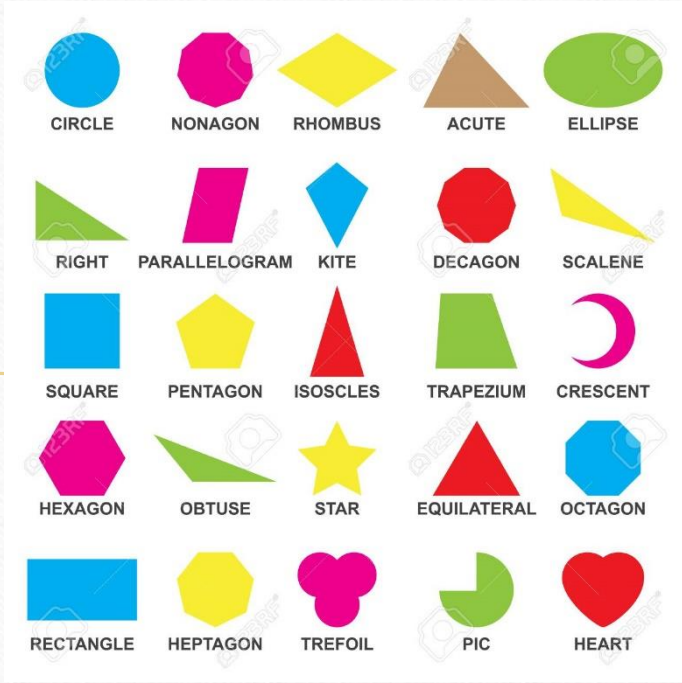
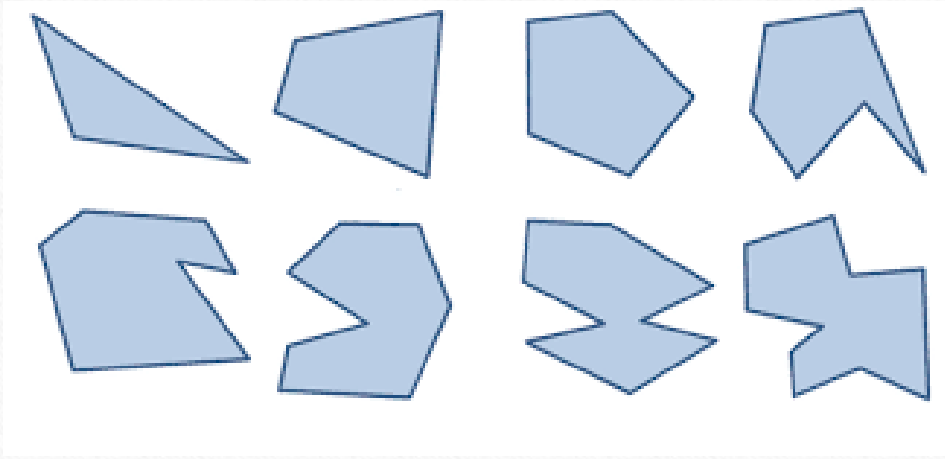
Polygon Area Calculator

Chris Pugel

Objective

- The goal of this assignment is to be able to find the area of any polygon
- Using a python script
- This has Geospatial applications
  - Lot sizes
  - Finding county areas
  - classification analysis
- As well as non Geospatial application
  - Video game design
  - Math modeling
  - 3D design



