Lecture 14 - Midterm Review

This is really an overview of "how to be a successful programmer".

Outline

1. Be good at setting up a working environment
2. Learn how to learn new languages
    - How to edit
    - How to test in a new language
    - How to create a function/module/procedure/subroutine
    - How to branch
    - How to loop, iterate, or recursive operations
    - What are the data types
    - How to build more complex data
    - What are the string operations
    - Lists / Arrays / Maps
    - What are the "special features" that make the language good or bad
3. Build Something
    - What is the language good for
    - What is it "not" good for
4. Be good at learning new things
5. Be good at analyzing the "market" for what is happening

Claim to Fame

Object Oriented Programming

We are going to go on to looking at how Python builds "objects". This is "object" oriented programming. It was popularized by Sun Corp in Java and it is one of a number of systems that is used to help organize code. It is not the only system.

The top 15 most in demand programming languates: From: https://www.tiobe.com/tiobe-index/

1. C - no objects - just functions and data types - growing in use
2. Java - Only Object Oriented - falling and has been for years

3. Python - Object Oriented - has non-object code too - growing in use
4. C++ - Objects on top of C - Taught Next Semester
5. C# - Only Object Oriented
6. Visual Basic - Objects and non-Object code
7. JavaScript - new version have objects on top of "Prototypes"
8. PHP - both objects and non-object code - growing in use
9. R - non-object code - growing fast
10. SQL - non-object code
11. Perl - both object and non-object code
12. Groovy - both
13. Ruby - both
14. Go - interfaces
15. MatLab - non-object code

The 2 Most important principals in software development

1. Do the following steps
    - Make it work
    - Make it fast
    - Make it elegant
2. Wars are won not by the smarter but by the less stupid.
    - Less code is better than more code
    - More pure functions are better than site-effect functions/methods/objects
    - Things that improve testing are winners
    - Deign is only as good as the tests that are applied to it (No battle plan survives first contact with the enemy!)

How did Objects come about.

Before Objects World.

The Rise of "Java".

The Fall of "Java" and object oriented programming.