



FEBRUARY 10, 2017

USING PYTHON INSIDE ARCMAP

TOPIC 5

VINCENT A DINOTO JR,
JEFFERSON COMMUNITY AND TECHNICAL COLLEGE
Vince.dinoto@kctcs.edu

Contents

Figures.....	2
Introduction.....	3
Tools and concepts	3
Label	3
Steps for Simple Labeling using Python Script.....	4
Advanced Labels.....	8
Assignment 5.1	10
Field Calculator and Python Script.....	11
Example.....	12
Points to Remember	14
Assignment 5.2	15
Index	16

Figures

Figure 1: Layer Property Window	4
Figure 2: Defining the Parser, Field and Advanced Editing	5
Figure 3: Saving the code.....	6
Figure 4: Results of the Label Expression	7
Figure 5: Advanced Labeling Results	9
Figure 6: Field Calculator	12
Figure 7: Defining a function.....	13
Figure 8: Pre-Logic Script	13

Introduction

In this module, the use of Python scripts inside specific Esri ArcMap tools will be demonstrated. In addition to the Python scripts, which will be developed, xml code will also be used and explained. The Label function will be expanded beyond simple labelling into functions that are more dynamic. In the Field Calculator, **If** statements will be used to change the value of an attribute.

This work will be performed without the use of an IDE or the Python Window of Esri ArcMap. The scripts will be written directly with the tool. The code used could be written in an editor, copied, and pasted into the tool, but for this example, the code will be composed directly. The length of the code can be similar to any other code generated and has no parameters that require it to a certain length. If the code is relatively short it should be written directly in the tool, but for lengthier and more complicated code, the use of an editor should be considered.

Tools and concepts:

- If/elif/else
- Return
- Def
- Xml Code

Label

Esri ArcMap must be opened and the file used in this example will be the KY_Railroad file and it should be visible on the map. This file contains railroads located in the Commonwealth of Kentucky and is the only required information for this example. Additional files such as a state or county map of Kentucky might be useful. For this example, an online background map should not be used because it will obscure the data being displayed and modified.

Steps for Simple Labeling using Python Script

1. Open the Layer Properties window and select the label tab, select the expression button to open the label expression window.

