GIS 6103

Programming for GIS

Assignment 5 Part B

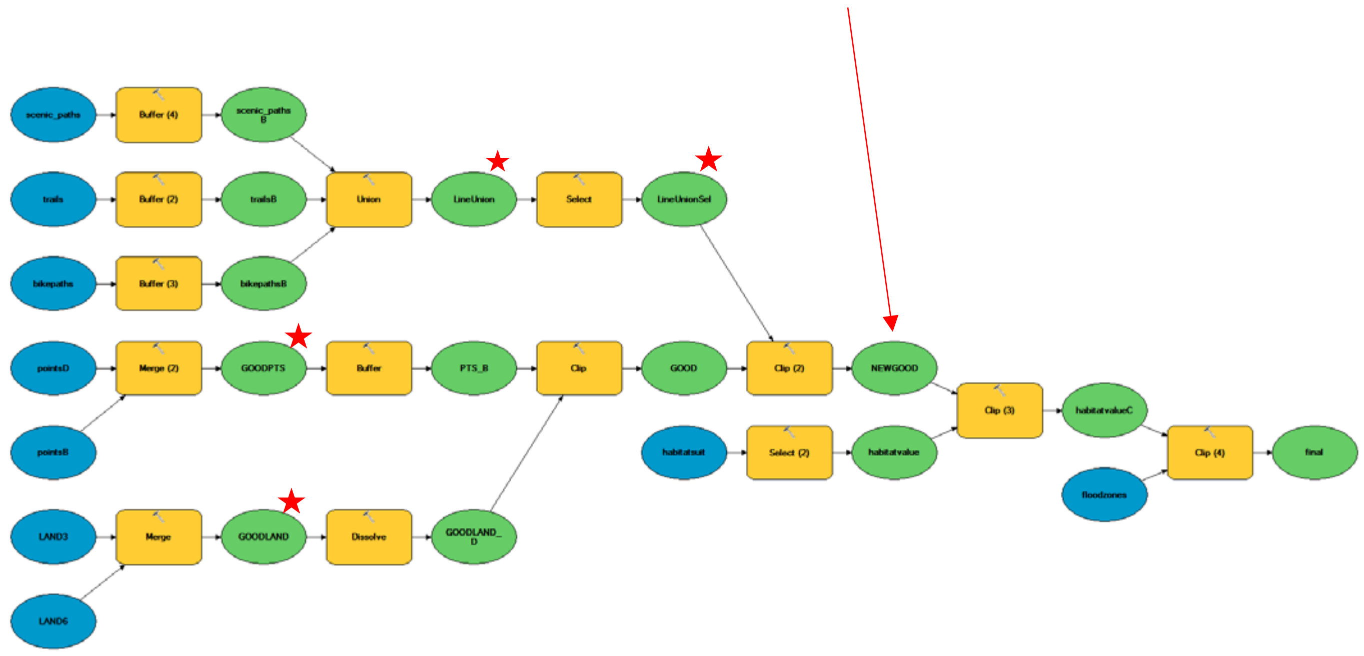
<u>Overview</u>

Following your great success with the monitoring sites analysis on your last contract, you have been contacted by an environmental consulting company to do some work for them. They want to produce a series of feature classes where each feature class shows locations that have a particular habitat score and those polygons can also inform whether they are inside or outside the floodplain (as an attribute). They want to do this for various habitat suitability scores (3, 6, 7, 8). However, here is the catch, they want only the locations that fall within a 'designated area' and since they are not sure what that designated area should be, they want to define it in various ways and produce the feature classes for each definition.

The definition of the designated area is:

- On the correct type of land use (does not vary)
- Within a certain distance (to vary) of field stations
- Within a certain distance (to vary) of **<u>any two, but only two</u>**, of scenic paths, trails, and bike paths.

You have been provided with a model for the basic analysis and it is reproduced below as a graphic. The feature class NEWGOOD below would represent the designated area for this 'run'.



From the model graphic, you will notice that the feature classes GOODLAND (the correct land use) and GOODPTS (the correct field stations) result from merging some feature classes. However, in both of these cases, the relevant feature classes that go into the merge need to be found from the data you have been supplied (and don't assume the ones shown on the model are the right ones).

