proximity to Roads: How Should Farmers and Retailers Adapt? Examples from the Ile-De-France Region.” *Land Use Policy*, Pergamon, 25 Mar. 2011, https://www.sciencedirect.com/science/article/abs/pii/S0264837711000226.

Singh, Vandana, et al. “Agriculture-Based Community Engagement in Rural Libraries.” *Journal of Librarianship and Information Science* (2021): 096100062110157. Crossref. Web.

Appendix A

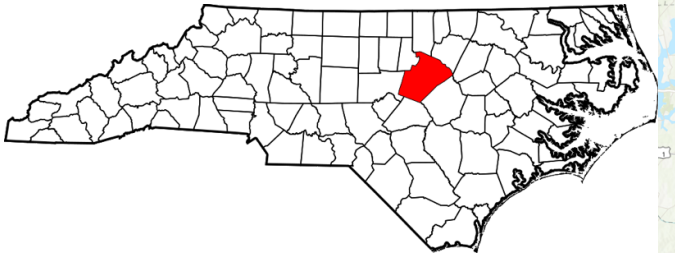


Figure 1. Map of Study Area in Context

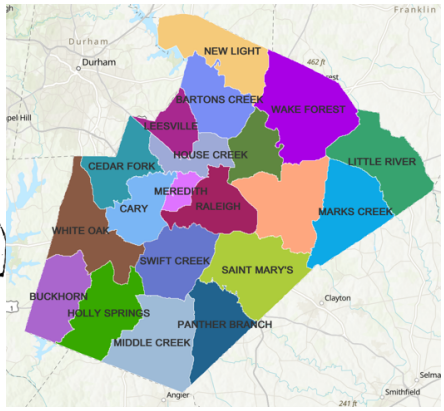


Figure 2. Map of Study Area

Suitability Map for Community Gardening
Wake County

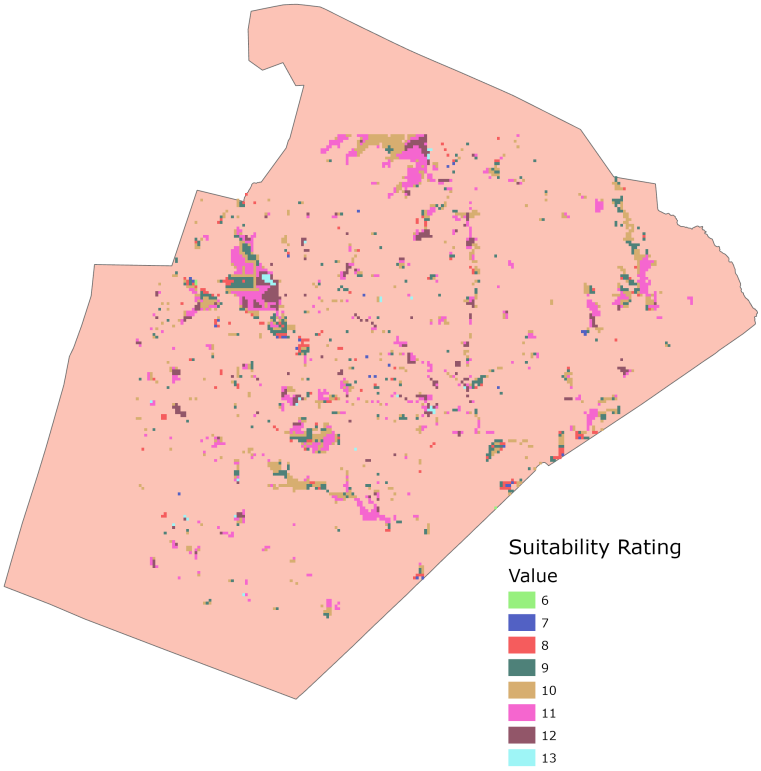


Figure 3

Appendix B

```
import arcpy
import os
from arcpy.sa import *
from arcpy.ia import *

##############################
### Workspace Setup
##############################
os.chdir("..")
print(os.path.abspath(os.curdir))

# make workspace
arcpy.env.workspace = "\\."
print("current workspace", arcpy.env.workspace)

# To allow overwriting outputs change overwriteOutput option to True.
arcpy.env.overwriteOutput = True



########################
### Zoning Classification
########################
```

```
# create fucntion to classify each zoning area
def zoneClassify(shpf):
    # create new field to store 0/1 values that mark if zoneing class is correct
    # for CG
    arcpy.AddField_management(shpf, "CGZone", "SHORT")

    # create update cursor to create label each zone 0/1 based on suitibility for CG
    with arcpy.da.UpdateCursor(shpf, ["CLASS", "CGZone"]) as zoneClassCursor:
        for row in zoneClassCursor:
            # residential districts suitable for CG overall; res. districts start with "R"
            if row[0].startswith("R") == True:
                row[1] = 1
            else:
                row[1] = 0
            # update cursor with updated list
            zoneClassCursor.updateRow(row)


# create search cursor for Raleigh because does not have CLASS field
arcpy.AddField_management(".\\data\\Zoning\\Raleigh_Zoning.shp", "CGZone", "SHORT")
zoneList = ["R-1", "R-2", "R-4", "R-6", "R-10", "AP", "MH", "RX", "OX", "IX", "OP", "NX", "CX",
"DX"]
with arcpy.da.UpdateCursor(".\\data\\Zoning\\Raleigh_Zoning.shp", ["ZONE_TYPE", "CGZone"])
as zoneClassCursor2:
    for row in zoneClassCursor2:
```

```
        # check each zone against the specific zones Raleigh laid out for CG suitibility
        if row[0].startswith(tuple(zoneList)) == True:
            row[1] = 1
        else:
            row[1] = 0
        # update cursor with updated list
        zoneClassCursor2.updateRow(row)



# create search cursor for Wakeforest because does not have CLASS field
arcpy.AddField_management(".\\data\\Zoning\\Wake_Forest_Zoning.shp", "CGZone",
"SHORT")
with arcpy.da.UpdateCursor(".\\data\\Zoning\\Wake_Forest_Zoning.shp", ["ZoneClass",
"CGZone"]) as zoneClassCursor3:
    for row in zoneClassCursor3:
        # check each zone against the specific zones Raleigh laid out for CG suitibility
        if row[0].startswith("R") == True:
            row[1] = 1
        else:
            row[1] = 0
        # update cursor with updated list
        zoneClassCursor3.updateRow(row)


