CS1110  Stepwise refinement, wrapper classes   29 Sept

Purpose of lecture: Give you examples of the stepwise-refinement development of methods that process strings; while showing you the OO structure of a real program.

Thursday’s lecture: no reading, but be there or be square (or lost)!
  *Recursion can be a difficult topic, but we’ll make it easy.

Prelim: 8PM Thursday 8 October (next week)
If you have a conflict and have not been contacted, email Maria Witlox. We will give a makeup. Do not ask the instructor of a course with a prelim conflict to give you a Makeup.

• Thursday: A handout will explain what is on prelim 1
• Sunday: 1-3PM. Review Session
• A3 is due Wed night on the CMS

Best. Study tip. Ever.
Cornell’s Learning Strategies Center posts a lot of great information on study skills, taking exams, time & stress management, etc. lsc.sas.cornell.edu/Sidebars/Study_Skills_Resources/SKResources.html

Every day after classes, retrieve your notes … --- you do take notes, don’t you? --- …and read them over.

This takes very little time, and yet:
1. really makes material “stick” in one’s mind, and
2. helps you figure out what you don’t understand early on, so you can get it straightened out faster.

This was a real game-changer for me.

Wrapper classes. Read Section 5.1 of class text
Soon, we’ll wish to deal with an int value as an object.
"Wrapper class" Integer provides this capability.

An instance of class Integer contains, or "wraps", one int value.
You can’t change the value. The object is immutable.

Instance methods: constructors, toString(), equals, intValue.
Static components provide important extra help.

Example of a program that deals with Strings
Creating a web page giving liberal studies courses

Java program reads the online Courses of Study webpages and extracts the courses that are liberal studies courses in A&S and CALS.

It builds tables of A&S, CALS, CA, HA, KCM, LA, and SBA courses and produces the liberal studies course website

Class Vector

An instance of class Vector maintains an expandable/shrinkable list of objects. Use it whenever you need to maintain a list of things.

Values of primitive types cannot be placed directly into the list of a Vector. That’s why we have the wrapper classes. In the interactions pane, we will do a few things, like these:

import java.util.*;
Vector v = new Vector();
v.add(new Integer(2));
v.add(3);
v.add('c');

In newer versions of Java, v.add(1) is allowed; the 1 is wrapped in an Integer object and the name of that object is added to v.

Doesn’t work in older versions.

Each primitive type has a corresponding wrapper class. When you want to treat a primitive value of that type as an object, then just wrap the primitive value in an object of the wrapper class!

<table>
<thead>
<tr>
<th>Primitive type</th>
<th>Wrapper class</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>Integer</td>
</tr>
<tr>
<td>long</td>
<td>Long</td>
</tr>
<tr>
<td>float</td>
<td>Float</td>
</tr>
<tr>
<td>double</td>
<td>Double</td>
</tr>
<tr>
<td>char</td>
<td>Character</td>
</tr>
<tr>
<td>boolean</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Integer k = new Integer(63);    int j = k.intValue();

Example of a program that deals with Strings
Creating a web page giving liberal studies courses

Java program reads the online Courses of Study webpages and extracts the courses that are liberal studies courses in A&S and CALS.

It builds tables of A&S, CALS, CA, HA, KCM, LA, and SBA courses and produces the liberal studies course website

String manipulation is key concern of this lecture. But OO structure of the program will also be discussed.
Class Webpage
Fields contain:
url, as a String
url, as an object of class
set of links on the page

Methods:
Webpage(String url) (constructor)
isHtmlPage()
getLinks() = an object that lets one read
the webpage line by line

Class DeptLink
Fields contain:
• dept name
• link to its
webpage in CoS
• Vector of all its courses

Class Courses
Fields contain:
course
e.g. ANTHR 1200
title every else you see to the left
category
HA-AS, HA CA-AS, CA SBA-AS, SBA etc.

/** Constructor: an instance who dept name and link are
contained in s. s has the form
... <a href="xxx">dept name</a> ...
where the xxx is a relative URL in directory LibStudies.prefix
Note: if s is not proper, dept name and link will be null.
*/
public DeptLink(String s) {
  // Remove ... <a href=" from s;
  Set k to index of > of the a tag;
  Store the link xxx in local variable lk;
  s = s.substring(k+2);
  Set k to index of </a>;
  Store dept name and lk in dept and link.
}

/** Constructor: instance for course given by s, in form:
<tr><td nowrap><a href="...
course name</a></td><td title=""></td></tr>
cat contains the category.
If s not proper, course, title, category will be null. */
public Course (String s, String cat) {
  // If s doesn’t start with <tr><td nowrap><a >, return;
  Remove from s everything before the course name;
  Save course name in local variable courseName;
  Remove from s everything before the title;
  Save title in local variable t;
  Store courseName, t, cat in the fields;
}