How Data Works

APTs will be due at Midnight!

Office hours after class!

Computer Science 201
Things you need to store

Circles

People

Weather

Scientific Data

(...presentation slides...)

Monday, September 3, 12
Primitives

boolean  T/F
char     ‘a’ or ‘7’ or ‘$’ or ‘D’ or...
byte     [-128, 127]
short    ± 33,000
int      ± 2 billion
long     ± 9 quintillion
float    ≈ 7 sig figs
double   ≈ 16 sig. figs

and that’s it!
<table>
<thead>
<tr>
<th>Primitives</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>T/F</td>
</tr>
<tr>
<td>char</td>
<td>‘a’ or ‘7’ or ‘$’ or ‘D’ or...</td>
</tr>
<tr>
<td>byte</td>
<td>[-128, 127]</td>
</tr>
<tr>
<td>short</td>
<td>± 33,000</td>
</tr>
<tr>
<td>int</td>
<td>± 2 billion</td>
</tr>
<tr>
<td>long</td>
<td>± 9 quintillion</td>
</tr>
<tr>
<td>float</td>
<td>≈ 7 sig figs</td>
</tr>
<tr>
<td>double</td>
<td>≈ 16 sig. figs</td>
</tr>
</tbody>
</table>

When might you need a long?
Properties of Primitives

They start with a lower-case letter.

They have *literals* (for creating).

```java
int x = 5;
double foo = 6.2;
```

They have *operators* (for verbing).

```java
double bar = x + foo;
bar *= 2;
double z = bar / 6;
// ++, --, ==, !=, etc.
```

In short: *special syntax!*
The loneliest number
The loneliest number

```java
int x = 5;
int[][] y = new int[10][12]; // starts out all zeros
y[0][0] = 5;
y[1][3] = 6;
...
double[] z = new double[1024]; // all 0.0
for (int i = 0 ; i < z.length ; ++i) {
    z[i] = Math.sqrt(i * 100.0);
}
...

int[][] foo = new int[5][10];
int[][][] bar = new int[5][10][15];
...
```
But what about...

What can we store about a person? Does it depend on what you’re doing?

(Thanks, Google+ people who didn’t know you were going to be on a slide!)
What can a Person do?

Does it depend on what you’re doing?

• Facebook / G+?
• Duke student database?
• IRS database?
• ...?
public class Circle {
    int xLocation;
    int yLocation;
    Color color;
    int radius;

    Circle(int x, int y, Color c, int r) {
        xLocation = x;
        yLocation = y;
        color = c;
        radius = r;
    }

    public int getXLocation() {
        return xLocation;
    }

    // some code omitted
}

// this brace closes the class
public class Circle {
    int xLocation;
    int yLocation;
    Color color;
    int radius;

    Circle(int x, int y, Color c, int r) {
        xLocation = x;
        yLocation = y;
        color = c;
        radius = r;
    }

    public int getXLocation() {
        return xLocation;
    }

    // some code omitted
}

// this brace closes the class

(You used this class in DrawCircles!)
public class Circle {
    int xLocation;
    int yLocation;
    Color color;
    int radius;

    Circle(int x, int y, Color c, int r) {
        xLocation = x;
        yLocation = y;
        color = c;
        radius = r;
    }

    // much code omitted
}

// this brace closes the class
History & Demo Time!

http://www.cs.utexas.edu/~aim/
Java Data Types

**Primitives**

Literals:
- `int x = 5;`
- `double y = 10.2;`
- `char z = '5';`

Operators:
- `int x = 5+10;`
- `double y = 3.0*6;`
- `double z = x/y;`

**Objects**

Constructors:
- `Foo a = new Foo();`
- `Person p = new Person("Mac", 75, "jmm71");`

Methods:
- `a.doSomethingUseful();`
- `p.teachALecture("Monday");`

We get to make our own types!
Object Cheat Sheet

Start with a capital letter (e.g. CirclesCountry)
Created with a constructor (using new)
Have member variables (which store data)
Have methods (which operate on data)
Be nervous about operators on objects.  

(More on Wednesday)
Object Cheat Sheet

Start with a capital letter (e.g. `CirclesCountry`)
Created with a constructor (using `new`)
Have member variables (which store data)
Have methods (which operate on data)
Be nervous about operators on objects.

---

**Snarf Sep3InClass**

You should be able to
identify the member data, constructor, and methods.

Add the data and methods necessary to store friends.

(Pseudocode ok; real code better. Be ready to tell us what you’ve done)
Object Recipe

0. “Hey, I have some data that are too complicated for a primitive (or array of primitives).”

1. Decide what data you need. (nouns)
2. Decide what methods you need (verbs)
3. Make a new class:
4. Add your data
   *Inside* the class; *outside* the methods
5. Write your constructor. (which fills in your data)
6. Write the rest.

*When in doubt: examine DrawCircles!*
Hangman!