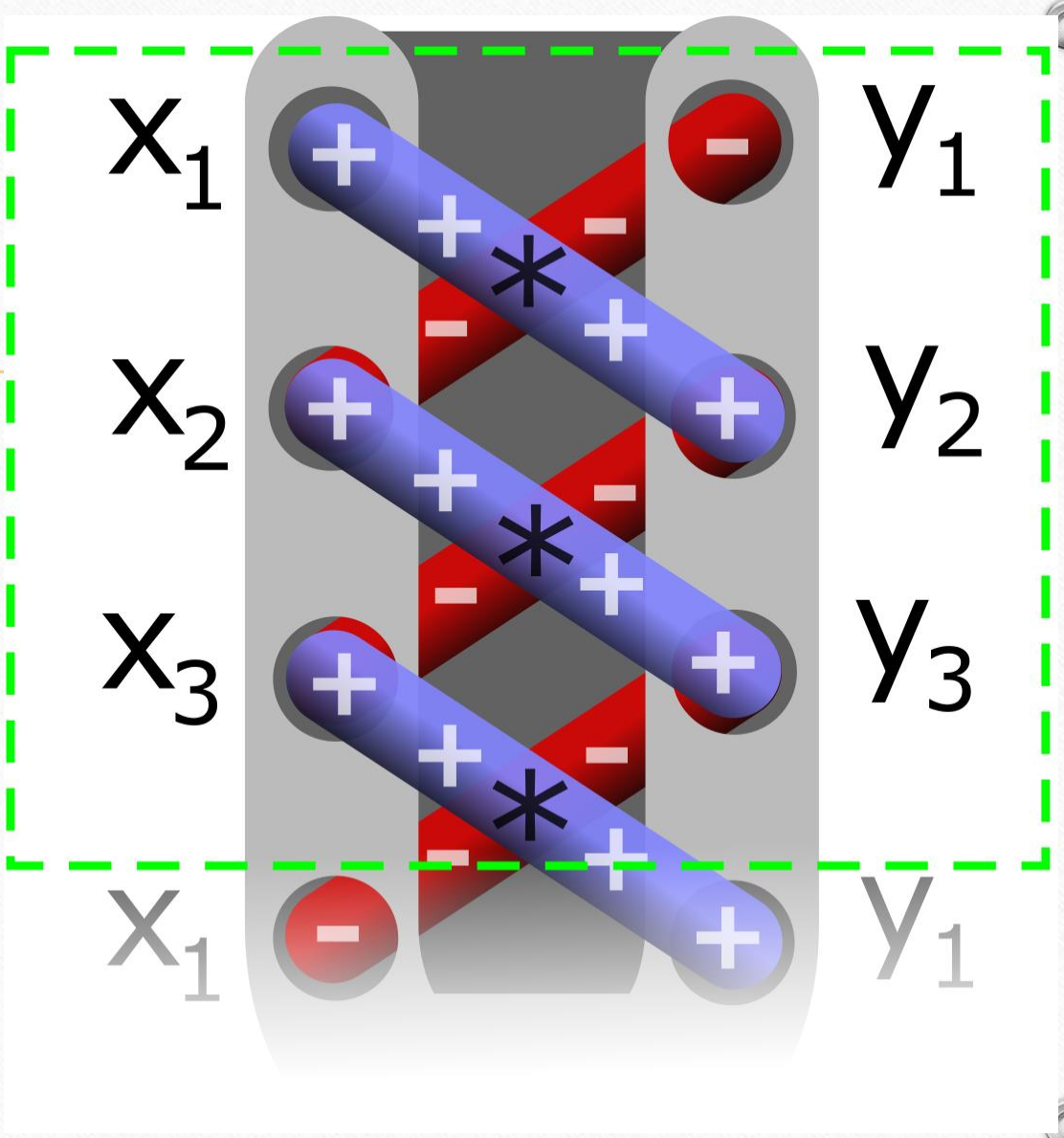


Background: Shoe lace theory

- Shoelace algorithm or Guess area formula by Meister in 1769 and by Carl Friedrich Gauss in1795

