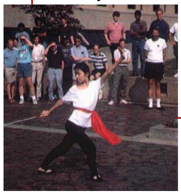
CS1110. Lecture 2, 2 Sep 2010. Objects & classes

Reading for this lecture: Sec. 1.3. **Study this section, practice** what is taught using DrJava over the weekend.

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

Reading for Tuesday, 7 Sep. Sections 1.4, (p. 41); 13.3.1 (p. 376).

Quote for the day: Computational thinking: a fundamental skill for everyone … [It] is … choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable.
Jeannette Wing



Not receiving emails from us from CMS?

1. Not registered in the CMS. Email Maria Witlox mwitlox@cs.cornell.edu, ask her to register you. Needs your netid.

2. Your email bounces. Your Cornell acct not set up correctly or place to which you forward it is having trouble. Find out:

email yourself, netid@cornell.edu, see what happens, fix it.

AEWs 1-credit AEW sections for CS1110.

Two hrs per week. Not remedial. See course website for link.

Wed 7:30-9:25pm; Mon 2:30-4:25.

Quiz Tuesday. Everyone get 100.!

1. What is a type? p. 17

2. How do you execute (carry out, perform) the assignment statement? box on p. 28

3. Be able to execute an assignment statement.

Consulting starts today.

Click “Staff” on course webpage, then the link at bottom of page

2

First-day surveys	
No programming experience:	158
< 5 months experience:	048
> 5 months experience:	047
Total	253

62%: no previous experience

CMS shows 278 students, so we are missing 25 surveys. Those who didn’t turn it in: no need to do so. It’s only to give us a good feel for the makeup of the course, and we have that.

Most frequent comments:

1. Concern about keeping up, ability to learn programming

2. Excited about learning to program

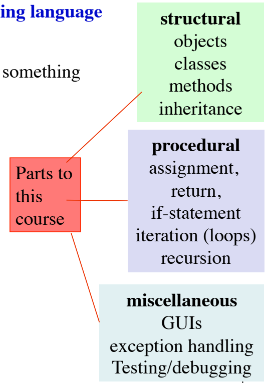
3

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: Recipe: sequence of instructions to carry out



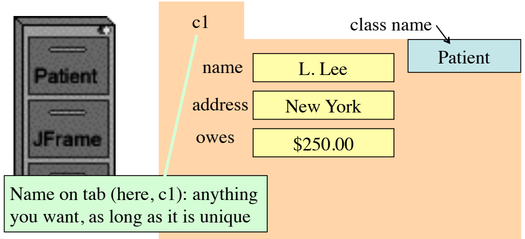
Parts to this course

structural
objects
classes
methods
inheritance

procedural
assignment,
return,
if-statement
iteration (loops)
recursion

miscellaneous
GUIs
exception handling
Testing/debugging

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



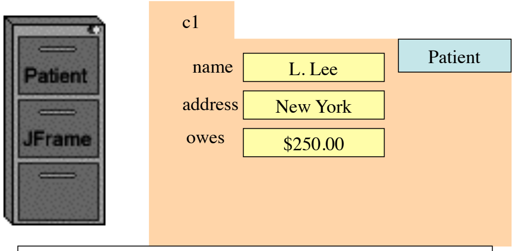
Name on tab (here, c1): anything you want, as long as it is unique

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

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

5

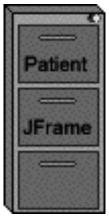
A class is a file-drawer. Contents: manila folders, each containing the same kind of information

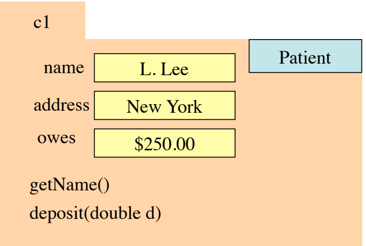


Instructions to be carried out by different people: change the name, get the name, bill the patient, receive money from patient, insert teeth xrays into the folder, …