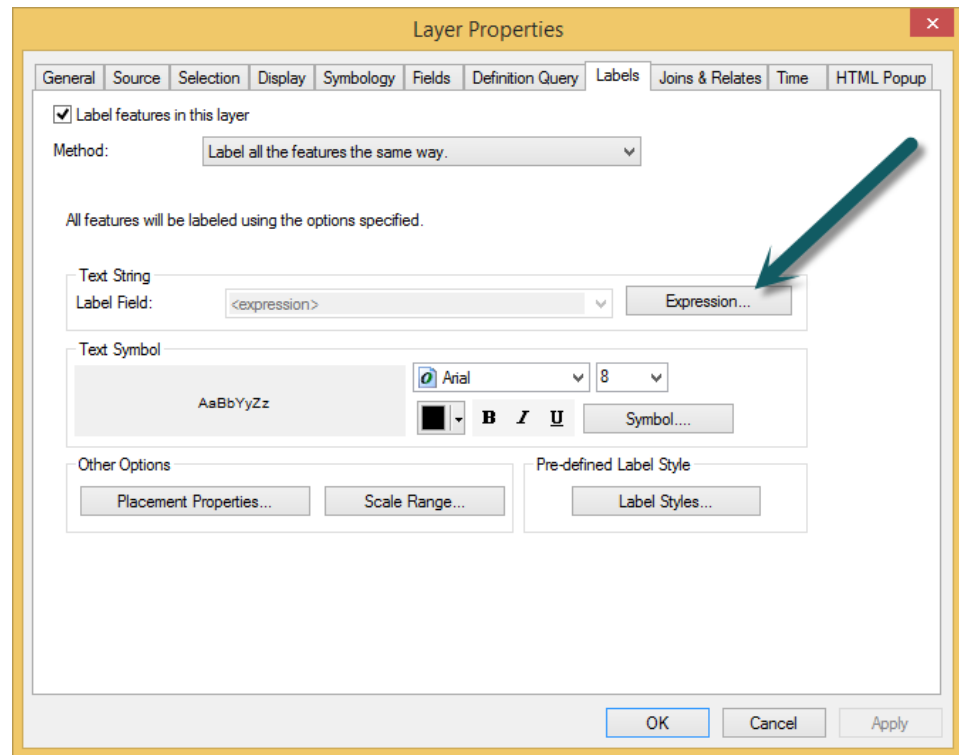


Figure 1: Layer Property Window

2. In Figure 2, three selections will be made and will need to be completed in the following order.
 - a. The Parser must be selected to be Python, this is done by using the pull down arrow, which is shown with the lower arrow.
 - b. Select the check box labeled Advanced, this is normally not selected, this is shown with the middle arrow.
 - c. In the fields table double click on the RROWNER field. The expression results will be similar to Figure 2.

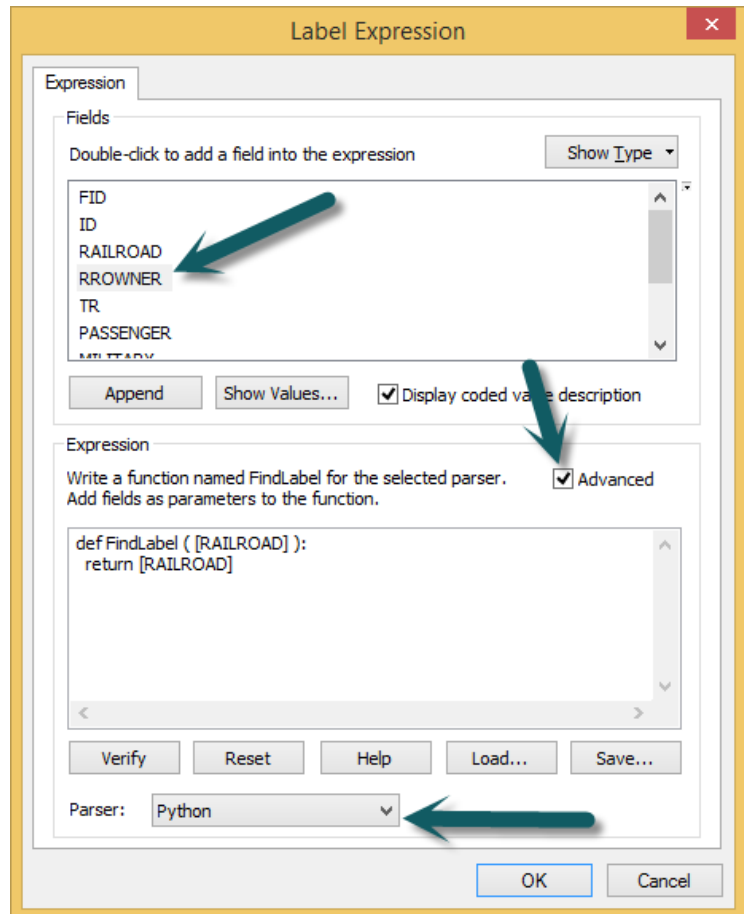


Figure 2: Defining the Parser, Field and Advanced Editing

- The final step is to write the Python script in the expression window, the code is shown in Table 1. The first column of the table shows the code and the second column explains the purpose of the code. Saving the code is preferable so it can be used again see Figure 3. If the code is not saved, it will execute but may require rewriting if it is to be used again. When Ok is pressed the code will be executed and the property window will close. See Figure 3: Saving the code

