Welcome to CS240

Foundations of Computer Systems
The Plan

1) What is CS240?
2) Why take CS240?
3) How will CS240 work?
4) Jump into the foundations of computer hardware
What is CS240?

Other areas of CS...

CS111, CS230, CS231, CS235, CS251
• What can a program do?
• How can a program solve a problem?
• How do you structure a program?
• How do you know it is correct or efficient?
• How hard is it to solve a problem?
• How is computation expressed?
• What does a program mean?
• ...

A BIG question is missing...
1) What is CS240?

How do computers work?

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```

Welcome to DrJava. Working directory is /Users/bpw/courses/cs240/cs240f14
> run HelloWorld
Hello, world!

Running main method of Current Document

Circuitboard image: CC-BY-NC-SA ifixit.com
Big Idea: Abstraction

interface
implementation

Algorithm, Data Structure, Application

Programming Language

Compiler/Interpreter

Operating System

Instruction Set Architecture

Microarchitectue

Digital Logic

Devices (transistors, etc.)

Solid-State Physics
Big Idea: Abstraction

Simple, general interfaces:
- Hide complexity of efficient implementation.
- Make higher-level systems easy to build.
- But they are not perfect.

Recurring Subplots:

Representation of data and programs

Translation of data and programs

Control flow within/across programs
Ada Lovelace writes the first computer program

Charles Babbage designs Analytical Engine

Prototype of Analytical Engine, (was never actually built), Science Museum, London

Image: public domain

George Boole describes formal logic for computers

Boolean Algebra

Countess Ava Lovelace, 1840s

George Boole, 1860s

University College Cork, Ireland

Image: public domain
1900s 1910s 1920s 1930s 1940s 1950s 1960s 1970s 1980s

Human computers

Alan Turing, 1940s
Imitation Game, 2014
Image: Flikr mark_am_kramer, Imitation Game poster

NASA computers, 1953
Hidden Figures, 2016
Image: NASA/JPL/Caltech, Hidden Figures
ENIAC (Electronic Numerical Integrator and Computer), First Turing-complete all-electronic programmable digital computer. University of Pennsylvania, 1940s

Image: public domain
Jean Jennings Bartik and Frances Bilas Spence with part of ENIAC.

*The programmers of ENIAC were six women.*

http://eniacprogrammers.org/, http://sites.temple.edu/topsecretrosies/

Image: public domain
Programming 1940s-style with switches and cables.
programs are data

machine translates instructions to control flow

Manchester “Baby” SSEM (Small-Scale Experimental Machine), replica first stored-program computer -- University of Manchester (UK), 1948
PDP-11
"minicomputers"

http://simh.trailing-edge.com/

http://www.pcworld.com/article/249951/if_it_aint_broke_dont_fix_it_ancient_computers_in_use_today.html?page=2
<table>
<thead>
<tr>
<th>1930s</th>
<th>1940s</th>
<th>1950s</th>
<th>1960s</th>
<th>1970s</th>
<th>1980s</th>
<th>1990s</th>
<th>2000s</th>
<th>2010s</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://i.imgur.com/1234567890.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/0987654321.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/8765432109.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/7654321098.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/6543210987.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/5432109876.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/4321098765.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/3210987654.png" alt="Image" /></td>
<td><img src="https://i.imgur.com/2109876543.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Images:
- "Ibm pc 5150" by Ruben de Rijcke - Own work. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons - [Link](http://commons.wikimedia.org/wiki/File:Ibm_pc_5150.jpg)
- "IBM PC Motherboard (1981)" by German - Own work. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons - [Link](http://commons.wikimedia.org/wiki/File:IBM_PC_Motherboard.png)
- "Macintosh-motherboard" by Shieldforyoureyes Dave Fischer - Own work. Licensed under Creative Commons Attribution-Share Alike 3.0 via Wikimedia Commons - [Link](http://commons.wikimedia.org/wiki/File:Macintosh-motherboard.png)
<table>
<thead>
<tr>
<th>1930s</th>
<th>1940s</th>
<th>1950s</th>
<th>1960s</th>
<th>1970s</th>
<th>1980s</th>
<th>1990s</th>
<th>2000s</th>
<th>2010s</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ENIAC</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Year</strong></td>
<td>1946</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Weight</strong></td>
<td>30 tons</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Volume</strong></td>
<td>2,400 ft$^3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Cost (USD, 2014)</strong></td>
<td>$6,000,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>few 1000 ops/sec</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>~100 bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Power</strong></td>
<td>150,000 W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Input/Output</strong></td>
<td>Switches, lights, later punchcards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Production</strong></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| **iPhone 7** | | | | | | | | |
| **Year** | 2016 | | | | | | | |
| **Weight** | 4.87 oz | | | | | | | |
| **Volume** | 4.02 in$^3$ | | | | | | | |
| **Cost (USD)** | $749 | | | | | | | |
| **Speed** | 50,000,000,000 ops/sec | | | | | | | |
| **Memory** | 2,147,483,648 bytes (2 GB) | | | | | | | |
| **Power** | <5W | | | | | | | |
| **Input/Output** | 3D Touchscreen, audio, camera, wifi, cell, ... | | | | | | | |
| **Production** | 21 million sold in first quarter | | | | | | | |
Modern Computer Organization

**Processor**

Executes instructions.

**Memory**

Stores program code + data during execution.

