# Table of Contents

1. Intro
   1. Explain compiler thing
   2. Explain purpose behind the apples and the A B thing. Different way of thinking.
   3. Explain levels of languages
   4. Give VBasic Crap!
   5. Examples of uses ala Labeling in GIS
   6. Stick with it
2. Programming Basics
   1. Relate to database design.
      1. Design logically first
   2. GIS Programming
      1. In ESRI
         1. ArcObjects and API for Pro
         2. ArcGIS Engine
         3. ArcGIS for Developers
      2. Web Apps
      3. Customizing Desktop GIS
         1. In GUI
         2. Python tools
         3. VisualStudio
      4. Open Source (Python and QGIS/GRASS)
   3. Packages and arcpy

# Intro

## Compiling and Interpreting

A compiler is a program that converts instructions into a machine-code or some lower-level format so that they can be read and executed by a computer.

An Interpreter is a program that directly executes instructions without requiring them to be first compiled.

Python uses an interpreter. This means that you can get on command prompt (by pressing ‘win+r’, typing “cmd”, and pressing enter) and run python code simply by telling cmd that you want to use the python interpreter. If you have installed python, you should be able to do this by simply typing “python” and then your python statement. When you hit the enter key, cmd will use the python interpreter to read your line of code and execute it accordingly.

In some programming languages, this is not possible. You must first *compile* the program so that a computer can understand and execute it. In python, you simply have to tell the computer to use the python interpreter and then it can read and execute your python commands.

The python interpreter is normally installed as /usr/local/bin/python3.7 but this can differ if you installed it differently or are using a different version of python. Just know that because you have an interpreter, you do not have to compile anything and can run python virtually anywhere. From cmd, ArcGIS python command lines, etc.

## The Examples

Dr. Zhang gives two examples of programming logic. The first example has to do with how *Booleans* work in programming and the second has to do with the step-by-step process in which you must tackle complex problems.

Booleans are logical “If” statements. The important thing to get from this example is simply that programming, although using English characters, is not logically like our English language. It is more formulaic and standardized than our normal talking English. “If” means a very specific thing in programming, and to get a certain desired result (like 6 apples AND 1 watermelon), you must very specifically state this. Programming is kind of like giving Amelia Bedelia instructions. She will do EXACTLY or LITERALLY what you tell her to do. She doesn’t catch things you imply. Only what you clearly state.

The second example is meant to get you to further see the specificity you must have when writing code. You cannot simply say “do this complex thing”. You must tell the computer step by step how to do that complex thing. With functions, however, once you tell the computer once, you can call that complex operation a name and have the computer do that complex thing again without spelling it all out, but we will get to that later.

## Programming Language Levels

Different programming languages were developed for different specific uses. Some languages gain popularity and subsequently grow in supported uses and focus, but the intended original use usually tends to characterize how a language works and what it is capable of.

Because of this, not all programming languages are created equally. Some have a wide enough scope and uses that they can run independent of a specific computer system and hence are extremely powerful at many things. Some other programs may specialize in a specific use such as statistical data analysis or program assembly.

A high-level language is a programming language that enables a programmer to write programs that are more or less independent of a particular type of computer. High-level languages are closer to human language than machine language. These languages typically are easier to use and can automate significant areas of a computing system. High-level languages focus on usability rather than optimal program efficiency. This carries a lot of implications, but you don’t need to know that yet. Examples of these languages are C++, Python, PHP, Ruby, and others.

A low-level programming language is a language that differs very little from a particular computer’s instruction architecture. This means that commands in the programming language map very closely to processor instructions on a machine. These languages can convert to machine code without using a compiler or interpreter. Low-level languages run very quickly and with a very small memory footprint. These languages are simple but difficult to use because there are a lot of technical details that a programmer must remember when writing a code in a certain low-level language. Examples of low-level languages are any machine code and assemblers.

Intermediate languages tend to hover in between what has been described here as high and low. Examples of these are C, Visual Basic, Fortran, Cobol, etc.

Low and High level languages can be relative, but it is useful to know the limitations and uses for different languages before you start a project in one or the other.

## Visual Basic

This is the point at which I will launch into a completely biased (but warranted) rant about Visual Basic. It is going away. Microsoft is removing most of its high-level capabilities and focusing on making VBasic a core language only. This means that it will only be useful in very specific instances within Microsoft programs. In the past, VBasic and C# were developed in tandem. Now, Microsoft has officially stated that they will focus VBasic on core-scenarios only. For this reason, if you want a job that uses programming and you need a high-level language on top of python, focus on C# or Java. Don’t waste your time with Visual Basic right now. If you learn C#, Java, or Python, you can figure out VBasic if you need to later.

From Microsoft program manager Mads Torgersen:

*“C# is used by millions and will keep evolving as a "state-of-the-art programming language". F# is used by "tens of thousands" and will be "the best tooled functional language on the market". VB, though, is mostly used for Windows Forms applications (plus a few ASP.NET Web Forms), and by new developers, many of whom switch to C# when they discover its richer ecosystem.”*

## Examples of scripting usage

So why bother with coding and scripting when we have the ability to do so much by clicking our mouse?

That is a perfectly good question and there are lots of things that we don’t need to code out anymore. There are lots of things that we do need code for though. Automation, customization, scheduling tasks, making certain lengthy processes quicker, and even creating a whole program that fits your needs can all be done with programming. In GIS especially, this can be a huge advantage over your standard click-monkey GIS tech.

## Stick with it!

A

# Programming Basics

## Designing a project

A

## GIS Programming

#### In ESRI

a

##### Web Apps

a

#### Customizing Desktop GIS

A

#### Open Source

A

## Packages and arcpy

a