Ian van den Bold

GIS Programming

Project

Output:

A screenshot of a computer

Description automatically generated

**Code without using the cursor and Create Featureclass:**

import arcpy

# Set the workspace environment

arcpy.env.workspace = r"C:\Users\ivand\OneDrive\Documents\ArcGIS\Projects\GISProgramming\_Project.gdb" # Set the path to your geodatabase

# Set the input CSV file

input\_csv = r"C:/Users/ivand/OneDrive/Desktop/geog4057-main/project/boundary.csv"

# Set the name for the output feature class

output\_fc\_name = "Points"

# Set the spatial reference (NAD 1983 StatePlane North Carolina FIPS 3200 (Meters))

spatial\_reference = arcpy.SpatialReference(32119) # 32119 is the code for the specified spatial reference

# Create the table view from the CSV file

arcpy.MakeTableView\_management(input\_csv, "CSV\_Table")

# Create the XY event layer from the table view

arcpy.MakeXYEventLayer\_management("CSV\_Table", "X", "Y", "XY\_Event\_Layer", spatial\_reference)

# Copy the features from the XY event layer to a feature class

arcpy.CopyFeatures\_management("XY\_Event\_Layer", output\_fc\_name)

print("Conversion completed successfully.")

**Code using cursor and Create Featureclass:**

import arcpy

import os

# Set the workspace environment

arcpy.env.workspace = r"C:\Users\ivand\OneDrive\Documents\ArcGIS\Projects\GISProgramming\_Project" # Set the path to your geodatabase

# Set the input CSV file

input\_csv = r"C:/Users/ivand/OneDrive/Desktop/geog4057-main/project/boundary.csv"

# Set the name for the output feature class

output\_fc\_name = "OutputFeatureClass"

# Set the spatial reference (NAD 1983 StatePlane North Carolina FIPS 3200 (Meters))

spatial\_reference = arcpy.SpatialReference(32119) # 32119 is the code for the specified spatial reference

# Create the feature class

arcpy.management.CreateFeatureclass(

arcpy.env.workspace,

output\_fc\_name,

"POINT",

spatial\_reference=spatial\_reference

)

# Add fields to the feature class (adjust field names and types as needed)

arcpy.management.AddField(output\_fc\_name, "Col", "LONG")

arcpy.management.AddField(output\_fc\_name, "Row", "LONG")

#I tried adding some of the code that I could see during class, seeing as our code was so similar, however, that did not work.

'''

#Add fields to the feature class

field\_names = ["Col", "Row"]

field\_names = ["Long", "Long"]

for field\_name, field\_type in zip(field\_names, field\_types):

arcpy.management.AddField(output\_feature\_class, field\_name, field\_type)

'''

# Open an InsertCursor

with arcpy.da.InsertCursor(output\_fc\_name, ["SHAPE@", "Col", "Row"]) as cursor:

# Read the CSV file and insert rows into the feature class

with open(input\_csv, "r") as file:

header = file.readline() # Skip the header line

for line in file:

values = line.strip().split(',')

col, row, x, y = map(float, line.strip().split(",")) # Assuming X and Y are in the 3rd and 4th columns

#the x and y, and the collumn and row were mixed up at first, so I had to change them around.

point = arcpy.Point(x, y)

row\_to\_insert = (point, col, row)

cursor.insertRow(row\_to\_insert)

print("Conversion completed successfully.")