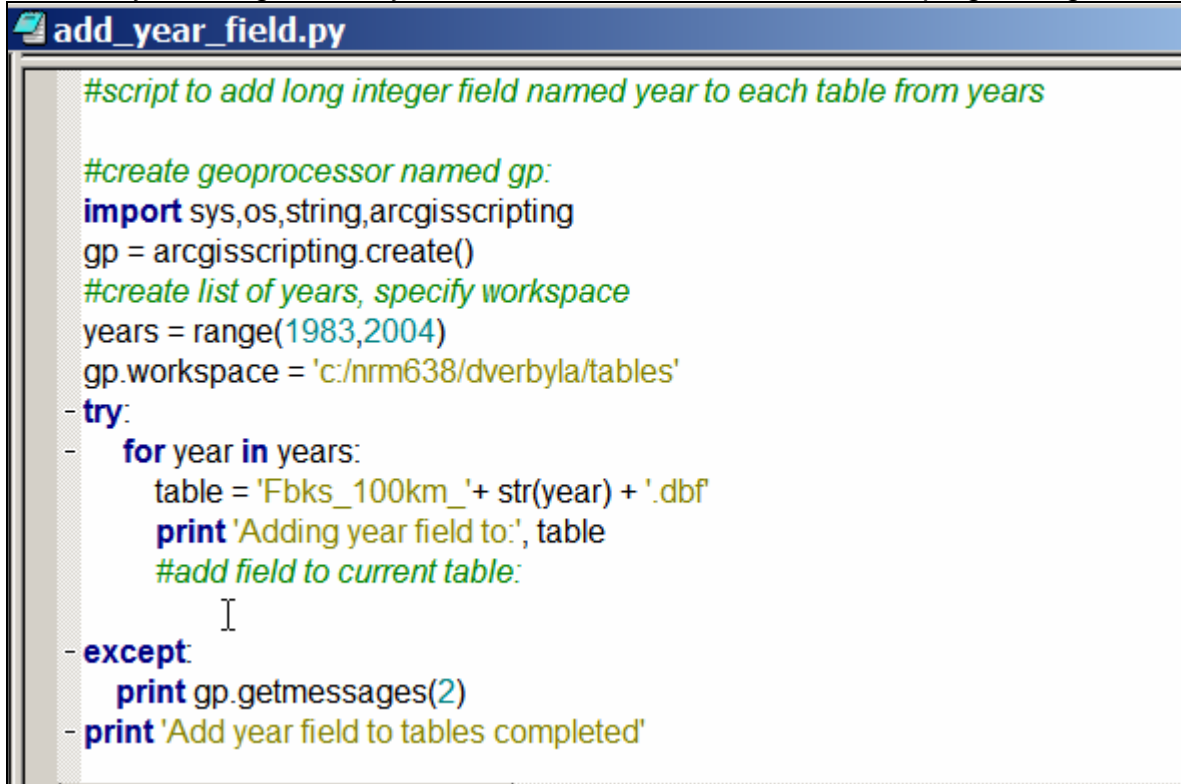


## Introduction to Python Geoprocessing

Due by Friday 6pm

Download and an executable winzip file called **tables.exe** from <http://nrm.salrm.uaf.edu/~dverbyla/nrm638/data>

- 1) Create a python script that will add a field called year to each table. Do this by creating a list of years from 1983 to 2003 and then looping through the list.



```

add_year_field.py

#script to add long integer field named year to each table from years

#create geoprocessor named gp:
import sys,os,string,arcpy
gp = arcpy.CreateGP()
#create list of years, specify workspace
years = range(1983,2004)
gp.workspace = 'c:/nrm638/dverbyla/tables'
- try:
-   for year in years:
-       table = 'Fbks_100km_' + str(year) + '.dbf'
-       print 'Adding year field to:', table
-       #add field to current table:
-           |
-   except:
-       print gp.getMessages(2)
-   print 'Add year field to tables completed'
  
```

Use the above script as a starting point, you simply have to add the correct line for using the AddField tool.

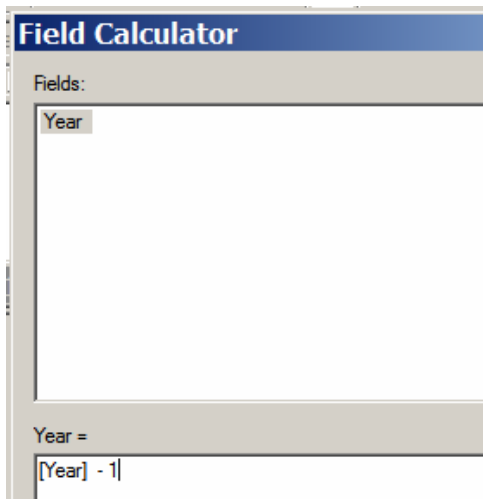
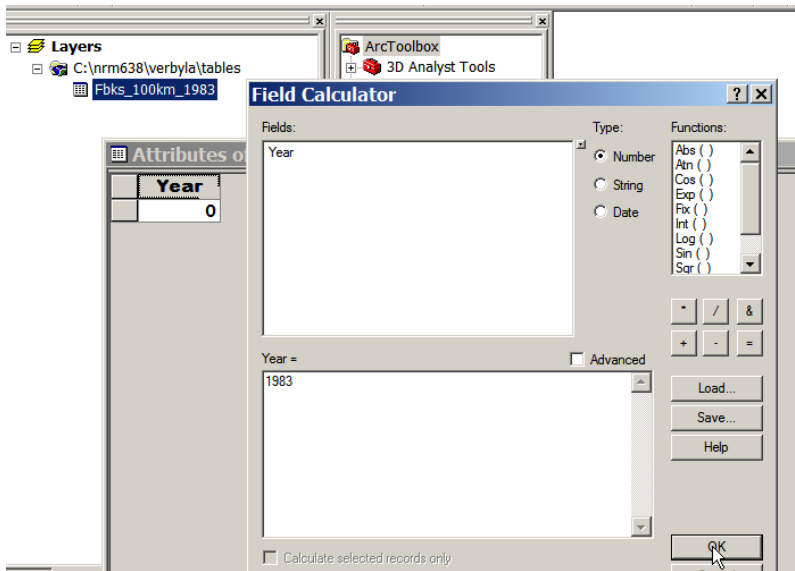
- 2) Create a python script that will calculate the appropriate year for the year field in each table, using the loop from years 1983 to 2003. The CalculateField tool can use Python or Visual Basic calculation expressions as follows:  
`gp.CalculateField_Management(table,field,expression,"VB" or "Python")`

For example:

```

>>> gp.workspace = 'c:/nrm638/dverbyla/tables'
>>> table = 'Fbks_100km_1983.dbf'
>>> gp.calculatefield_managment(table,'Year',1983,'VB')
  
```

The above is equivalent in Arcmap as:



This can be done in scripting as follows:

```
>>> gp.CalculateField_Management(table, 'Year', '[Year] - 1', 'VB')
'Fbks_100km_1983.dbf'
```

And Python expressions and variables can be used as follows:

```
>>> count = 1983
>>> gp.CalculateField_Management(table, 'Year', 'count', 'PYTHON')
'Fbks_100km_1983.dbf'
```

Email me ([D.Verbyla@uaf.edu](mailto:D.Verbyla@uaf.edu)) your two python scripts by Friday.