CSC207.01 2014S, Class 09: Classes and Objects, Revisited

Overview

- Preliminaries.
  - Admin.
  - Questions.
- Writing fields.
- Writing constructors.
- Writing methods
- Standard methods.
- Lab!

Preliminaries

Admin

- Apologies if I'm discombobulated - I had code problems in 151 and 207 today.
- You should have received current grades from me. Let me know if you have questions.
- Extra credit:
  - Convo Feb. 5 at noon in JRC 101: Pussy Riot and Putin.
  - Wednesday Extra, February 5 at 4:30 in 3821, AppDev
  - CS Table Friday: NP Completeness.
  - More?
- New lab partners
- Mentor session tonight at 7:00 in 3821.
- Lots of people absent. They should be sending me email to explain why.

Upcoming Work

- Reading: Debugging
- Homework 3, Due Wednesday
- Today's writeup: Exercises 2 and 4
  - Due Wednesday.
  - Subject: CSC 207 Writeup 7: Classes (YOUR NAME(S))

Questions on the homework

Writing classes

- A class is a template for making objects
- It includes methods
- It includes fields - the data
- It includes constructors - how to build new objects
- Each of these have a protection level
  - public - anyone
  - protected - class, package, subclass
  - nothing (package) - class and package
  - private - only this class

- In Java, we also have

  - static fields - shared by all objects
  - static methods - can be used without objects
  - constants
  -

Writing fields

- A lot like variable declarations

  TYPE NAME;

- Convention: name starts with lowercase letter and uses camelCase

- Can have a protection level. - Usually package or private.
- Becomes usable in *every* object in the class - individual
- Tend to refer to them internally as
  - `this.fieldName` (in the current object)
  - `other.fieldName` (in another object)

- You can drop the `this.` and it is assumed. But it's ambiguous

  public class Amby { int x;

  public int multiplyBy(int x) { return this.x*x; } } // class Amby

Writing methods

- A lot like C

  RETURNTYPE NAME(PARAMS-WITH-TYPES) { CODE }

- Optional protection level
  - Sam's custom: Separate public and non-public methods in your class
- camelCase again
- Can refer to fields

Writing constructors

- Look a whole lot like methods
- Name *must* match the name of the class.
- No return type

- No return call

  ClassName(PARAMS) { } // ClassName(Params)

Standard methods

- toString()
- equals()
- hashCode() - a number that represents the object
- compareTo()

Lab

We'll talk about exceptions on Wednesday.

Why doesn't Sam like

```
public Fraction multiply(Fraction multiplicand)
{
  int productNum = this.num * multiplicand.num;
  int productDenom = this.denom * multiplicand.denom;
  ...
} // multiply
```

Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.