CSCI 1300
Intro to Computing

Gabe Johnson

Lecture 22 March 6, 2013

Java Intro 3
Cheat Sheet

Lecture Goals

1. Tests are back
2. Slug Race
3. Java Cheat Sheet
4. Codebat
5. Java Graphics

Upcoming Homework Assignment

HW #5 **Due: Friday, Mar 8**

Slug Race

We'll talk about the Slug Race in class today. Short version: you need to edit the Slug class (in Slug.java or in slug.py) and flesh out the *next_round* method.

I intentionally made the Java and Python versions pretty easy to compare and contrast. If you can do the Python version, you should be able to tweak it to get your Java version to work.

Test Results Page 2

The solution to exam 2 is on GitHub. Check it out, and if you think you can make a case for getting points back, **come up to us after class**.

Be aware that page 2 was graded incorrectly. It is perfectly OK to write ‘foo’ to get a string. Single quotes work. If we took points off for a single-quoted string, we’ll give you those points back.

Slug Race

The Java version is very similar to the Python version.

The *next_round* function is part of your slug. There is a *Race* class that I wrote that will call your slug's *next_round* function repeatedly. This is the only function you need to edit.

The goal is to keep your slug rested and fed while also moving towards the finish line.

How To (Build/) Run It

In Python:

```
$ python race.py
```

In Java:

```
$ javac *.java
$ java Race
```

Many Ways To Do It

It isn't hard to get your slug across the finish line. But there's lots of ways to do it. You could:

1. Repeat the same sequence over and over.
2. Eat when hungry, rest when sleepy, move otherwise.
3. Write a simulation that optimizes your slug. (advanced)
4. Cheat.

Yes, *Cheat*. You can trick the Python version into winning in one round. I don't think it is possible with the Java version, however.

Java Syntax Cheat Sheet

I am uploading a Syntax cheat sheet to GitHub. You can use it to write your Slug Race code.

The biggest difference between the Python and Java versions of this assignment is that the Java version has *protected variables*.

Protected Variables

In Python you can access any variable you like. But Java was designed to be used in large software engineering projects, where people like to have more control over what is and is not visible. This is both for security's sake and for basic human sanity. Engineers are overwhelmed when there is too much going on, both from a documentation perspective, but also from a debugging perspective; so we intentionally try to keep some things hidden.

Protected Variables 2

```
class BaseSlug {
   private double hunger = 5.0;
   ...
   public final getHunger() {
      return hunger;
   }
}
```

Since *hunger* is private, this means the only place we can read or write that variable is in the BaseSlug class. Since Slug is a *subclass* of BaseSlug, it can't access it directly. So we create an *accessor* method *getHunger()* to give read-only access to that variable.

Using Accessors

```
class Slug extends BaseSlug {

  public Decision next_round() {
      if (getHunger() < 1.4) {
        return Decision.EAT; // HUNGRY!
      } else {
        // NOT HUNGRY! DO SOMETHING ELSE!
      }
  }
}
```

To use an accessor method (or any other method, for that matter), just *call* it. This looks just like it does in Python, except we don't need to use *self*.

CodingBat.com

One of the students pointed this out to me. It seems excellent.

http://codingbat.com/

"Talent is a rumor, but practice works."

You can use this to practice coding beyond the homeworks. It gives you instant gratification by telling you what works and what doesn't.

It is maintained by Nick Parlante at Stanford.

Graphics are Up Next

Starting next time we'll start doing some graphics programming. Check out the 'bouncing balls' app on github in...

cs1300 / code / java / bouncy.

See if you can get it to run.