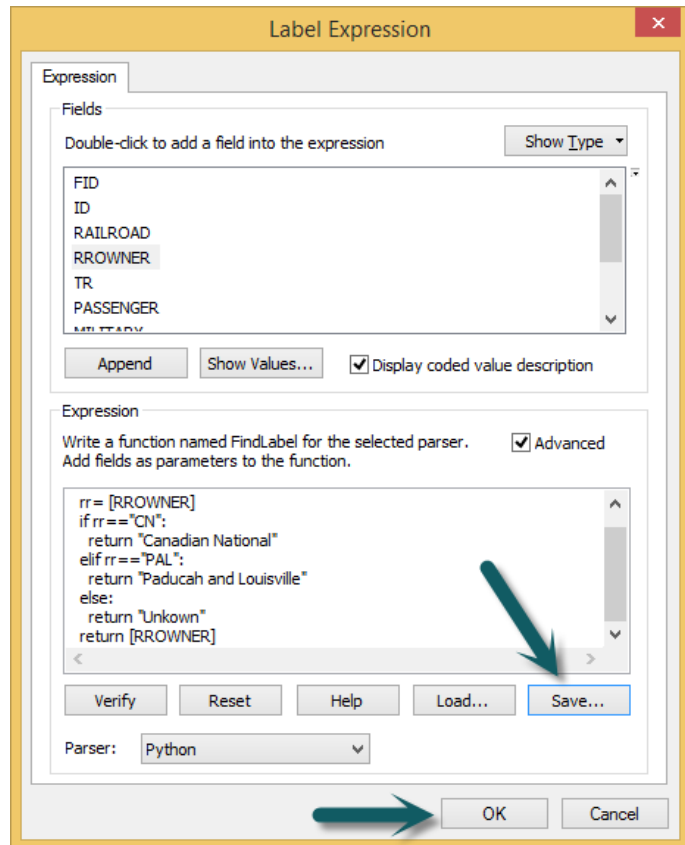


Figure 3: Saving the code

Code	Description
<pre>def FindLabel ([RROWNER]): rr= [RROWNER] if rr=="CN": return "Canadian National" elif rr=="PAL": return "Paducah and Louisville" else: return "Unknown" return [RROWNER]</pre>	<ul style="list-style-type: none"> RROWNER is a field in the KY_Railroad file, which defines who owns the tracts. The list variable rr represents the values located in the RROWNER field. The If statement is used to determine if the railroad track owner is equal to CN and only equal to CN. The value Canadian National is returned and placed on the map. If the expression is not true, then the script drops to the Elif statement to see if rr is equal to and only equal to PAL, if true then Paducah and Louisville is placed on the map. If neither of these two expressions are true then the track is labeled Unknown in the Else statement.

Table 1: Code 1 Full Name

When the attributes of the railroad file are displayed, they will be abbreviated; not the full name. If the abbreviated names are used on a map, the user might not know what the abbreviated name means for all the railroads. For example, while a national railroad line like CN (Canadian National) might be known, a regional line like PAL (Paducah and Louisville) might not. Therefore, on the produced map, the names displayed should be the full name and not the abbreviated name.

Additional **Elif** statements could be written for other railroads in Kentucky, review the attributes to see what other railroads are contained in the file. In the code shown in the first column of the table, two new commands are introduced. The **def** statement is a defining function. There must be a colon placed at the end of the statement. In this code, the **FindLabel** statement is used to locate the field named **RROWNER** as long as the indentation is used. **FindLabel** is a reserved name for the label window. The other statement was the return statement. In many ways, this statement is similar to a **print** command, which shows a value on the console. With the **Return** command, the value is added to the map.

The result of running the expression is displayed in Figure 4.

Note that there are tracks labeled: Unknown, Paducah and Louisville and Canadian National, only a small part of the state is visible. No content has been changed in the attribute table only how the information is displayed has been modified.

