CS100J. Lecture 2, 25 January 2007

Today's topic: **Objects and classes**

Reading for this lecture: Section 1.3. It's most important that you **study this section over the weekend** and **practice** what is taught using DrJava.

PLive: Activities 3-3.1, 3-3.2, 3-3.4 (not 3-3.3), 3-4.1, 3-4.2.

Summary of lectures: On course page, click on "Handouts" and then "Outline of lectures held so far".

Quiz on Tuesday. Tell me (1) what a type is, (2) how to execute an assignment statement, (3) how to draw a manilla folder (instance, object) of a class. See course web page for more details.

Quote for the day:
Computers in the future may weigh no more than 1.5 tons.
--Popular Mechanics, forecasting the relentless march of science, 1949

1

About CMS

CMS: **C**ourse **M**anagement **S**ystem.Uses to maintain grades, handle submitted assignments, post grades, handle regrades, etc. Developed by the CS Department.

Look at this URL: http://cms3.csuglab.cornell.edu

Click on "Secure extended login", if you see it. You will be asked for your cornell netid and password. After you have entered it, you will either be in the CMS and will see the course description or you will see on the right something like this:

CMS Overview
My Courses
Com S 100J (student)

If it lists CS100J, click on it, and you are in the CMS. If it doesn't, you are not in the CMS; email Stacey Shirk at shirk@cs.cornell.edu and ask her to register you in the CMS for CS100J.

2

Two aspects of a programming language

- Organization – structure
- Procedural —commands to do something

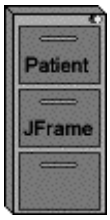
Example: Recipe book

- **Organization: Several options; here is one:**
  - **Appetizers**
    - **list of recipes**
  - **Beverages**
    - **list of recipes**
  - **Soups**
    - **list of recipes**
  - **…**
- **Procedural: A recipe is a sequence of instructions to carry out**

We begin by studying organization — structure

3

A class is a file-drawer. Contents: manila folders, each containing the same kind of information



Bill

class name

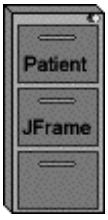
Patient

name	B. Clinton
address	New York
owes	$250.00

manila folder: an **object** or **instance** of the class

4

A class is a file-drawer. Contents: manila folders, each containing the same kind of information



Bill

Patient

name	B. Clinton
address	New York
owes	$250.00

name, address, owes: **variables**, called **fields** of the folder

5

A class is a file-drawer. Contents: manila folders, each containing the same kind of information

Bill

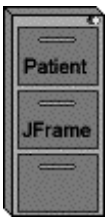
Patient

name	B. Clinton
address	New York
owes	$250.00

Name on tab (Bill): can be anything you want, as long as it is unique

6

A class is a file-drawer. Contents: manila folders, each containing the same kind of information



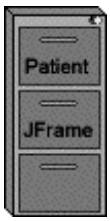
Bill

name	B. Clinton
address	New York
owes	$250.00

Patient

Instructions to be carried out by different people:
change the name, get the name, bill the patient, receive money from patient, insert teeth xrays into the folder, …

A class is a file-drawer. Contents: manila folders, each containing the same kind of information



Bill

name	B. Clinton
address	New York
owes	$250.00

getName()
deposit(double d)

Patient

Instructions to be carried out by different people: methods.
getName is a **function**; it returns a value.
deposit is a **procedure**; it does some task, doesn't return value

pat Bill

variable contains the name of the folder

Bill

name	B. Clinton
address	New York
owes	$250.00

getName()
deposit(double d)

Patient

pat.getName() function call. Its value is "B. Clinton"

pat.deposit(250.0); procedure call. Change the value of field owes to 0.

pat Bill

variable contains the name of the folder

Bill

name	B. Clinton
address	New York
owes	$250.00

getName()
deposit(double d)

Patient

new Patient() An expression that creates a new folder, puts it in file-drawer Patient, and gives as its value the name of the folder.

pat= **new** Patient(); A statement that creates a new folder, puts it in file-drawer Patient, and stores the name of the folder in variable pat.

package: A collection of classes that are placed in the same directory on your hard drive. Think of it as a room that contains file cabinets with one drawer for each class.

package **java.io** classes having to do with input/output

package **java.net** classes having to do with the internet

package **java.awt** classes having to do with making GUIs

package **javax.swing** newer classes having to do with GUIs

To reference class JFrame in package javax.swing, use:

javax.swing.JFrame

Instead: import javax.swing.*;

Then use simply JFrame

The expression

new JFrame()

creates a new folder that goes in file drawer JFrame.
The statement

jf= **new** JFrame();

creates a new folder and places its name in variable jf (jf should have first been declared).
Thereafter, use

jf . *method-name* (*arguments, if any*)

to call methods of folder (object) jf.

- **Read section 1.3.**
- **Practice what you saw in class in DrJava.**
- **Try the self-review exercises on page 40.**