Announcements

- Review sessions
  - Have classrooms for the reviews:
    - Thursday 6-8pm, 207 MATH
      - Basics
    - Friday 12-2pm, 717 Hamilton
      - Basics
    - Friday 2-4pm, 717 Hamilton
      - Intermediate
    - Sunday 12:30-2:30, 833 MUDD
      - Intermediate
    - Sunday 3-5pm, 833 MUDD
      - Intermediate
Review Basics

- I am going to start a review of Java

- Want to schedule more sessions (Tas)
  - Want to know what works
  - Will be hands on based + review

- By the end of the week you should be familiar enough with basic java programming to be able to work on the homework

- Need to practice if this is really new to you, and please stop by to talk
Java Refresher Course II

3134 – DS in Java
Spring 2007
Basics II
• Last time we reviewed the basics

• Here are some questions from the feedback papers
Creating variables

- String s1;    //single
- String s1,s2,s3;  //multiple
- String s4 = new String();
  //same as
- String s4;
  s4 = new String();
Special String class

- String is a special class
- Anything between quotes will be turned into a string for you
- String s = "hello";
  is turned into
  String s = new String("hello");
Strings

- Sequence of Unicode characters
  - (Technically, code units in UTF-16 encoding)
- length method yields number of characters
- "" is the empty string of length 0, different from null
- Special class in Java
  - Assigning a string literal to a string reference creates an instance!
- charAt method yields characters:
  
```java
char c = s.charAt(i);
```
String II

- substring method yields substrings:
- "Hello".substring(1, 3) is "el"
- Use equals to compare strings
  if (greeting.equals("Hello"))
- == only tests whether the object references are identical:
  if ("Hello".substring(1, 3) == "el") ... // NO!
String concatenation

- + operator concatenates strings:
  - "Hello, " + name
- If one argument of + is a string, the other is converted into a string:
  int n = 7;
  String greeting = "Hello, " + n;
  // yields "Hello, 7"

- toString method is applied to objects

Date now = new Date();
String greeting = "Hello, " + now;
  // concatenates now.toString()
  // yields "Hello, Wed Jan 17 16:57:18 PST 2001"
Converting Strings to Numbers

- Use static methods
  - WHY???
  - `Integer.parseInt`
  - `Double.parseDouble`

- Example:
  ```java
  String input = "7";
  int n = Integer.parseInt(input);
  // yields integer 7
  ```

- NOTE:
  - If string doesn't contain a number, throws a `NumberFormatException`(unchecked)
We want to represent a person

- What components would be important??
Person, how would you represent each one??

- firstName
- lastName
- homeAddress
- phoneNumber
- email
- etc
Address

- What exactly is an address ??
Address

- Street
- City
- State
- Country
- Zip
public class Address {
    private String stAddress, city, state, country;
    private String zipcode;
}

What is missing ??
Lets add

- Constructor
  - What arguments will we be taking?
Next

- What accessor methods do we need?
Public Address(String street, String town, 
String st, String cnty, String zip) 
{

stAddress = street;

\textcolor{red}{city = town;}
state = st;
country = cnty;
zipcode = zip;

\textcolor{red}{}
}
Using the address class

- Address ad1 = new Address("555", "main st", "Brooklyn", "NY", "usa", "111111");

- What happens when you do:
  System.out.println(ad1);
Every class in java is also a java Object class

Which means no matter what class will have the following methods available:

- `toString()`
- `getClass()`
- `equals()`
- `clone()`
default

- **toString()**
  - `nameofclass@10b62c9`

- **getClass()**
  - Special static id of the class

- **equals()**
  - Default will only tell you if it’s the same copy

- **clone()**
  - Will throw an exception by default
So...

- How would you fix the `toString` of our `address` class??
public String toString() {
    return stAddress + "\n" + city + " ," + state + " " + country + " " + zipcode;
}

Back to person

- What does the person class now look like?

?
- Ok, let's write some more code

- Create a few people and print them out in main
Working with code

- **Compile time errors**
  - Use API to look up stuff
  - Use Eclipse to help you solve the problem

- **Run time errors**
  - Sometimes easy to spot since you know what line
  - What if line involves a bunch of variables, and you know one of them is causing the error, but how to figure out which one?
Solution 1

- Print out a bunch of stuff

- Problem:
  - Might be complicated to find out which
  - Mess up neat code
  - Might not be printed out exactly
debugging

- Idea of debugger

- Example of running program