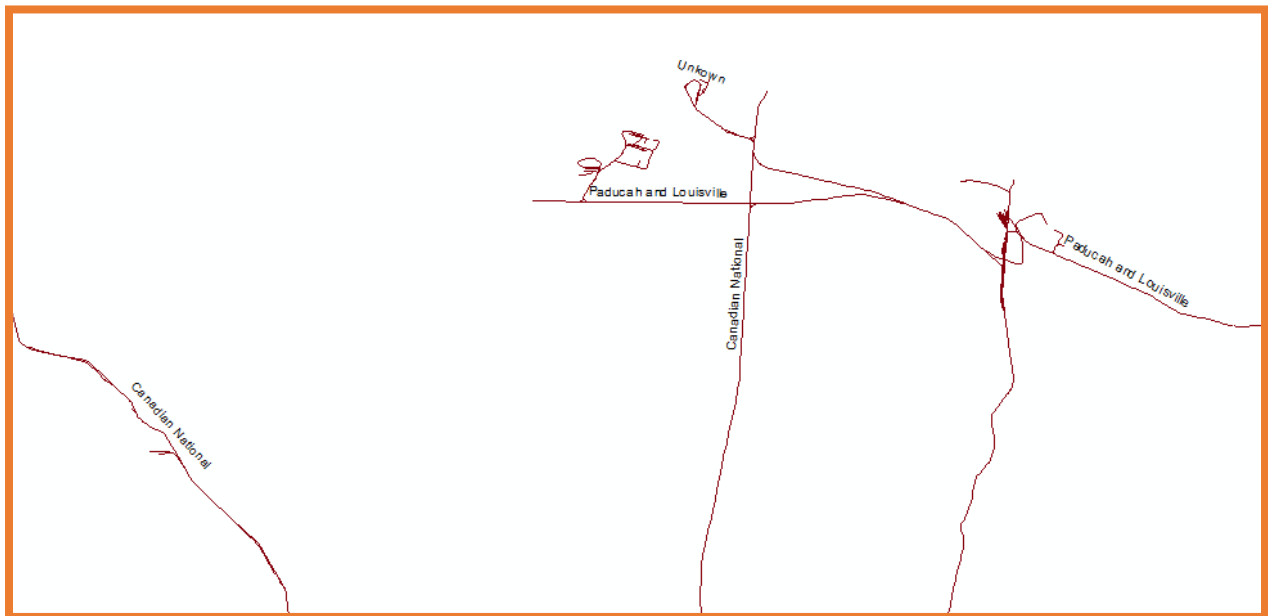


Figure 4: Results of the Label Expression

Advanced Labels

In the previous example, Python Code was placed in the expression window. In the Label Expression window, Python scripts can have embedded code. Xml code is commonly used in web page design. The xml code can be used to control such features as color, font size, font face, along with features such as bold, underline, mark through, etc.

The format for xml code is "<tag>" + text + "</tag>", xml code always is opened with a < > and closed with </>. To find a list of potential tags that can be used in Esri ArcMap use the following hyperlink: <http://desktop.arcgis.com/en/desktop/latest/map/working-with-text/formatting-tags-available-in-arcmap.htm>

To explore these advanced labeling features the same code used in the previous example will be the starting point and then modified as seen in the second table. The changes that will be made: The labeling for the Canadian National railroad will be made red and a larger font size than the default font. The labeling for the Paducah and Louisville railroad will be made bold and have a green color.

The only changes that have been made to the two lines for Canadian National railroad and Paducah and Louisville railroad. It is very important to notice in the format on how the brackets are used.

Code	Description
<pre>def FindLabel ([RROWNER]): rr= [RROWNER] if rr=="CN": return "<CLR red='255'><FNT size = '14'>" + "Canadian National" + "</FNT></CLR>" elif rr=="PAL": return "<BOL><CLR Green='200'>" + "Paducah and Louisville" + "</CLR></BOL>" else: return "Unknown" return [RROWNER]</pre>	<ul style="list-style-type: none">• CLR is used to represent the color and there are three-color names red, green and blue (RGB). The number represents the intensity of the color with 255 being the maximum and 0 being the minimum. More than 65,000 unique colors can be generated (this is known as eight-bit color).• FNT represents font and then the size is noted, these are the same point values as used with word processing.• BOL represent bold and has no parameters.• Therefore, the first return line after the If statement has given a maximum value to red, and since the other two colors are not represented therefore they will have a minimum value of zero. The font size has been made to be 14, which is larger than the default

	<p>size. At the end of the line note that the font and color commands are closed using the forward slash. The ordering of the closing is important and they must be done in such a way that the last command opened is the first command closed. Thus, color was opened first and closed last.</p> <ul style="list-style-type: none">• In the next return statement, which is after the elif, the code makes the font bold next the color is set. Only green color is noted, so the other two colors are a minimum, but the green color is not stated at full intensity since a value of 200 is used. Note how the closing of the commands are completed.
--	---