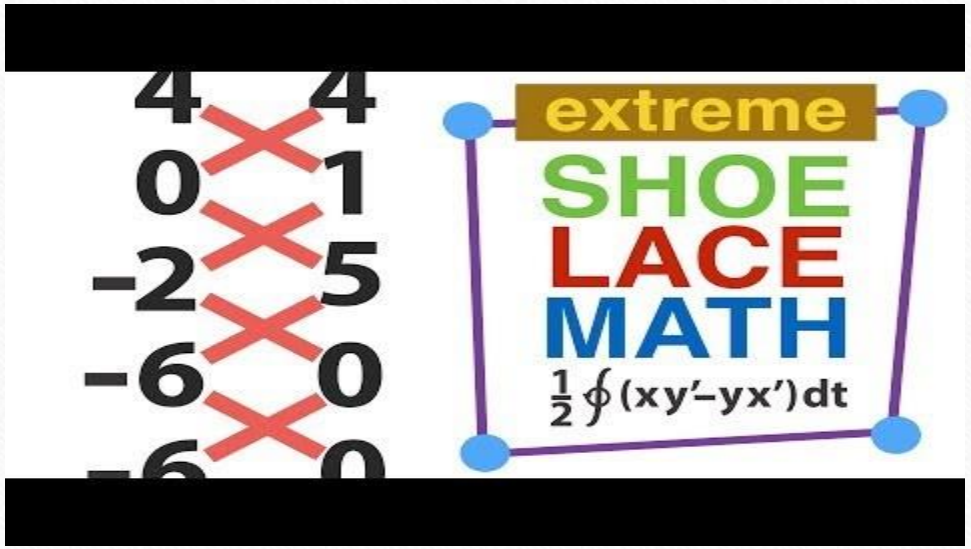
$$\mathbf{A} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

- It can be used to find the area for regular or irregularly shaped closed polygon
- Can not intersect itself



https://www.101computing.net/the-shoelace-algorithm/ https://en.wikipedia.org/wiki/Shoelace_formula

Quick Video



What I Just Said But In Python Form

```
# import file with coordinates doesn't have to be co_bounds_2 but it is for this example
import co_bounds_2
# corners is the variable for coordinates
corners = co_bounds_2.co_bounds['Gwinnett']
# start of the  sholelaces Algorithm
def PolyArea(corners):
    # total number of points in corners variable
    points = len(corners)
    # Starting point for the variables sum1 and sum 2
    sum1 = 0.0
    sum2 = 0.0
    # for loop that runs until the last coordinate
    for i in range(0, points - 1):
        #the shoe lace formula works by multiply these two together in a zig pattern and adding them together
        #corners[i][0] is the x coordinate of the first coordinate
        #while corners[i+1][1] is the y coordinate of the second coordinate

#The for loops multiply these two function together and then add the results
        sum1 = sum1 + corners[i][0]*corners[i+1][1]
        # sum2 is the same formula as sum1 except this time you start with the y coordinate of the first coordinate #and the x coordinate of the second coordinate
        sum2 =sum2 + corners[i][1] * corners[i+1][0]


    # subtract the absolute value of the two variables so that there no negative number
    # then dived by 2 will give you the area in meters
    area = ((abs(sum1 - sum2) / 2 ))
    # return takes area out of the function so that it can be used else where
    return area

# define area as part of the function
area = PolyArea(corners)
# converts meter into KM
km = (area * .000001)
#'{0:1.4f}'.format(area) makes it where the code only prints to the 4 sig figure
print ('{0:1.4f}'.format(km), 'km squared')
# I was not the one to come up with this formula I used:
#https://www.101computing.net/the-shoelace-algorithm/
#https://en.wikipedia.org/wiki/Shoelace_formula
# https://stackoverflow.com/questions/41077185/fastest-way-to-shoelace-formula
# as my references
```

First Step – import points file

```
# import file with coordinates doesn't have to be co_bounds_2 but it is for this example
import co_bounds_2
# corners is the variable for coordinates
corners = co_bounds_2.co_bounds['Gwinnett']
```

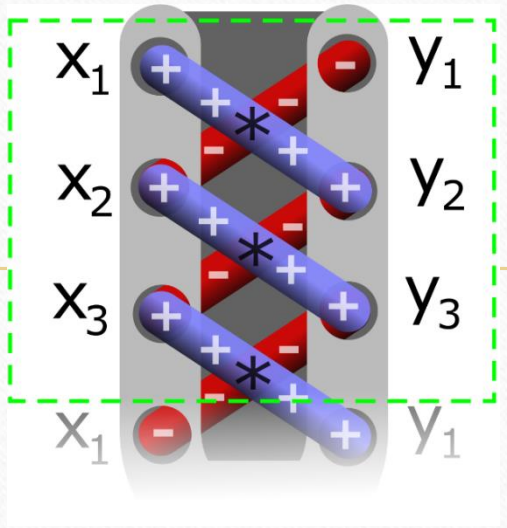
- The first step was to import the point going to be used in the function. For testing A tuple of tuples of coordinates for Georgia counties was used

