Announcements

• If you have not picked up a syllabus, please do so

• Sign and return form on last page of syllabus – will be part of Lab 1 assignment.

• If you are interested in changing to another recitation and there are open seats, please change your registration through MyUB. If there are not open seats, please fill out form.

• If you are experiencing any kind of registration issues, please see me after class.

• Recitations began this week
Programming a Computer

• We can write instructions to the computer in machine language (the native language of that particular machine), but this is difficult and time-consuming.

• So, early in computer programming history, we created assembly language.

```
ADD r1 r2  3+4
STOR r3 r6
```
Assembly Language

• Simple mnemonics that indicate the type of action to be performed.
• Low-level language
• There is a one-to-one correspondence between the lines of assembly language and the machine language for the particular machine.

Higher-level Languages

• Assembly language works well, but a need was recognized to make programming languages more like human languages, and higher-level languages were developed.
• Higher-level languages are more complicated than assembly language
• There is no one-to-one correspondence between one line of a higher-level language and machine language
The Boehm-Jacopini Theorem

- If interested in more information, feel free to check out Bill Rapaport’s webpage on the Boehm-Jacopini Theorem:
  http://www.cse.buffalo.edu/~rapaport/111F04/greatidea3.html
- In order to compute, we need the following three things:
To program

- Sequencing - define the order things will be executed in
- Selection - choice
- Repetition

In order to create programs, we need several software tools to help us.
Our IDE: Eclipse

- Check out the Resources/Course-Specific Resources section of the course website for information about downloading Eclipse on your own machine
Object-oriented programs

System of objects that work together to solve some problem

Objects
- can do things
- have properties & state
Recall our demonstration

<table>
<thead>
<tr>
<th>If I tell you...</th>
<th>...object 1, you should...</th>
<th>...object 2, you should...</th>
<th>...object 3, you should...</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>Raise and lower your arms repeatedly</td>
<td>Start counting out loud, from zero (somewhat slowly) 0...1...2...3</td>
<td>Do nothing</td>
</tr>
<tr>
<td>STOP</td>
<td>Put arms down and remain still</td>
<td>Stop counting, but remember where you left off</td>
<td>Do nothing</td>
</tr>
<tr>
<td>CONTINUE</td>
<td>Do nothing</td>
<td>Keep counting from where you left off</td>
<td>Do nothing</td>
</tr>
<tr>
<td>ANYTHING ELSE</td>
<td>Do nothing</td>
<td>Do nothing</td>
<td>Do nothing</td>
</tr>
</tbody>
</table>