Table 2: Code, using XML

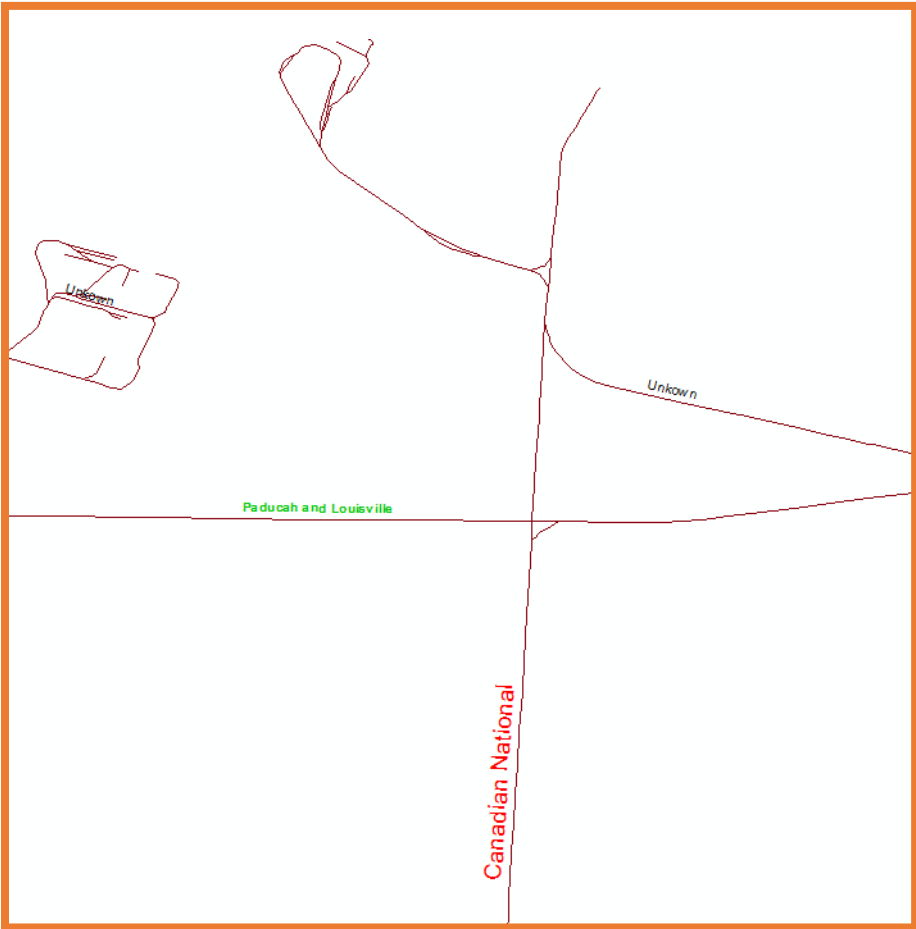


Figure 5: Advanced Labeling Results

The label for the Canadian National railroad now appears red and that the size of the font is larger than the other labels. In addition, the Paducah and Louisville railroad is green and bold. The boldness is a little hard to discern. The Unknown label was unchanged and the default parameters were used.

In both of these label color changes, only a single color was used while the depth of the color was different. It is important to experiment with how using two or three colors together will change the color that is displayed. If all three

colors were set to the maximum the color would be white. If all three colors are set to the minimum value, the color is black. For example, pure yellow would be R=255, G=255, B=0. More than 65,000 colors can be defined using 8 bit RGB. For a better understanding a good website is:

http://rapidtables.com/web/color/Web_Color.htm?T1=255+255+255#color%20table

If more than one color is being used after the return statement do not put a comma between the colors.

CMYK (cyan, magenta, yellow, black) can also be used, but the scale range is 0 - 100.

For example, (partial script) return "< CLR yellow = '95'...

