CSISO - Multiple Inheritance & Deeper Understanding of Objects

Announcements

HW due Friday
Quiz recap

class Student (Person):

  #2 functions

  - called parent versions (with appropriate parameters)
Last time

Inheritance with cs1 graphics

Examples:
- star (from Polygon)
- Car (from Layer)
- car (from Drawable) - see p. 316
Multiple Inheritance

Only supported by some languages.
Allows you to use code from multiple classes.

Ex: Labeled Rectangle in o1graphics

This object is both Text & Rectangle

Inherits: from Text: set Message, set Font Color...
          from Rectangle: set Width, set Fill Color...

          (from both: move, scale, etc.)
Child: Deeper understanding of objects

Example of a simple, mutable object: Account class

Suppose we say: mySavings = Account()

Picture of memory:
- mySavings is really a reference to the actual Account.
- We view it as a pointer to the object.
- Each call to the constructor creates a new Account object.

Like: myChecking = Account()

(Try the id command - the 2 labels will be distinct)
References to the same object

Now try:

```
myDebit = myChecking
id(myDebit)
```

So my Debit is an alias, & changing it will change both:

```
myDebit.deposit(100.0)
myChecking.getBalance()
```
Equivalence Testing

- Can use is:
  - my Savings is my Checking
  - my Checking is my Debit
  - my Checking is my Checking

How does it work?

checks id
Operator :=

In contrast, `operator=` is a broader notion of equivalence.

```python
mySavings.deposit(100)
mySavings := myChecking
(Remember, we coded this one.)
```