CSC207.01 2013F, Class 14: Inheritance, Continued

Overview

- Preliminaries
  - Admin.
  - Questions on HW4.
- Q&A.
- Lab.

Admin

- Don't forget study break tonight at 8pm!
- Would you prefer that the current eboard link link to the source rather than the HTML?
- Sorry about the delays in grading. Class and assignment prep takes priority. It also doesn't help that I'm booked every weekend until fall break.
- Today's strange conversation with my youngest son: "Dad, why do people go to hack-a-thons?" "Lots of reasons ... and to build their programming chops." "Hack. Chop. Funny dad."
  - On that note, we'll get a quick report on MHack.
- Reading for Tuesday: Documentation with Javadoc
- Android starts Wednesday. I think I have one tablet for every two students. I'll let you negotiate partners and who, if anyone, gets to hold on to the tablet.
- EC Opportunities
  - Learning from Alumni: Jim Finnessy
  - CS Extras Thursday @ 4:30: Jennelle Nystrom on Microsoft
  - CS Table Friday (The Story of Mel, A Real Programmer)
  - G-Tones, Saturday, 3:30 at Herrick Chapel
  - Baseball Scrimmage Sunday
  - Water Polo Meet. MITC.
- I'm trying to do very little lecture today. We'll see if I succeed.
- HW4 is due Tuesday at 10:30 p.m.
  - Are folks okay if I generally move the due time for homework from Monday to Tuesday?

Questions on HW4

Can I store the numerator and denominator as BigInteger values?

Certainly.

Why doesn't my `equals` method work?

```
public boolean equals(Fraction other) {
    return this.numerator.equals(other.numerator) &&
             this.denominator.equals(other.denominator);
} // equals(Fraction)
```

If you write an `equals` method with a signature of `equals(Fraction other)`, the Java often doesen’t know to call it. In general, it thinks it should call `equals(Object other)`. So, you have to cast.

```
public boolean equals(Object other) {
    return this.equals((Fraction) other);
} // equals(Object)
```

or

```
public boolean equals(Object other) {
    return (other instanceof Fraction) &&
            (this.equals((Fraction) other));
} // equals(Object)
```

Why does the constructor for fractions fail?

I think I wrote something like `if (this.denominator.equals(...))` without first asssigning to `this.denominator`. (Only some of you seem to have had this problem.)

We compare two fractions by subtracting one from the other and then return the numerator as converted to an integer. What do you think about this awesome approach?

Okay in theory, but ... overflow issues. Try using .signum() or use BigInteger.compareTo.

Can we assume that our procedures work?

No. But simple tests can test multiple procedures (e.g., constructor and toString().

What should `evaluate` return when given an assignment?

The value assigned.

Is this like C in which you can have beautiful stuff like `r0 = r2 = r0 + 1`?

No, you do not have to support that.

Q&A on Other Issues

How would you have solved the "separate strings" problem from HW3?

```
// Count how many times the separator appears
    // Repeatedly use indexOf
// Build the array of the correct size
// Step through the string (using indexOf), with substring
```

Lab

Continue Friday’s lab.