**Pseudocode**

Get the user input: *County* shapefile

**Intersect** with *GLO\_Shoreline\_Mask* shapefile

Get the user input: Intersect output

Automatically generate field " Owner Name "

Set the expression = """Reclass({}, {}, {}, 3, 2, 1)""".format(!ValueFieldName!, threshold1, threshold2)

Define the codeblock = """def Reclass(field "OwnerName", threshold1, threshold2, x, y, z):

Define the expression1, expression2, expression3 in the codeblock

**if** expression1:

return x

**elif** expression2:

return y

**else**:

return z

**Add Field** in the attribute table

**Calculate Field** of Owner Name (Similar process repeated 3 times to generate and calculate 3 new field: OwnerName, OwnerAddress, PropertyAddress)

Search cursor?

Write to txt file?

**MakeFeatureLayer** to get *Countymap* layer

**CopyFeatures**

Get the user input: existing Map Document (.mxd)

**Add layer** to existing Map Document as “TOP” layer and save the .mxd file

**Save a copy** to a new Map Document as” *CountyMap.mxd*”

del mxd

**Add 3 new fields** in attribute table (.dbf)

intersected shapefile

3 new fields:

1. OwnerName
2. OwnerAddress
3. PropertyAddress

**Add layer to existing Map Document**(.mxd) as “**TOP**” layer, and assign a new name as” *Countymap*” layer, **save the .mxd**

*County*

shapefile

**Intersect** with *GLO\_Shoreline\_Mask* shapefile

Export to intersected shapefile

Start

**Save a copy** to a new Map Document as “*CountyMap.mxd*”

End

*CountyMap.mxd* file

**Work Flow Chart of the Parcel Tool**

**Field Calculator**

**Resources & tips:**

<https://geonet.esri.com/thread/77437>

"ADDRESS" LIKE '%|%%' escape '|'

**Psudocode:**

import arcpy

arcpy.env.workspace = "C:/data/…"

# Owner\_Name

arcpy.AddField\_management("Owner\_Shoreline.dbf", "Owner\_Name", "TEXT")

arcpy.CalculateField\_management("Owner\_Shoreline.dbf", "Owner\_Name", expression, "PYTHON", codeblock)

expression = """ReclassON(!file\_as\_na!, !addr\_line1!, !addr\_line2!)"""

codeblock = """def ReclassON(infield, infield2, infield3):

expression1 = infield LIKE '%&' OR infield2 LIKE '&%'

expression2 = (infield LIKE '%&' OR infield2 LIKE '&%') AND (infield2 LIKE '%&' OR infield3 LIKE '&%')

expression3 = infield = ' '

**if** expression1:

return infield + ' '+infield2

**elif** expression2:

return infield + ' '+infield2+' '+infield3

**elif** expression3:

return infield2

**else**:

return infield

# (Check "file\_as\_na" doesn’t have "&" in the end +"addr\_line1"/ "addr\_line2" doesn’t have "&" in the beginning)

"""

# \*Manually situation\*: both "file\_as\_na" and "addr\_line1" doesn’t have "&", but both is part of Owner Name. => Maybe run the code first, then check the result of situations only return "file\_as\_na"?

# Owner\_Address

arcpy.AddField\_management("Owner\_Shoreline.dbf", "Owner\_Address", "TEXT")

arcpy.CalculateField\_management("Owner\_Shoreline.dbf", " Owner\_Address ", expression, "PYTHON", codeblock)

expression = """ReclassOA(!addr\_line1!, !addr\_line2!, !addr\_line3!, ! addr\_city!, ! addr\_state!, !zip!)"""

codeblock = """def ReclassOA(infield, infield2, infield3, infield4, infield5, infield6):

expression1 = infield LIKE 'C/O%'

expression2 = infield2 LIKE 'C/O%'

expression3 =

**if** expression1:

return infield+ ", "+infield2+ ", "+ infield3 + ", "+ infield4+ ", "+ infield5+ ", "+infield6

**elif** expression2:

return infield2+ ", "+ infield3 + ", "+ infield4+ ", "+ infield5+ ", "+infield6

**elif** expression3:

return

**else**:

return infield+ ", "+infield2+ ", "+ infield3 + ", "+ infield4+ ", "+ infield5+ ", "+infield6

"""