Assignment 5.1

In this assignment, use the same file that was used in the technical skills lesson, i.e. KY_railroads. The assignment requires the classification of all railroads in Kentucky, using the ownership attribute. Make the font size 14 and the font face Arial for all of the railroad labels. The font color of each railroad should be unique, which will require the use of more than one color tag (RGB) in the same code statement. Provide your instructor with both the script and map image.

Abbreviation	Railroad
TKEN	Tennken Railroad
WTNN	West Tennessee Railroad
KWT	K.W.T. Railway
USG	Union Station Gateway

Table 3: Railroad Names



Field Calculator and Python Script

A new field will be created in an attribute table and an expression calculated using the values contained within other fields of the table. The values placed in the calculated field will be done using a Python script; there are other methods this task could be accomplished without using Python. A method to complete the task using a Python script will be demonstrated, this might not be the most efficient method, depending on the amount of data. The user in the production environment must determine the most efficient method to complete the required tasks.

The script will be written directly in the field calculator but could have been written in an IDE, copied and pasted into the field calculator window. It is important to remember that the attribute table is displaying the information contained in a database and not a spreadsheet. In a spreadsheet, the equation is stored in each cell and a new calculation is completed every time the data is utilized. In a database, a calculation is done once and the new value is written into the cell. The equation is not stored as part of the cell content and thus a new calculation not completed each time the information is accessed.

In this example, a data set of U.S. colleges will be used. The file should be loaded into Esri ArcMap, and then open the attribute table. The process requires that a new field be manually created. Make sure no special characters are used in the field name. The field calculator should be opened for the newly created field. In this example, a looping statement will be used. Remember that a colon is placed at the end of the statement, in the IDE, the colon causes an indentation of the next line of code, but in the scripting window the indentation must be handled manually, always use the same number of spaces for the same level of indentation.

Example

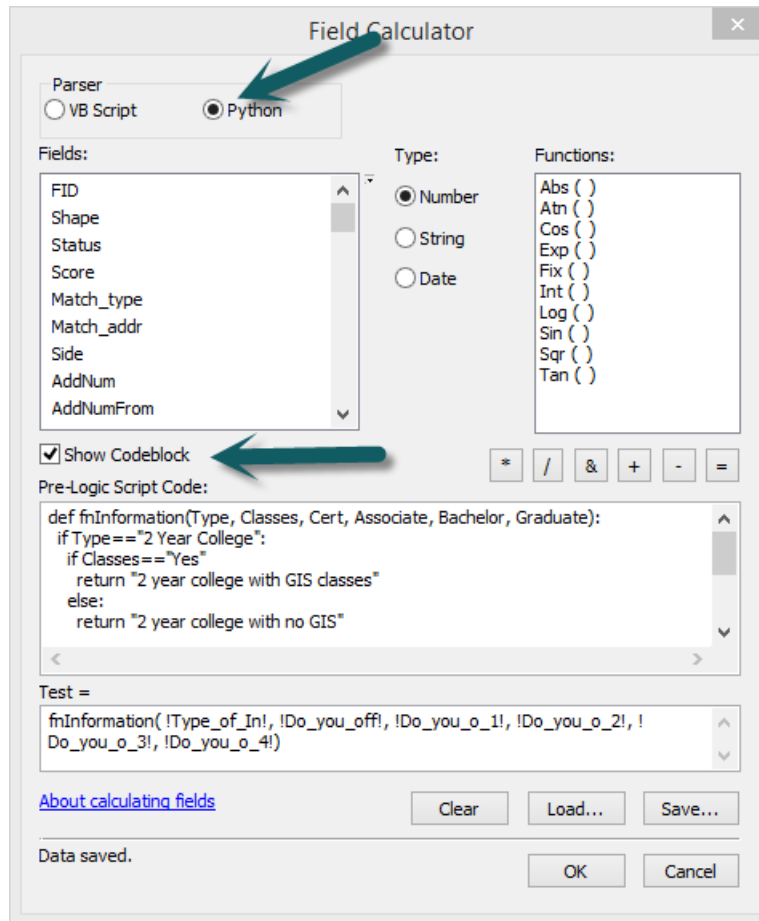


Figure 6: Field Calculator

In this example, a classification will be performed on the offering of geospatial technology classes and the type of institution. The value placed in the new field will depend on the values of other attributes in the table. Nested **if** statements will be used to modify the content of the field. While this process could be completed manually it would take extensive time, which could be impacted by typographical errors, the code will create results that are always identical for each cell with the same parameter. The new field, that was manually created, is a text field and has been given the name Test. Once the Field Calculator Window is open, select the Python Parser and select the check box Show Codeblock. See Figure 6: Field Calculator