# create list of files to loop through function with
cities = ["Wake_County", "Apex", "Cary", "Fuquay-Varina", "Garner", "Holly_Springs",
"Knightdale", "Morrisville", "Rolesville", "Wendell", "Zebulon"]
```

```
# loop through each city's zoning file
for city in cities:
    # create file name for each city
    cityFile = ".\\data\\Zoning\\" + city + "_Zoning.shp"
    # call zoneClassify function
    zoneClassify(cityFile)

########################
### Park Classification
########################

# define function to classify Open_Space dataset
def parkClassify(shpf):
    # create new field to store 0/1 values that mark if park
    arcpy.AddField_management(shpf, "CGZone", "SHORT")

    # create update cursor to create label each zone 0/1 based on suitibility for CG
    with arcpy.da.UpdateCursor(shpf, ["TYPE", "CGZone"]) as parkCursor:
        for row in parkCursor:
            # residential districts suitable for CG overall; res. districts start with "R"
            if row[0] == "PARK":
                row[1] = 1
            else:
```

```
            row[1] = 0
        # update cursor with updated list
        parkCursor.updateRow(row)

# classify park layer
parkClassify(".\\data\\Open_Space\\Open_Space.shp")

#####################################
### Rasterize Zoning, Parks, and WCL
#####################################

# create function to raterize zoning and parks layers
def rasterizeZoningParks(shpf):
    # retrieve file name for raster output file name
    cityFile = os.path.split(shpf)
    # cut down characters in file name to avoid ERROR 000878 (grid name longer than 13)
    nCF = cityFile[1]
    # take of ending to edit for output file name
    cf = nCF[0:4]
    # specify output file name for each shpf
    outName = ".\\output\\ZoningRastFINAL\\" + cf + "Rst"
    # rasterize feature layer
    arcpy.conversion.FeatureToRaster(shpf, "CGZone", outName)
```

```
# create list of files to loop through function with

cities = ["Apex", "Cary", "Fuquay-Varina", "Garner", "Holly_Springs", "Knightdale", "Morrisville",
"Rolesville", "Wendell", "Zebulon", "Wake_Forest", "Raleigh"]


# loop through each city's zoning file

for city in cities:

    # create file name for each city

    cityFile = ".\\data\\Zoning\\" + city + "_Zoning.shp"

    # call rasterizeZoningParks function

    rasterizeZoningParks(cityFile)


# rasterize wake county line for no data areas

arcpy.conversion.FeatureToRaster(".\\data\\WakeCountyLine\\Wake_County_Line.shp",
"FTR_CODE", ".\\output\\WCLRastFINAL\\WCLNoDatRst")


# rasterize park layer

arcpy.conversion.FeatureToRaster(".\\data\\Open_Space\\Open_Space.shp", "CGZone",
".\\output\\ParkRastFINAL\\ParkRst")


################################

### Land Raster Mosaic and Merge

################################


# initialize list of raster files

rastFiles = []

#assign directory
```

```
directory = ".\\output\\ZoningRastFINAL"

# use loop to create lsit of raster files in directory

for filename in os.listdir(directory):

    f = os.path.join(directory, filename)

    rastFiles.append(f)


# merge zoning rasters

arcpy.MosaicToNewRaster_management(rastFiles, ".\\output\\ZoningRastFINAL",
"TestRastMos", "", "", "", 1)