**Bus**

- Persistent Storage
- Network
- USB
- Display

Timeline:
- 1930s
- 1940s
- 1950s
- 1960s
- 1970s
- 1980s
- 1990s
- 2000s
- 2010s
Modern Computer Organization

Processor

Stores program code + data during execution.

Memory

Executes instructions.

Processor repeats:
1. fetch instruction
2. fetch data used by instruction
3. execute instruction on data
4. store result or choose next instruction
Desired computation represented as instructions.

Hardware/Software Interface

Physical implementation of instructions and resources.
Microarchitecture (Implementation of ISA)

- Instruction Fetch and Decode
- Registers
- ALU
- Memory
Instruction Set Architecture (HW/SW Interface)

**processor**
- Instruction Logic
- Registers

**memory**
- Encoded Instructions
- Data

**Instructions**
- Names, Encodings
- Effects
- Arguments, Results

**Local storage**
- Names, Size
- How many

**Large storage**
- Addresses, Locations

**Computer**
Machine Instructions

(adds two values and stores the result)

00000010100010101100100000010000

Instruction Set Architecture specification
Assemblers and Assembly Languages

```
addl %eax, %ecx
```

000000101000101100100000010000

Assembly Language specification

assembly program → Assembler → machine code program → Hardware
Higher-Level Programming Languages

\[ x = x + y; \]

`addl %eax, %ecx` → `00000010100010101100100000010000`

Programming Language specification

- Compile time:
  - high-level language program
  - Compiler
  - assembly program
  - Assembler
  - machine code program

- Run time:
  - Hardware
A-o: first compiler, by Grace Hopper

Early 1950s
Maybe closer to assembler/linker/loader
Later:
  B-0 → FLOW-MATIC
  → COBOL, late 50s

Jean Sammet also involved
  • headed first sci comp group at Sperry in the '50s
  • Later first female president of ACM
  • Mount Holyoke alum, class of 1948
More and more layers...

- Operating systems
- Virtual machines
- Hypervisors
- Web browsers
- ...
CS 240: a story in 3 parts

Hardware *implementation*
   From transistors to a simple computer

Hardware-software *interface*
   From instruction set architecture to C

Abstraction and application for practical systems
   Memory hierarchy
   Computer Security
   Operating systems
   Embedded Systems

Sometimes system abstractions "leak."
Implementation details affect your programs.
int \neq integer
float \neq real

int x = ...;
x*x >= 0 ?
  40000 * 40000 == 1600000000
  50000 * 50000 == -1794967296

float a = ..., b = ..., c = ...;
(a + b) + c == a + (b + c) ?
  (-2.7e23 + 2.7e23) + 1.0 == 1.0
  -2.7e23 + (2.7e23 + 1.0) == 0.0

Examples of Leaky Abstractions
Exploded due to cast of 64-bit floating-point number to 16-bit signed number. Overflow.

"... a Model 787 airplane ... can lose all alternating current (AC) electrical power ... caused by a software counter internal to the GCUs that will overflow after 248 days of continuous power. We are issuing this AD to prevent loss of all AC electrical power, which could result in loss of control of the airplane." --FAA, April 2015
Arithmetic Performance

x / 973  
x / 1024

Memory Performance

```c
void copyji(int src[2048][2048],
            int dst[2048][2048])
{
    int i,j;
    for (j = 0; j < 2048; j++)
        for (i = 0; i < 2048; i++)
            dst[i][j] = src[i][j];
}

void copyij(int src[2048][2048],
            int dst[2048][2048])
{
    int i,j;
    for (i = 0; i < 2048; i++)
        for (j = 0; j < 2048; j++)
            dst[i][j] = src[i][j];
}
```

several times faster due to hardware caches
The **GHOST vulnerability** is a buffer overflow condition that can be easily exploited locally or remotely, which makes it extremely dangerous. This vulnerability is named after the `GetHOST` function involved in the exploit.

Cyber-Safe

All computers are flawed -- and the fix will take years

by Selena Larson  @selenalarson

January 26, 2018: 12:07 PM ET

Meltdown and Spectre

A Heart Device Is Found Vulnerable to Hacker Attacks

By BARNABY J. FEDER
Published: March 12, 2008

To the long list of objects vulnerable to attack by computer hackers, add the human heart.

The threat seems largely theoretical. But a team of computer security researchers plans to report Wednesday that it had been able to gain wireless access to a combination heart defibrillator and pacemaker.
2) Why take CS240?

- Learn how computers execute programs.
- **Build software tools** and appreciate the value of those you use.
- Deepen your appreciation of *abstraction*.
- Learn enduring **system design principles**.
- Improve your **critical thinking** skills.
- Become a **better programmer**:
  - Think rigorously about execution models.
  - Program carefully, defensively.
  - Debug and reason about programs effectively.
  - Identify limits and impacts of abstractions and representations.
  - Learn to use software development tools.
- **Foundations** for:
  - Compilers, security, computer architecture, operating systems, parallel computing...
- Have fun and feel accomplished!
Also: C programming language

- Invented to build UNIX operating system, 1970s
  - OS manages hardware, C close to machine model

- Simple pieces look like Java:
  - if, while, for, local variables, assignment, etc.

- Other pieces do not:
  - structs vs. objects, functions vs. methods
  - addresses, pointers
  - no array bounds checks
  - weak type system

- Important language, still widely used, but many better PL ideas have come along since.
3) How will CS240 work?

https://cs.wellesley.edu/~cs240/