Second Step Defining the Function

```
def PolyArea(corners):
    # total number of points in corners variable
    points = len(corners)
    # Starting point for the variables sum1 and sum 2
    sum1 = 0.0
    sum2 = 0.0
```

Create the For loop

- The next step is the python way to find the summation of variables



- sum1 = sum1 + corners[i][0]*corners[i+1][1] = $\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1$
- sum2 =sum2 + corners[i][1] * corners[i+1][0] = $-\sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n$

```
# for loop that runs until the last coordinate
for i in range(0, points - 1):
    #the shoe lace formula works by multiply these two together in a zig pattern and adding them together
    #corners[i][0] is the x coordinate of the first coordinate
    #while corners[i+1][1] is the y coordinate of the second coordinate

#The for loops multiply these two function together and then add the results
    sum1 = sum1 + corners[i][0]*corners[i+1][1]
    # sum2 is the same formula as sum1 except this time you start with the y coordinate of the first coordinate #and the x coordinate of the second coordinate
    sum2 =sum2 + corners[i][1] * corners[i+1][0]
```

https://www.101computing.net/the-shoelace-algorithm/

Subtract them and dived by two

```
# subtract the absolute value of the two variables so that there no negative number
# then dived by 2 will give you the area in meters
area = ((abs(sum1 - sum2) / 2 ))
# return takes area out of the function so that it can be used else where
return area
```

Printing the Results

```
# define area as part of the function
area = PolyArea(corners)
# converts meter into KM
km = (area * .000001)
#'{0:1.4f}'.format(area) makes it where the code only prints to the 4 sig figure
print ('{0:1.4f}'.format(km), 'km squared')
# I was not the one to come up with this formula I used:
#https://www.101computing.net/the-shoelace-algorithm/
#https://en.wikipedia.org/wiki/Shoelace_formula
# https://stackoverflow.com/questions/41077185/fastest-way-to-shoelace-formula
# as my references
```

Testing the Results

- Fulton 1,380 km²
- Hall 1110 km²
- Early 1,340 km²
- Newton 720 km²
- Gwinnett 1,130 km²

```
========== RESTART: P:\Python\Final\New folder\Final2.0.py ==========
1383.8951 km squared
>>> #Fulton
>>>
>>>
========== RESTART: P:\Python\Final\New folder\Final2.0.py ==========
1111.8687 km squared
>>> #Hall
>>>
========== RESTART: P:\Python\Final\New folder\Final2.0.py ==========
1337.3300 km squared
>>> #Early
>>>
========== RESTART: P:\Python\Final\New folder\Final2.0.py ==========
723.1196 km squared
>>> #Newton
>>>
>>>
========== RESTART: P:\Python\Final\New folder\Final2.0.py ==========
1131.2433 km squared
>>> #Gwinett
```

Counties	Code Area	Actual Area	Error Rate
Early	1340	1337.33	0.199651544
Fulton	1380	1383.9	0.28181227
Gwinnett	1130	1131.24	0.109614229
Hall	1110	1111.87	0.16818513
Newton	720	723.12	0.431463657

Future Developments

- The next step of my code is to create a user interface that allows a use to input there own data as right now that has to be change inside the code but not in the function

References

Bibliography

101computing. (2019). *TheShoelace Algorithm* . Retrieved from 101Computing.net : https://www.101computing.net/the-shoelace-algorithm/

Mathologer. (2017). *Gauss's magic shoelace area formula and its calculus companion - Mathologer*. Retrieved from https://www.reddit.com/r/math/comments/6gezt7/gausss_magic_shoelace_area_formula_and_its/: https://www.reddit.com/r/math/comments/6gezt7/gausss_magic_shoelace_area_formula_and_its/

Wikipedia . (2019). *Shoelace Formula* . Retrieved from Wikipedia : https://en.wikipedia.org/wiki/Shoelace_formula