# reclassify wake county line raster to reflect no data

WCLReclass = Reclassify(".\\output\\WCLRastFINAL\\WCLNoDatRst", "VALUE",
RemapValue([[0, 99999]]))

WCLReclass.save(".\\output\\WCLRastFINAL\\WCLNoDatRstRe")


# merge wake county line raster and zoning mozaic raster

landRast = arcpy.ia.Merge([".\\output\\ZoningRastFINAL\\TestRastMos",
"..\\output\\WCLRastFINAL\\WCLNoDatRstRe"], "FIRST")

landRast.save(".\\output\\LandRastFINAL\\landRast")


#######################################

### Euclidean Distance Feature Layers

#######################################


roadsED = EucDistance(".\\data\\RoadsWC\\RoadsWC.shp")

roadsED.save(".\\output\\EDRastFINAL\\RoadsED")
```

```
print("roadsED completed.")

lakesED = (".\\data\\LakesPonds\\LakesPonds.shp")
lakesED.save(".\\output\\EDRastFINAL\\LakesED")
print("lakesED completed.")

riversED = EucDistance("..\\data\\Hydrolines\\hydrolines.shp")
riversED.save("..\\output\\EDRastFINAL\\RiversED")
print("riversED completed.")

schoolsED = EucDistance("..\\data\\WCSchools\\WCSChools.shp")
schoolsED.save("..\\output\\EDRastFINAL\\SchoolsED")
print("schoolsED completed.")

librariesED = EucDistance("..\\data\\Libraries\\Libraries.shp")
librariesED.save("..\\output\\EDRastFINAL\\LibrariesED")
print("librariesED completed.")

########################################
### Reclassify Euclidean Distance Layers
########################################

# Reclassify each feature layer's euclidean distance
RoadsED = Reclassify(".\\output\\EDRastFINAL\\RoadsED", "VALUE",
```

```
                RemapRange([[1001,86239.2,1],[0,1000,0]]))
RoadsED.save(".\\output\\EDRastRecFINAL\\RoadsEDRe")

RiversED = Reclassify(".\\output\\EDRastFINAL\\RiversED","VALUE",
                RemapRange([[4801,,0],[3601,4800,1], [2401,3600,2], [1201,2400,3],[0,1200,4]]))
RiversED.save(".\\output\\EDRastRecFINAL\\RiversEDRe")

LakesED = Reclassify(".\\output\\EDRastFINAL\\LakesPondsED","VALUE",
                RemapRange([[4801,,0],[3601,4800,1], [2401,3600,2], [1201,2400,3],[0,1200,4]]))
LakesED.save(".\\output\\EDRastRecFINAL\\LakPonEDRe")

SchoolsED = Reclassify(".\\output\\EDRastFINAL\\SchoolsED","VALUE",
                RemapRange([[10561,,0],[0,10560,1]])
SchoolsED.save(".\\output\\EDRastRecFINAL\\SchoolsEDRe")

LibrariesED = Reclassify(".\\output\\EDRastFINAL\\LibrariesED", "VALUE",
                RemapRange([[10561,,0],[0,10560,1]]))
LibrariesED.save(".\\output\\EDRastRecFINAL\\LibrariesEDRe")
```

```
#######################################
### Add All Raster Layers
#######################################

# create variable for each raster file
zoning = Raster(".\\output\\LandRastFINAL\\landRast")
parks = Raster(".\\output\\ParkRastFINAL\\ParkRst")
roads = Raster(".\\output\\EDRastRecFINAL\\RoadsEDRe")
rivers = Raster(".\\output\\EDRastRecFINAL\\RiversEDRe")
lakesPonds = Raster(".\\output\\EDRastRecFINAL\\LakPonEDRe")
schools = Raster(".\\output\\EDRastRecFINAL\\SchoolsEDRe")
libraries = Raster(".\\output\\EDRastRecFINAL\\LibrariesEDRe")

# add raster layers to create suitibiltiy analysis raster
cgSuitRast = zoning + parks + roads + rivers + lakesPonds + schools + libraries
cgSuitRast.save(".\\output\\SuitibilityFINAL\\CGSuitRast")
```