Lecture 1: Introduction

Today’s topics:
1. Course information
2. Computer basics
CMSC 131 Section 03*

- Name: “Object-Oriented Programming I”
- Instructor: Bonnie Dorr
- Lab leader: Matthew Mah
- Class meetings
  - Lecture: MWF 3-3:50 CSIC 2117
  - Lab sections (CSIC 2118)
    - 0301: MW 1-1:50
    - 0302: MW 2-2:50
Coordination with Other Sections

- Two other sections of CMSC 131, both taught by Fawzi Emad
- All sections will be closely coordinated:
  - Same lecture material on same day
  - Same projects
  - Same labs
  - Coordinated exams
What Is This Course?

- A *fast-paced* introduction to techniques for writing computer programs!
- There will be a lot of work!
- Keys to success
  - Attend all classes and lab sections
  - Ask questions
  - Start assignments early
  - Get help early if you are having trouble
  - Study every day
  - *Check announcements on course web-page every day*
Course Web-Page

- www.cs.umd.edu/class/spring2007/cmsc131/
- Check daily!
- Review:
  - Announcements
  - Syllabus
  - Contact
  - Schedule
  - Lecture slides
Book

Java™ Software Solutions: Foundations of Program Design (5th edition), by Lewis & Loftus

- Lectures do not follow book closely
- Book is very useful reference
Course Software

- **Eclipse**
  - An IDE (integrated development environment)
  - You will use it for writing Java™ programs
  - Access to Eclipse (it’s free!)
    - You can install it on your own machine: [http://www.cs.umd.edu/eclipse](http://www.cs.umd.edu/eclipse)
    - Also accessible in Workstations at Maryland (WAM) labs around campus: [http://www.wam.umd.edu/](http://www.wam.umd.edu/)

- **CVS (Concurrent Versions System)**
  - A version-management system
  - You will use it for submitting your projects
  - We will talk more about this later
Study Questions

- Available on web-page
  - Login: study
  - Password: daily
- Look at them on evenings before class; they will help you keep up
Computer Organization

- Hardware: physical parts of computer
  - Monitor, mouse, keyboard
  - Chips, boards
  - Cables, cards
  - etc.

- Software: non-physical ("logical") parts of computer
  - Programs = instructions for computer to perform
Hardware Overview

- **CPU** = central processing unit
  - Executes the "instructions" in programs

- **Main memory** = random-access memory = “RAM”
  - Stores data that CPU accesses, including instructions
  - FAST, but temporary; wiped out when computer is shut off!

- **Secondary memory**: Hard disks, CDs, DVDs, flash memory, etc.
  - Stores data that can be loaded into main memory
  - SLOWER, but permanent

- **I/O devices**
  - How you communicate with your machine
  - Keyboard, monitor, mouse, speakers, etc.

- **Networking equipment**
  - How others communicate with your machine
  - Networking “cards”, cables, etc.
Main Memory

- Computer data consists of 0’s and 1’s (really!)
- A cell in main memory that can hold either a 0 or 1: *bit*
- A sequence of 8 bits: *byte*
- A sequence of 4 bytes: *word*
- Main memory: table of bytes indexed by “addresses”

<table>
<thead>
<tr>
<th>Address</th>
<th>Byte value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 0 0 1 1 1 1 0 1</td>
</tr>
<tr>
<td>2</td>
<td>0 0 0 1 1 0 0 1</td>
</tr>
<tr>
<td>3</td>
<td>1 1 1 1 1 1 0 1</td>
</tr>
<tr>
<td>4</td>
<td>1 1 0 0 0 1 0 0</td>
</tr>
</tbody>
</table>
How Many Different Values in a...

- Bit?
  2
- Two bits?
  $4 = 2 \times 2$
- Byte?
  $256 = 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 = 2^8$
- Word?
  $4,294,967,296 = 2^{32}$
How Are Characters, Etc., Represented?

Via encoding schemes

Example: ASCII (American Standard Code for Information Interchange)

- Standard for encoding character values as bytes
- In ASCII:
  - ‘A’ 01000001
  - ‘a’ 01100001
  - ‘,’ 00101100
  - etc.

There are other character encoding schemes also: Shift-JIS, Unicode, etc.
Other Standard Terminology

- 1 KB = 1 “kilobyte” = $2^{10}$ bytes = 1,024 bytes
- 1 MB = 1 “megabyte” = $2^{10}$ KB = 1,024 KB
- 1 GB = 1 “gigabyte” = $2^{10}$ MB = 1,024 MB
Software Overview

1. **Operating system**: manages computer's resources; typically runs as soon as computer is turned on. Typical responsibilities:
   - *Process management*
     Determines when, how programs will run on CPU time
   - *Memory management*
     Controls access to main
   - *I/O, window system, network control*
     Performs low-level drawing, communication operations
   - *Security*
     Manages user IDs, passwords, file protections, etc.

2. **Applications**: programs users interact directly with; usually are explicitly run.
   Examples:
   - Word processors
   - Games
   - Spreadsheets
   - Music software,
   - Etc
How Programs Are Executed

Program “foo” initially stored in secondary storage

Program copied into main memory

CPU executes program instruction-by-instruction