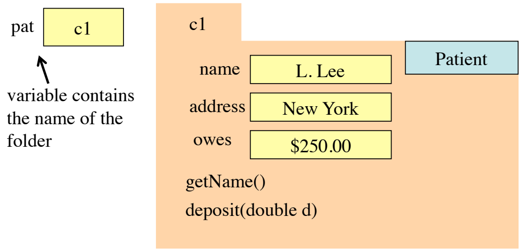
6

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



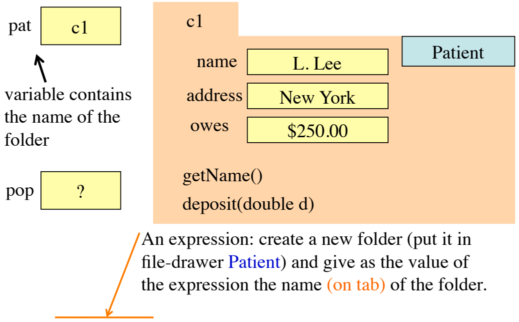


Instructions to be carried out by different people: methods.
Assume getName is a **function**: it returns a value.
Assume deposit is a **procedure**; it does a task, doesn't return value



pat.getName() function call. Its value is "L. Lee"

pat.deposit(250.0); procedure call. Subtract 250.0 from field owes.



An expression: create a new folder (put it in file-drawer Patient) and give as the value of the expression the name (on tab) of the folder.

pat= **new** Patient();
pop= new Patient();

A statement: evaluate **new** Patient() and store its value (the name of the new folder) in the variable.

Question: How much time did you spend reading the material for this lecture?

A. 0 minutes

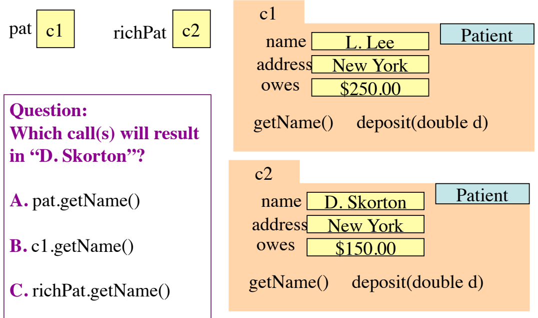
B. 1..10 minutes, with no distractions

C. 1..10 minutes with distractions

D. > 10 minutes with no distractions

E. > 10 minutes with distractions

distraction: listening to music at the same time, watching TV or something on the internet at the same time, etc.



Question:
Which call(s) will result in "D. Skorton"?

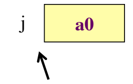
A. pat.getName()

B. c1.getName()

C. richPat.getName()

D. c2.getName()

E. both C and D



variable contains name of folder

An object (manila folder) of class

Javax.swing.JFrame

is associated with a window on your computer monitor. It has (among others) these functions:

getHeight() getWidth() getX() getY()
getTitle() isResizeable()

and these procedures:

show() hide()
setTitle(String) setSize(int, int)
setLocation(int, int) setResizable(boolean)

We will demo the use of most of these methods

In groups of 2, draw an object (manila folder) of this class, and put the name **a0** on its tab.
You don't know what the fields are, so don't draw any.

Comments from last semester

I understand classes and objects fairly well, and I thought the file drawer/file folder analogy was very helpful.

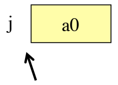
I think I'm definitely prepared for 2110. The folder/file drawer analogy was actually very helpful for a first-time Java programmer in understanding them.

I did learn the concept before coming to this class, CS1110 is really what made me understand how objects and classes work.

The folder was a great way to learn objects and classes. It simplified a very complex concept.

Teaching methods were terrible. … boxes and folders made the subject more confusing than it should be.

I'm still a bit dubious about the whole file folders and cabinets thing.



variable contains the name of the folder

- Read section 1.3.
- Practice what we did in class in DrJava.
- Try the self-review exercises on page 40.

```
j= new javax.swing.JFrame();
j.show();
…
```

Expression **new** JFrame()

Create new folder and put in file drawer JFrame.

Statement jf= **new** JFrame();

Create new folder, as above, and place its name in variable jf.

Thereafter, use

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

to call methods of folder (object) jf.

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