, for details, ignore the code, it will be discussed in the next steps.

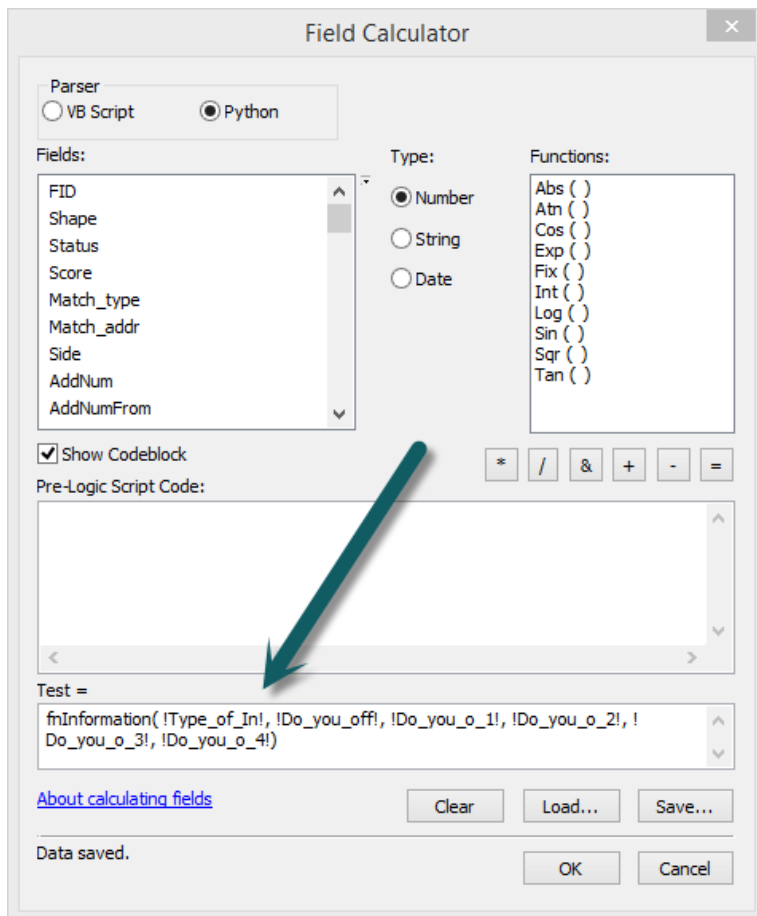


Figure 7: Defining a function

In the lower box seen in Figure 7:
Defining a function

a function will be defined called **fnInformation**. Inside the parenthesis are different field names. It is important that the field names be inputted by double clicking on the field name in the Fields box in the upper left corner of the Field Calculator window. The name for the fields were not properly titled when this file was created. Make sure a comma is placed between each of the field names. In the Pre-Logic Script, **fnInformation** is defined. The Pre-Logic script must have a one to one correspondence with the function (each having the same number of elements) that was defined. See Figure 8, for the definition of the Pre-Logic Script.

The first element represents the type of institution such as community or university. The next element defines if the institution offers GIS courses. The full script for the process is shown in the table. Nested **if** statements were used to solve this problem, it could have been created in a different way and only a single **if** statement used. The **Return** statements will populate the field **test** with what is enclosed inside of the quotes. It is important when the field **test** was created that sufficient characters were allowed so that the complete statement can be saved.

