Topics: Course overview, Java intro

1 Welcome

• Welcome to 61B
• Summer is short and intense (but fun)
• Attend lab and discussion!
• Read the webpage and newsgroup
• Sections 104 and 105 have been cancelled
• First homework assignment is up

2 Course Overview

• Get a copy of Course Overview handout (also on webpage)
• For scheduling concerns, see Michael-David in 379 Soda
• Get the readers from Copy Central on Hearst Ave.
• Schaum’s outline is recommended, Goodrich & Tamassia is required

3 Coursework

• One homework per week
• Two labs per week
• Three projects in the semester
• Homeworks and projects are substantial - start them immediately

4 Exams

• Two midterms and a final
• Midterms scheduled for 6:40PM, July 9 and July 30
• Final scheduled for 6:40 PM, August 15
• Exams are cumulative, open book, open notes, closed all non-paper items
• Contact me at least one week in advance if you have a scheduling conflict
5  Grading
   • Gradescale on handout
   • No curve
   • It is likely that scores on homework and projects will be higher than on exams
   • Grading based on a old 200 point scale, but with more than 200 points available

6  Feedback
   • We are very open to suggestions
   • In a week, we’ll be asking for feedback through an anonymous form on the webpage
   • In the next lecture, we’ll try a new peer-instruction technique you may have heard of

7  Java - simple example
   • A simple program that prints “Hello, World!”
   • public denotes access
   • void denotes the type of the return value
   • main is a method

   public class Hello {
      public static void main(String[] args) {
         System.out.println("Hello, World!");
      }
   }

8  Java - simple example
   • After placing the code in Hello.java, execute:

      # compile, enable debugging
      \$ javac -g Hello.java

      # java Hello
      Hello, World!

9  Hello.java dissected
   • All definitions are within a class
   • A main method makes the class executable
   • main has no return value (a void return type) and takes String[] (an array of Strings) as an argument
   • The . (period) accesses things within one another (System.out.println = the println method of the object out inside of the class System)
10 Hello.java dissected

- All statements end with a semicolon
- Blocks of code are enclosed with {} 
- String values can be represented by characters in double quotes

11 A limerick generator

Problem: A program that generates (bad) limericks

- A limerick has the structure AABBA

Plan: Ask user for input

- User can provide rhyming words, we string them together

import java.io.*; /* uses classes from java.io */

public class Limerick {
    public static void main(String[] args) {
        try { // executes the block, handles an exception
            BufferedReader inp = new BufferedReader(new InputStreamReader(System.in)); // set up the reader
            while (true) {
                System.out.print("[1] Noun: "); // print to the screen, without an end-of-line
                String noun = inp.readLine(); // read from the user
                System.out.print("[2] Place: ");
                String place = inp.readLine();
                System.out.print("[3] A thing that [1] has done + rhymes with [2]: ");
                String does = inp.readLine();
                System.out.print("[4] A thing that [1] has: ");
                String has = inp.readLine();
                System.out.print("[5] A color + rhymes with [4]: ");
                String color = inp.readLine();
                System.out.print("[6] A word + rhymes with [2]: ");
                String word = inp.readLine();
                System.out.println("There was a " + noun + " from " + place);
                System.out.println("Who went in for a " + does);
                System.out.println("got a horrible " + has);
                System.out.println("went terribly " + color);
                System.out.println("and never again went " + word);
            }
        } catch (IOException e) {
            System.out.println("Caught an exception!");
        }
    }
}
12 Limerick.java dissected

- `import` inclusion of Java standard library packages for easy use
- `/* */` denotes a comment (multi-line)
- `//` denotes a comment (until end of line)
- `try { } catch (Exception e) { }` allows handling of exceptions where `Exception` is the type of exception thrown
- `while(condition)` allows looping
- Every variable has a declared type - `inp` is an object of type `BufferedReader`
- The keyword `new` is used to instantiate (create) new objects of a certain type
- When `new` is used, a constructor is called - the constructor can take arguments
- Assignment to a variable is achieved with the `=` operator
- `+` can concatenate Strings

13 Definitions

- Object - A container for data
- Class - A type of object
- Variable - A name for a reference to an object
- Method - A function that operates on an object or class

14 Types

- Either primitive or reference (classes)
- Primitive types include boolean, int, char, double (there are more)
- All variable must have a declared type
- Defining a class creates a new type (which means objects of that type can be created)
- By defining new classes to hold information in interesting ways, we create data structures