In the case of GOODLAND, what you know is that the names of the correct input polygon feature classes all begin with the letters "LAND" and they have a field named "DATA2015". In the case of GOODPTS, you know that the correct input point feature classes have a shape type of 'Point' and have 10 or fewer point features in them.

Once GOODPTS is produced, you will notice from the model that the points (field stations) are buffered. However, the buffer distance will be varied using values of 1250, 1500, and 1750 meters.

Similarly, you will notice that the scenic paths, the trails, and the bike path line features classes are each buffered before the buffers are brought together using the Union tool. The buffer distances for these features are also varied using values of 750, 1000, and 1250 meters. However, note that we don't want all the different combinations of using these values, we simply use one of the values for all 3 buffers, and then change the value.

The resulting feature class from the Union tool is LINEUNION in the model graphic above. Notice that a Select (by attributes) tool is used on LINEUNION to produce LINEUNIONSEL. This is the tool that limits the area to just 2 of the 3 buffers (although which 2 can vary). You should note that there are therefore 3 possible combinations of 2 from 3. *(Side note: this part of the script will require a little thought and trickery).*

So, in terms of 'looping', you need to consider that you need to do this analysis for 4 different habitat scores, 3 different values for the point (field station) buffers, 3 different values for the line (paths/trails) buffers, and then there are 3 different ways to define 2 from 3 for the line buffers. In other words, you will produce 108 (4 x 3 x 3 x 3) final feature classes for the company. Your looping structure should use the habitat scores as the outer loop, the point buffers as the first inner loop, then the line buffers as the second inner loop, and finally the selection of line buffers to use as the third inner loop. Conceptually, you have:

```
Habitat score loop
    Point Buffer loop
        Line Buffer loop
            Line Buffer Selection loop
```

Your final feature classes should be named, and so individually identified, based on the example below:

H7P1250L750_1

The example above corresponds to habitat score 7 (H7), point buffer of 1250 meters (P1250), line buffer of 750 meters (L750) and then an identifier of one of the three ways to choose 2 from 3 line buffers (_1).

I have provided a script with comments to help guide you, and that script also has all the tool statements exported from the model, <u>though not edited</u>. However, make sure you read through the notes and tips below.

Notes and Tips

- All the feature classes produced by your script, except for the final feature classes, should be written to ptBtemp.gdb. The final feature classes should be written to ptBresults.gdb. However, except for the part of the scripts where you check/delete feature classes in ptBresults.gdb at the start, the environment workspace should be set to partB.gdb. What this means is that you need to make sure that if you run your model, you comment out the parts you are not working on or testing. You don't need to be waiting on tools running or generating errors from parts of the script you haven't got to yet.

- Similarly, when developing and potentially running your model, restrict your 'looping' to just one value or element in the lists you create for the looping structures. For example, the habitat scores need to loop thru the values 3,6,7,8 and so you create a list for those values. But, while developing, create a separate list with just one of those values and use that (its fine to loop thru using **for** just one value!). Once you know the script works, substitute in your full list. This also applies to the point buffer and line buffer values. However, it doesn't apply to the *line buffer selection looping* since that needs to be done in such a way that all scenarios are included.

- Make good use of print statements to verify you are producing what you think you are producing, especially when doing operations like listing feature classes. In regards to this and since I am a wonderful professor and a super nice guy, I will let you know that you need to find 4 feature classes for the GOODLAND merge and 2 for the GOODPTS merge.

- In this exercise, you 'know' that your starting data is in partB.gdb (i.e. you know there are no feature datasets within it), so you don't need to *walk* the talk ;).

- all new feature classes produced by the script should be written to, and retrieved from in some cases, the correct locations. In other words, create string variables with paths that enable this – just don't forget to use those string variables where necessary!

- Remember that the tools as exported from the model to the script (I have done this for you) do not necessarily follow the same sequence when you start 'inserting' them into your script. Make sure you follow the logic of the model (and your own script) in ordering the tools.

- The one part of this scripting task that is a little tricky involves the line buffer selection part. Remember that this is where you need to be able to select areas that are in ***just*** 2 of the 3 types of line buffers, but you do need to produce feature classes for all the 3 ways of doing this *2 from 3*. This is done using the fields that will exist in the LINEUNION feature class. As a tip, you only need one Select statement to achieve this. Try to think thru how this could be done (tips: looping, variable substitution, conditional statement) by considering what values the relevant fields in LINEUNION can possibly take.