## CS1110 Wrapper classes, stepwise refinement 27 Sept

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

**Prelim:** Tuesday, 6 Oct, 7:30–9:00PM, Statler Auditorium

Learning Strategies Center posts a lot of great info 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 and read them.
Takes 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.* Professor Lillian Lee



## Wrapper classes.

Soon, need to deal with an **int** value as an object.

"Wrapper class" Integer provides this capability.

Instance of class Integer contains, or "wraps", one **int** value.

Can't change value. *immutable*.

Static components provide important extra help.

**Read Section 5.1**

a0 → ??? 5 Integer

Integer(int) Integer(String)
toString() equals(Object) intValue()
Static components:
MIN_VALUE MAX_VALUE
toString(int) toBinary(int)
valueOf(String) parseInt(String)




**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!

| Primitive type | Wrapper class |
|---|---|
| **int** | Integer |
| **long** | Long |
| **float** | Float |
| **double** | Double |
| **char** | Character |
| **boolean** | Boolean |

Each wrapper class has:
- Instance methods, e.g. equals, constructors, toString,
- Useful static constants and methods.

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

You don't have to memorize the methods of the wrapper classes. But be aware of them and look them up when necessary. Use Gries/Gries, Section 5.1, and ProgramLive, 5-1 and 5-2, as references.



## 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
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.



## Example of a program that deals with Strings
## Creating a web page giving liberal studies courses
## http://www.cs.cornell.edu/gries/ccgb/index.html

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

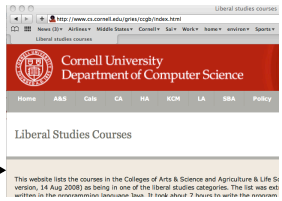

String manipulation is key concern of this lecture. But OO structure of the program will also be discussed



## Example of a program that deals with Strings
## Creating a web page giving liberal studies courses
## http://www.cs.cornell.edu/gries/ccgb/index.html

Java program: read online Courses of Study webpages and extract liberal studies courses.

Build tables of A&S, CALS, CA, HA, KCM, LA, and SBA courses. Produce liberal-studies website →



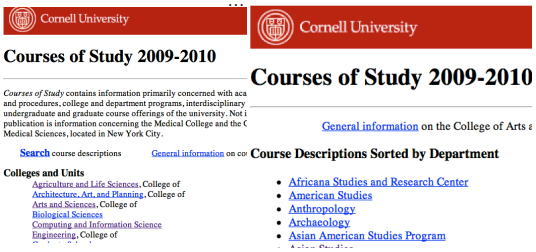

CA: cultural analysis
HA: historical analysis etc.

String manipulation is key concern of this lecture. But OO structure of the program will also be discussed

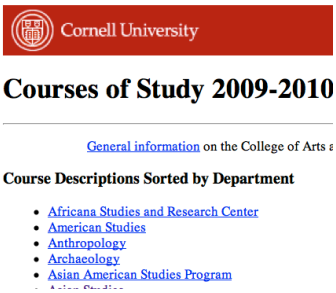
## Class Webpage

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()
getReader() = 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

```
<li>
<a href="CoScourses.php?college=AS&dept=American+Studies">
American Studies</a>
</li>
```

```
<li>
<a href="CoScourses.php?college=AS&dept=American+Studies">
American Studies</a>
</li>
```

The form is: … <a href="xxx">dept name</a> …

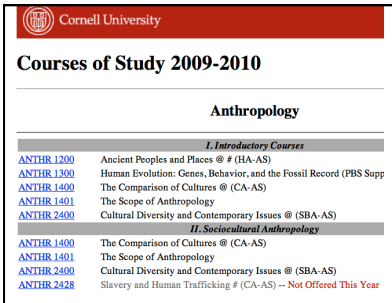
```
/** Constructor: an instance who dept name and link are
    contained in s.  s has the form
      … <a href="xxx">dept name</a> …
    where the xxx is a relative URL in directory LibStudies.prefix
    Note: if s is not proper, dept name and link will be null.
    */
```

9

```
/** Constructor: an instance who dept name and link are
    contained in s.  s has the form
      [redacted]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.
}
```

10

## Class Courses




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

```
<tr><td nowrap>
<a href="CoSdetail.php college=AS&number=1400&prefix=ANTHR&
title=The+Comparison+of+Cultures+%40+%28CA-AS%29">
ANTHR 1400</a></td><td>The Comparison of Cultures @
(CA-AS)</td></tr>
```

11

```
/** Constructor: instance for course given by s, in form:
    [redacted]title</td></tr>
    cat contains the category.
    If s not proper, course, title, category will be null. */
public Course (String s, String cat) {

}
```

12