ECE 2400 / ENGRD 2140
Computer Systems Programming
Course Overview

Christopher Batten
School of Electrical and Computer Engineering
Cornell University

http://www.csl.cornell.edu/courses/ece2400
ECE 2400 / ENGRD 2140
Computer Systems Programming

What is Computer Systems Programming?

Activity: Comparing Algorithms

Trends in Computer Systems Programming

Course Logistics
The Computer Systems Stack

Application

Gap too large to bridge in one step (but there are exceptions, e.g., a magnetic compass)

Technology
In its broadest definition, computer engineering is the development of the abstraction/implementation layers that allow us to execute information processing applications efficiently using available manufacturing technologies.

Computer Engineering is at the interface between hardware and software and considers the entire system.

Traditional Computer Science

Traditional Electrical Engineering
Python for Application-Level Programming

- High-level, user-facing software
- Enable productively developing applications that provide new functionality to users
- Enable productively collecting, analyzing, visualizing data
- Sometimes called a productivity-level language
C/C++ for System-Level Programming

- Connects application software to the low-level computer hardware
- Enables carefully managing performance and resource constraints
- Sometimes called an efficiency-level language

Diagram:
- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

- Application-Level Software
- System-Level Software
The standard Python interpreter is called CPython and it is written in C!
Computer Systems Programming is Diverse

- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

- Python, MATLAB
- Ruby, Javascript
- SQL, LINQ
- NumPy
- GUI frameworks

- Interpreters
- Compilers
- Databases
- Numerical libraries
- Operating systems
- Embedded control
Aside: C/C++ for Application-Level Software

Computer Engineering

- Application
- Algorithm
- Programming Language
- Operating System
- Compiler
- Instruction Set Architecture
- Microarchitecture
- Register-Transfer Level
- Gate Level
- Circuits
- Devices
- Technology

Application-Level & System-Level Software
<table>
<thead>
<tr>
<th>Python Programming Language</th>
<th>C/C++ Programming Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduced: 1991</td>
<td>Introduced: 1972(C), 1979(C++)</td>
</tr>
<tr>
<td>Most of the machine details</td>
<td>Most of the machine details</td>
</tr>
<tr>
<td>Programmer gives up some</td>
<td>Programmer is in complete</td>
</tr>
<tr>
<td>control for improved</td>
<td></td>
</tr>
<tr>
<td>productivity</td>
<td></td>
</tr>
<tr>
<td>Easily supports multiple</td>
<td>Easily supports multiple</td>
</tr>
<tr>
<td>programming paradigms</td>
<td>programming paradigms</td>
</tr>
<tr>
<td>Extensive standard library</td>
<td>More limited standard library</td>
</tr>
<tr>
<td>is included</td>
<td>included</td>
</tr>
<tr>
<td>Slow and memory inefficient</td>
<td>Fast and memory efficient</td>
</tr>
</tbody>
</table>
Comparing the Popularity of Python vs. C/C++

<table>
<thead>
<tr>
<th>Language Rank</th>
<th>Types</th>
<th>Spectrum Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Python</td>
<td>🌐📱💻</td>
<td>100.0</td>
</tr>
<tr>
<td>2. C++</td>
<td>📱💻📱</td>
<td>99.7</td>
</tr>
<tr>
<td>3. Java</td>
<td>🌐📱💻</td>
<td>97.5</td>
</tr>
<tr>
<td>4. C</td>
<td>📱💻📱</td>
<td>96.7</td>
</tr>
<tr>
<td>5. C#</td>
<td>🌐📱💻</td>
<td>89.4</td>
</tr>
<tr>
<td>6. PHP</td>
<td>🌐📱</td>
<td>84.9</td>
</tr>
<tr>
<td>7. R</td>
<td>📱💻</td>
<td>82.9</td>
</tr>
<tr>
<td>8. JavaScript</td>
<td>🌐📱</td>
<td>82.6</td>
</tr>
<tr>
<td>9. Go</td>
<td>🌐💻</td>
<td>76.4</td>
</tr>
<tr>
<td>10. Assembly</td>
<td>📱💻</td>
<td>74.1</td>
</tr>
<tr>
<td>11. Matlab</td>
<td>📱</td>
<td>72.8</td>
</tr>
</tbody>
</table>

The 2018 Top Programming Languages, IEEE Spectrum
Comparing the Performance of Python vs. C/C++

The Computer Language Benchmarks Game
Program = Algorithm + Data Structure

While this course covers C/C++ and system-level programming, this course also builds off of your prior programming experience to further develop your understanding of algorithms and data structures.

- **Algorithm**: Clear set of steps to solve any problem instance in a particular class of problems.
- **Data Structure**: Way of efficiently organizing and storing data along with methods for accessing and manipulating this data.
ECE 2400 / ENGRD 2140
Computer Systems Programming

What is Computer Systems Programming?

