```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""

This script is used to clean up shapefiles when building an ANUGA mesh
Author: Alexandra Christensen
Created: Thu Oct 15 12:33:55 2020
Updated: 12/01/2022

"""
from osgeo import ogr
from shapely.wkt import loads
import numpy as np
import math
from collections import OrderedDict

# Write a new Shapefile
def segmentize(df,triangle):
    for col,row in df.iterrows():
        geom = row['geometry']
        wkt = geom.wkt  # shapely Polygon to wkt
        triangle_length = triangle
        geom = ogr.CreateGeometryFromWkt(wkt)  # create ogr geometry
        geom.Segmentize(triangle_length)  # densify geometry
        wkt2 = geom.ExportToWkt()  # ogr geometry to wkt
        new = loads(wkt2)  # wkt to shapely Polygon
        df.loc[col,'geometry'] = new
    #df.to_file('segmented.shp')
    return df
# def morepoints(x,y):
#     newxy = np.zeros((len(x)*2,2))
#     d=0
#     for c in range(len(x)-1):
#         distance = ((x[c+1]-x[c])**2 +(y[c+1]-y[c])**2)**(.5)
#         slope = (y[c+1]-y[c])/(x[c+1]-x[c])

#         if distance<10:

#             x2 = x[c+1]+slope*10
#         newxy[d,0] = x1
#         newxy[d,1] = y1
#         newxy[d+1,0] = (x1+x2)/2
#         newxy[d+1,1] = (y1+y2)/2
#         d = d+2
   # return newxy
def lesspoints(xy,factor):
    num = len(xy)/factor
    rounded = int(len(xy)/factor)
    diff = int((rounded - num)*factor)

    removed = np.linspace(0,len(xy)+diff-factor,rounded).astype('int')
    xy = xy[removed]
    #np.delete(xy,removed,axis=0)
    return xy
```

```
    # wkt = geom.wkt  # shapely Polygon to wkt
    # geom = ogr.CreateGeometryFromWkt(wkt)  # create ogr geometry
    # geom.Segmentize(100)
    # wkt2 = geom.ExportToWkt()  # ogr geometry to wkt
    # new = loads(wkt2)  # wkt to shapely Polygon
    # return new

    # c = 0
    # removed = np.zeros((len(x),2))
    # while c<len(x)-2:
    #     removed[c,0] = x[c]
    #     removed[c,1] = y[c]
    #     c = c+factor #remove every #th point
    # removed = removed[removed[:,0]!=0]
    # return removed
    '''
   # distance = 100
    starting = math.ceil(len(x)/(distance))
    lessxy = np.zeros((starting,2))
    d=0
    for c in range(0,len(x)-1,math.ceil(distance)): #distance = increment
            lessxy[d,0] = x[c]
            lessxy[d,1] = y[c]
            d=d+1
    er = starting - d + 1
    lessxy = np.delete(lessxy,slice(d,starting),0)
    return (lessxy[:,0], lessxy[:,1])
    '''
def removedegenerate(x,y):
    removed = np.zeros((len(x),2))
    d=0
    c=0
    while c<len(x)-2:
    #for c in range(0,len(x)-2,math.ceil(distance/100)):
        a = round(((x[c+1]-x[c])**2   + (y[c+1]-y[c])**2)**(0.5),3)   #
side 1
        b = round(((x[c+2]-x[c+1])**2 + (y[c+2]-y[c+1])**2)**(0.5),3)
# side 2
        z = round(((x[c+2]-x[c])**2   + (y[c+2]-y[c])**2)**(0.5),3)
# side 3
        #z = round(((x[c+2]-x[c])**2 + (y[c+2]-y[c])**2)**(0.5),3) # side
3
        if ((a + b <= z) | (b + z <= a) | (a + z <= b)):
            c = c + 2
            d = d + 2
            removed[d,0] = x[c]
            removed[d,1] = y[c]
        else:
            removed[d,0] = x[c]
            removed[d,1] = y[c]
            d = d + 1
            c = c + 1
        #test = round(a+b,3)
        #if test==z:
```

```
        #    c=c+2
        #else:

    #print(len(removed))
    removed = removed[removed[:,0]!=0]
    #print(len(removed))
    # removed2 = np.zeros((len(removed),2))
    # d=1
    # c=1
    # x = removed[:,0]
    # y = removed[:,1]
    # while c<len(x)-2:
    # #for c in range(0,len(x)-2,math.ceil(distance/100)):
    #     a = round(((x[c+1]-x[c])**2   + (y[c+1]-y[c])**2)**(0.5),3)   #
side 1
    #     b = round(((x[c+2]-x[c+1])**2 + (y[c+2]-y[c+1])**2)**(0.5),3)
# side 2
    #     z = round(((x[c+2]-x[c])**2   + (y[c+2]-y[c])**2)**(0.5),3)
# side 3
    #     #z = round(((x[c+2]-x[c])**2 + (y[c+2]-y[c])**2)**(0.5),3) #
side 3
    #     if (a + b <= z) | (b + z <= a) | (a + z <= b):
    #         c = c + 1
    #         d = d + 1
    #     else:
    #         removed2[d,0] = x[c]
    #         removed2[d,1] = y[c]
    #         d = d + 1
    #         c = c + 1
    #     #test = round(a+b,3)
    #     #if test==z:
    #     #    c=c+2
    #     #else:
    # removed3 = np.zeros((len(removed2),2))
    # d=2
    # c=2
    # x = removed2[:,0]
    # y = removed2[:,1]
    # while c<len(x)-2:
    # #for c in range(0,len(x)-2,math.ceil(distance/100)):
    #     a = round(((x[c+1]-x[c])**2   + (y[c+1]-y[c])**2)**(0.5),3)   #
side 1
    #     b = round(((x[c+2]-x[c+1])**2 + (y[c+2]-y[c+1])**2)**(0.5),3)
# side 2
    #     z = round(((x[c+2]-x[c])**2   + (y[c+2]-y[c])**2)**(0.5),3)
# side 3
    #     #z = round(((x[c+2]-x[c])**2 + (y[c+2]-y[c])**2)**(0.5),3) #
side 3
    #     if (a + b <= z) | (b + z <= a) | (a + z <= b):
    #         c = c + 1
    #         d = d + 1
    #     else:
    #         removed3[d,0] = x[c]
    #         removed3[d,1] = y[c]
```

```
#         d = d + 1
#         c = c + 1
#     #test = round(a+b,3)
#     #if test==z:
#     #    c=c+2
#     #else:
# print(len(removed3))
# removed3 = removed3[removed3[:,0]!=0]
# print(len(removed3))

# res = OrderedDict()
# for point in zip(removed[:,0],removed[:,1]):
#     res.setdefault(point[:2],point)
# mypoints = np.array(list(res.values()))
# print(len(mypoints))
return removed
```