Compsci 201
Java, Classes, Objects

Susan Rodger
January 15, 2020
Announcements

• Assignment P0 due Thursday, January 16
  • Really means January 17
  • No late penalty since first assignment til Jan 24
• APT-1 due Tuesday, January 21
• No Discussion on Monday, January 20!
• Reading on calendar
  • try to read by date posted, ok if you need a few more days
How can you access code?

https://coursework.cs.duke.edu/201spring20/classcode

• Browse code, use code
  • Change code? Can you?
  • Yes if …
  • To push? Must fork

• Use your eyes and experience …
  • These are enough
Reminder: Course Information


• Sakai

Course Info
B is for …

• Bug
  • What you will always have and need to fix

• Bytecode
  • Java on the “bare metal”
PFtD and PFtW

• How to succeed in finishing and submitting APTs, and Assignment P0
  • Java Knowledge
  • Git, Submission knowledge

• Concepts in P0: Classes, Objects, Arrays
• Concepts in APTs: Accessing and Structuring Data
  • Readings will help fill in more details
  • Think about what you’re reading and programming
Defying Gravity: WOTO-Style

• How to do an APT
  • https://www2.cs.duke.edu/csed/newapt/gravity.html
  • From text editor to IntelliJ

https://youtu.be/Yf9Bt5WFZKs?t=111
Anatomy of falling due to Gravity

• The class Gravity has one method, falling
  • The method falling has two parameters
  • The method falling returns a double value

```java
public class Gravity {
    public double falling(double time, double velo) {
        // fill in code here
        return 1.0;
    }
}
```
Think Before You Code

• Solve by hand … Check your understanding of examples … think about the solution you’ll write

• Then think before fingers on keys
Problem Solving to Code
7 Step Process

1. Work small examples by hand
2. Write down what you did in words (algorithm)
3. Find Patterns (generalize algorithm)
4. Work another example by hand (does your algorithm work? If not, go back to 2)
5. Translate to code
6. Test several cases
7. Debug failed test cases
Learning a new language

• You'll translate what you know into the new language when you first start
  • You’ll view Java, the new language, in terms of the language you know
  • Eventually you’ll program idiomatically and colloquially!

• Arithmetic with integers and doubles as an initial and common vernacular?
Coding Interlude

- Working on Gravity APT in IntelliJ, but jShell first
Completing the Gravity APT

• Is getting all-green a requirement?
  • Can you do well without getting all green?
  • [http://thegreendance.com/](http://thegreendance.com/)

[https://www.youtube.com/watch?v=1QmvBBMBurA](https://www.youtube.com/watch?v=1QmvBBMBurA)
WOTO (4 minutes)

What Would FB Say?
The most important single decision I ever made was to change the IBM 360 series from a 6-bit byte to an 8-bit byte, thereby enabling the use of lowercase letters. That change propagated everywhere.
Emails used to look like this…

SUSAN,

WOULD YOU LIKE TO MEET ME FOR LUNCH TOMORROW?

THOMAS
Emails used to look like this…

THOMAS,

NOT IF YOU ARE GOING TO YELL AT ME!

SUSAN
Why is Programming Fun?

• First is the sheer joy of making things
• Second is the pleasure of making things that are useful
• Third is the fascination of fashioning complex puzzle-like objects of interlocking moving parts
• Fourth is the joy of always learning
• Finally, there is the delight of working in such a tractable medium. The programmer, like the poet, works only slightly removed from pure thought-stuff.
Java Primitives and Objects

• Java has two different kinds of constructs
  • Primitive: int, double, char, boolean, …
  • Objects: pointers/references to (bigger) things

• We'll look first at primitive types: numeric to start
  • Move to boolean and char later
  • Move to Strings and other objects later
Primitive Values and Variables

• Changing value stored in memory for x has no effect on value stored in memory for y

```java
int x = 5;
int y = x;
x += 7;
// values of x and y are?
```

• Operations on numeric types:
  • +, -, /, * for int and double. *Careful on /*
  • % for int
Primitive Values and Variables

• Changing value stored in memory for x has no effect on value stored in memory for y

```java
int x = 5;
int y = x;
x += 7;
// values of x and y are?
```

x is 12, y is 5

• Operations on numeric types:
  • +, -, /, * for int and double. Careful on /
  • % for int
Object Values and Variables

• Object variables are labels that reference/point to storage. (labels that are put on boxes)

```java
String s = new String("Duke");
String t = new String("Duke");
// are if statements true or false?
if (s == t)
    // they label the same box
if (s.equals(t))
    // contents of boxes the same
```

What's in the boxes?

"Duke" is in the boxes
Object Values and Variables

- Object variables are labels that *reference/point to* storage. (labels that are put on boxes)

```java
String s = new String("Duke");
String t = new String("Duke");
// are if statements true or false?
if (s == t)
    // they label the same box
   FALSE
if (s.equals(t))
    // contents of boxes the same
   TRUE
```

What's in the boxes?

"Duke" is in the boxes
Object Values and Variables

- Object variables are pointers, value is a memory location or address. Copying value is copying address.

```java
String s = new String("Duke");
String t = s;
// are if statements true or false?
if (s == t)
if (s.equals(t))
```

![Diagram showing memory boxes labeled s and t]
Object Values and Variables

- Object variables are pointers, value is a memory location or address. Copying value is copying address.

```java
String s = new String("Duke");
String t = s;
// are if statements true or false?
if (s == t)
    // they label the same box
    TRUE
if (s.equals(t))
    // contents of boxes the same
    TRUE
```
Pictures worth $2^{10}$ words?

Arrays

- Java – array
- Python – lists
- Matlab – arrays

- Indexed collections
- Index – reference to an item
Java Array Basics

- Constructing and initializing ...
  - `int[] a = new int[100];`
  - `int[] b = {1,2,3,4,5};`

- Indexing starts at 0, exceptions on bad indexing
  - `a[5] = 7;`
  - `b[5] = 7;`
Java Array Basics

• Constructing and initializing ...
  • `int[] a = new int[100];`
  • `int[] b = {1,2,3,4,5};`

• Indexing starts at 0, exceptions on bad indexing
  • `a[5] = 7;` ✓
  • `b[5] = 7;` ❌
Coding Interlude

• Let's look at Strings and Arrays