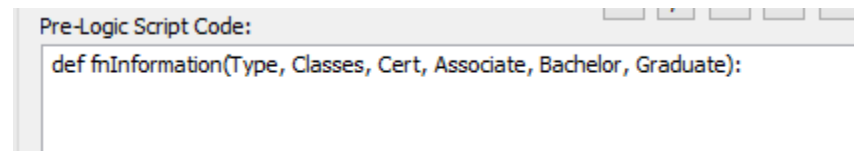


Figure 8: Pre-Logic Script

```
def fnInformation(Type, Classes, Cert, Associate, Bachelor, Graduate):
    if Type == "2 Year College":
        if (Classes=="Yes" and Cert=="Yes" and Associate=="Yes"):
            return "2 year college offering classes, certificates and degrees"
        elif (Classes=="Yes" and Cert=="Yes"):
            return "2 year college offering classes and certificates"
        elif (Classes=="Yes" and Associate=="Yes"):
            return "2 year college offering classes and associate degree"
        elif Classes=="Yes":
            return "2 year college offering only classes"
        else:
            return "2 year college no GIS"
    else:
        return "4 year institution"
```

Table 4: Code for Field Calculator

Points to Remember

- ☐ When a **define** statement is used a colon must be placed at the end of the statement and each line after the statement must be indented. The IDE will auto indent, but when typing in the script window, the user must manually create the indentation. If nested statements are used each level of indentation must have different spacing.
- ☐ The first **if** statement is establishing if the institution is a 2-year college. Providing it is true the logical path will be to the next **if** statement (note it is indented more than the first **if** statement), if it is false it will drop to the **else** statement at the end of the code with the same indentation level as the original **if** statement. The data set only has two types of institutions; two-year college and four year institution.
- When the first **if** statement is true it will drop to the next **if** statement which is using a logical '**and**' function. For the **if** statement to return a true result all members must be yes. If this is true then the 2-year college offering classes, certificates and degrees would be placed into the cell for the field Test. If not true, it would drop to the first **elif** statement and continue the process.
- ☐ Since two year colleges do not offer bachelor or graduate degrees they are not included in the script logic, but are part of the file.

It is important to note the indentation of the script, which was inputted manually by typing it directly into the code box.

Assignment 5.2

This assignment on the use of Python scripts in the Field Calculator is an extension of the example demonstrated in this technical skills lesson. In the example, only two-year colleges were classified, since the first **if** statement eliminated four-year institutions. For this assignment, the already demonstrated classification will be used, but in addition, there will be a classification of four-year institutions. Remember four-year colleges offer bachelor and graduate degrees.

- ☐ Use the college data set from before
- ☐ Create a flowchart of the code discussed in the lesson or create a flowchart of the code for the assignment, note which flowchart is provided.
- ☐ Create code for the following classification of four-year institution, keep the technical skills lesson code in the script (modifications of that code may be required depending on the methodology used).
 - Check to see if four- year colleges offer classes
 - Check to see if four-year colleges offer classes and bachelor degrees but no graduate degrees.
 - Check to see if four-year colleges offer classes, bachelor degrees and graduate programs.
- ☐ There are some four year colleges that also offer certificates and associate degrees, but those classifications are not required to receive the full credit for the assignment, for those seeking bonus points include all possible parameters for four-year colleges, this will require a greater number of decisions than what was shown in the lesson for two year colleges.
- ☐ Provide your instructor with the flowchart, script and a screen capture of part of the attribute table to show the correct writing of data.



Index



JCTC_ CIT 299_Module 2_Topic 5_Using_Python_Inside_ArcMap by Vincent A. DiNoto, Jr. is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).

This workforce product was funded by a grant awarded by the U.S. Department of Labor's Employment and Training Administration. The product was created by the grantee and does not necessarily reflect the official position of the U.S. Department of Labor. The U.S. Department of Labor makes no guarantees, warranties, or assurances of any kind, express or implied, with respect to such information, including any information on linked sites and including, but not limited to, accuracy of the information or its completeness, timeliness, usefulness, adequacy, continued availability, or ownership. This is an equal opportunity program. Assistive technologies are available upon request and include Voice/TTY (771 or 800-947-6644).