Announcements

- HW due Friday (part 1)
  next part due 1 week later

- Midterm 2 will be coming just
  before or after Easter
Some notes on the tW

Do not sort!

'Zoo' is before 'apple'
'in'
'In'

- Each word ends in newline
  'in' ≠ 'in

- Don’t forget a unit test
  + comments / docstrings.
Some User Interface Notes:

Goals:
- Easy to use
- Efficient - minimal input

Last HW: If input isn't of correct type, what should the program do?
- reprompt in a loop
Class guidelines

Classes should be self-contained

* all values for class are inside the class
* generally variables are treated as private

\[
\text{mypoint} = \text{Point}(1) \quad \text{BAD!}
\]
\[
\text{mypoint}.x = 5 \quad \text{BAD!}
\]
\[
\Rightarrow \text{mypoint}.setX(5) \leftarrow \text{BAD!}
\]
Class Methods

- perform a specific function

- input either gotten in the function: `getInterest()`
or provided as input parameter: `setX(5)`

- If input is wrong type, what to do?

depends:
- `getInterest` - re-prompt
- `setX` - throw an error
Testing a class

Unit tests treat the class like a "black box": the internals are hidden!

So you just call the functions which are part of the class, and check if they behaved correctly.
New chapter: Inheritance

Goal: Minimize duplication of code and effort.

So never cut and paste!

We've used:

- loop
- constants
- function
What about classes?

Clearly, sometimes classes have things in common.

Ex: person class:
   - Name
   - Address
   - Age

student class:
   - Name
   - Address
   - Age
   - Major
   - Classes
Inheritance

We say a child class inherits the data and methods of its parent class.

Generally, the child class will either:

- **augment**: add new data or methods
- **specialize (and override)**: rewrite some methods
Simple example: 3-D points

Remember our point class?

Methods:

- setX
- setY
- getX
- getY
- __mul__
- __add__

- constructor
- scale (useful, but need to augment)
- distance

Data:

- self.x
- self.y

Need to be rewritten
3-D points

Need x, y, and z.

Methods:

  add    set z
  get z

  rewrite  --add--
           --mul--
Syntax

```
from myPoint import Point

class ThreeDPoint(Point):

    def __init__(self, x, y, z):
        # call parent constructor
        Point.__init__(self, x, y)
        self.z = z
```

How to construct our point?
Other functions

- Steal what we can, python will call parent class' function for us if we don't write a new version.

Our job:

- add new functions
- override old functions that are now incorrect