Welcome to CSCI-UA 101
Schedule

• Introductions
• Syllabus
• Expectations
• Let’s get this started
• hw assignment
Hi, I’m Brett.

• Adjunct Professor, Courant Institute of Mathematical Sciences
Introductions

• Preferred name

• Programming experience - languages, etc

• Programming interests? What do you want to learn this semester?

• CS major or minor
A bit about me

• Adjunct Professor, Courant Institute of Mathematical Sciences

• Senior Developer of Interactive Exhibits, American Museum of Natural History
Syllabus

• Let’s check out the website!

http://cs.nyu.edu/courses/spring17/CSCI-UA.0101-006
Expectations of you

- Come to class!
  - (I know it's super early)
  - The class will get harder. Especially the second half, and everything builds on previous topics
  - You can ask questions, get help, the book will make more sense
  - Come to office hours or see the tutors if things aren’t making sense. Don’t fall behind - it’ll be hard to catch up
- Read the Book! The students who do well in the class stay on top of the reading
- Participate!
  - It’ll be more fun, promise
  - You’ll stay awake
  - It might help your grade…
Expectations of me

• Come to class!
  • (prepared)

• Write clear homework assignments

• Give prompt feedback

• Be responsive to emails, etc

• Hold office hours
Let’s get started
This class is about programming
• Learning to program is learning to tell a computer what to do
So what is a computer?

- Electronic device to store and process data
- Hardware
  - Central Processing Unit (CPU)
  - Memory
  - Storage devices
  - Input/output devices
  - comm devices
- Software
CPU

- Computer’s brain
- 2 parts:
  - Control unit
  - Arithmetic/logic unit
- Made of transistors
Storing data

• 1’s and 0’s

• bits

• byte = 8 bits
### Counting in Binary and Decimal

<table>
<thead>
<tr>
<th></th>
<th>32</th>
<th>16</th>
<th>08</th>
<th>04</th>
<th>02</th>
<th>01</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** You don’t need to memorize converting to binary! That’s the computer’s job
Memory

• Computer’s work area

• every byte in memory has a unique address

• RAM = random-access memory

• important when writing programs!
Talking to the machine

- High-level languages
- Assembly language
- Machine code
- Electronic circuitry
Machine code

- Instructions in binary

- like this: 1101101010011010
Assembly Language

Start:

sei ; pretty standard 6502 type init here
cld
lda #%00010000 ; init PPU control register 1
sta PPU_CTRL_REG1
ldx #$ff ; reset stack pointer
txs

VBlank1: lda PPU_STATUS ; wait two frames
bpl VBlank1

VBlank2: lda PPU_STATUS
bpl VBlank2

ldy #ColdBootOffset

High-level language

- Platform-independent
- English-like, easier to learn and use
- You write source code, translated using an interpreter or compiler
Java

• Designed in 1991 for embedded systems, called oak

• Renamed Java in 95, redesigned for web apps

• Super popular (http://www.tiobe.com/tiobe-index/)
Wait, Java?

- Why Java? It’s not the new hotness in terms of languages, but it is a great language to learn coding principles.

- If you know Java, jumping to C# or C++ or another language is a lot easier than from some other scripting or interpreted languages.

- Plus…
Java powers all this:

- Server technology
- websites
- desktop apps
- mobile apps (Android)
- Interactive installations and data visualizations (Processing)
- Controls MARS ROVERS
- etc…
Some definitions


• **Application program interface (API) (JavaDoc!)** - library. predefined classes and interfaces. [https://docs.oracle.com/javase/8/docs/api/index.html](https://docs.oracle.com/javase/8/docs/api/index.html)

• **Java Standard Edition (Java SE)** - We’ll use this one. For client-side applications

• **Java Enterprise Edition (Java EE)** - server-side apps (servlets), etc

• **Java Micro Edition (Java ME)** - mobile devices like old cellphones
How to program in java

• Make sure the JDK is installed!

• Use a text editor to write java program, then compile, then run.

• OR use an IDE (Integrated development environment)
public class HelloWorld {
    public static void main(String[] args) {
        // Displays message to console
        System.out.println(“Hello World!”);
    }
}
Homework

• Read chapters 1&2

• Install JDK and Eclipse

• Assignment #00