Object Oriented Programming

Before: Functions and Variables
Next: Objects

Very Important Stuff!!!

Modularity - Independent software modules because we can't do everything ourselves!

EVERYTHING IS AN OBJECT (In Python anyway)

Some languages have primitive types that are special (e.g. Java) or no objects at all (e.g. C)

Objects have attributes and methods

Attributes - Variables attached to an object
Methods - Functions attached to an object

We've been using them all along!

string.lower() is a string method
list.append(elt) is a list method

Class Definitions

Allow us to define a **Type** of object

e.g. String is a type of object. "abc" and "xyz" are different strings, but they are both strings (show comparison of values vs. types)

Bring up LetterList to demonstrate syntax

Init Method

Called every time you create a new instance

Instance - a particular object of some type: e.g. "abc" and "xyz" are each instances of String type

Show instance creation syntax in shell

Can take arguments (Modify on the spot to take an initial list as an argument)
Storing init argument as attribute allows us to "remember" it

(But where does the self argument go?)

Self

self is ALWAYS the first argument of a Python method.

self is used in the method definition to refer to “this particular instance” of a class
(show two different instances and how they call methods)

When calling a method **self** comes out of the parentheses and goes in front of the dot
Indicates “whose method” we are calling or “whose attribute” we are accessing

Attributes and Methods

(Repeat)
Attributes are variables attached to objects - represents things, properties, or tools an object has
Methods are functions attached to objects - represents things an object can do

Unlike local variables (which disappear when the function is done), **attributes** persist as long as the instance it is attached to still exists

In addition to its arguments, **methods** can use the attributes of the object it is attached to

Example: LetterList - explain methods and attributes, show them in use

Show how direct attribute access is dangerous

Getters and Setters - Types of methods that set or return an attribute, so that they don’t have to be directly accessed

Special Method Names

Some methods are called with special syntax
Names begin and end with __

Examples

__init__
__getitem__
__str__
__+__

Example - Inefficient Dictionary
Object Oriented Programming

This isn't just a tool, it's a paradigm!

Python is strictly Object Oriented, EVERYTHING is an object
Not true in all languages (e.g. in Java, primitive types are not objects, and are treated specially)

It's about modeling things as objects

An object should be an entity in your program, and encapsulate relevant information about it as attributes, and ways it can manipulate or be manipulated by other elements of your program as methods

e.g. In a video game:

Player has a location and a level that it is in. The level has a list of players in it.
Player may have methods like "move" and "attack". Level might have a "reset" method

Example (with remaining time)

Define a Car class
(Write code if a lot of time, talk conceptually if not)

What sorts of attributes might it have?

Methods?