Activity: Comparing Algorithms

Trends in Computer Systems Programming

Course Logistics
Activity: Comparing Algorithms

Application: Sort 16 numbers

Activity Steps
1. Half the class will use Algorithm A, half uses Algorithm B
2. When instructor starts timer, flip over worksheet
3. Sort 16 numbers using assigned algorithm
4. Lookup when completed and write time on worksheet
5. Raise hand
6. When everyone is finished, then analyze data

Algorithm A
(remote students use: http://tiny.cc/ece2400-activity)

repeat 16 times
find smallest number not crossed off in input list
copy smallest number to next open entry in output list
cross smallest number off input list
Activity: Comparing Algorithms

Algorithm B

repeat 8 times, once for each pair in column 1
  copy smallest from input pair into next entry in column 1
  copy largest from input pair into next entry in column 1

repeat 4 times, once for group of 4 in column 2
  repeat 4 times
    compare top two numbers not crossed off in both groups
    copy smallest number to next open entry in column 2
    cross smallest number off input list

... and so on ...
ECE 2400 / ENGRD 2140
Computer Systems Programming

What is Computer Systems Programming?

Activity: Comparing Algorithms

Trends in Computer Systems Programming

Course Logistics
Trend towards IoT and Cloud w/ Novel Hardware

Roughly every decade a new, smaller, lower priced computer class forms based on a new programming platform resulting in entire new industries.
Trend towards IoT and Cloud w/ Novel Hardware

Roughly every decade a new, smaller, lower priced computer class forms based on a new programming platform resulting in entire new industries.

**Cloud Computing**
- Often requires low-latency, high-throughput to meet overall application requirements
- Increasingly w/ specialized HW

**Internet-of-Things**
- Very limited resource constraints (e.g., energy, memory)
- Requires carefully managing these resources to meet overall application requirements
- Increasingly w/ specialized HW
Example Application: Image Recognition

Starfish

Dog
Machine Learning (ML): Training vs. Inference

Training
- Many images
- Model =?
- "starfish"
- Backward error

Inference
- Few images
- Model forward
- "dog"
Computer Systems Programming in ML

Google TPU
- Training is done using the TensorFlow C++ framework
- Training can take weeks
- Google TPU is custom chip
- High-level ML frameworks use C++ under the hood

Movidius Myriad 2
- Custom chip for ML on embedded IoT devices
- Carefully crafted C/C++ ML libraries for inference
- Embedded control also in C/C++
ECE 2400 / ENGRD 2140
Computer Systems Programming

What is Computer Systems Programming?

Activity: Comparing Algorithms

Trends in Computer Systems Programming

Course Logistics
ECE 2400 Within the Engineering Curriculum

ECE 2400 is also an ENGRD and thus satisfies the engineering distribution requirement.

ECE 2400 can be an excellent way to generally incorporate programming into your non-ECE engineering curriculum.
Course Objectives

- **describe** a variety of algorithms and data structures and how to analyze these algorithms and data structures in terms of time and space complexity

- **apply** the C/C++ programming languages to implement algorithms and data structures using different programming paradigms

- **evaluate** algorithm and data structure alternatives and make a compelling theoretical and/or practical argument for one approach

- **create** non-trivial C/C++ programs (roughly 1,000 lines of code) and the associated testing strategy from an English language specification

- **write** concise yet comprehensive technical reports that describe a program implemented in C/C++, explain the testing strategy used to verify functionality, and evaluate the program to characterize its performance and memory usage
Course Structure

- **Part 1: Procedural Programming**
  - introduction to C, variables, expressions, functions, conditional & iteration statements, recursion, static types, pointers, arrays, dynamic allocation

- **Part 2: Basic Algorithms and Data Structures**
  - lists, vectors, complexity analysis, insertion sort, selection sort, merge sort, quick sort, hybrid sorts, stacks, queues, sets, maps

- **Part 3: Multi-Paradigm Programming**
  - transition to C++, namespaces, flexible function prototypes, references, exceptions, new/delete, *object oriented programming* (C++ classes and inheritance for dynamic polymorphism), *generic programming* (C++ templates for static polymorphism), *functional programming* (C++ functors and lambdas), *concurrent programming* (C++ threads and atomics)

- **Part 4: More Algorithms and Data Structures**
  - trees (binary trees, binary search trees), tables (lookup tables, hash tables), graphs (DFS, BFS, shortest path first, minimum spanning trees)
zyBook: New Interactive Online Textbook

1.2 Programming basics

A first program

A simple C program appears below.

- A program starts in main(), executing the statements within main's braces (), one at a time.
- Each statement typically appears alone on a line and ends with a semicolon, as English sentences end with a period.
- The int wage statement creates an integer variable named wage. The wage = 20 statement assigns wage with 20.
- The printf statements output various values.
- The return 0 statement ends the program (the 0 tells the operating system the program ended without error).

The following code (explained later) at the top of a file enables the program to get input and put output:

```c
#include <stdio.h>
```
Programming Assignments

- **PA1–3: Fundamentals**
  - PA1: Math functions
  - PA2: List and Vector Data Structures
  - PA3: Sorting Algorithms

- **PA4–5: Handwriting Recognition System**
  - PA5: Linear vs. Binary Searching
  - PA5: Trees vs. Tables

- **Every programming assignment involves**
  - C/C++ “agile” programming
  - State-of-the-art tools for build systems, version control, continuous integration, code coverage
  - Performance measurement
  - Short technical report
Frequently Asked Questions

▶ I have not taken CS 1110 nor CS 1112, can I take this class?
  ▶ We assume some basic programming experience, discuss with instructor

▶ ECE Majors – How does ECE 2400 satisfy degree requirements?
  ▶ ECE 2400 can count as your second ENGRD course
  ▶ ECE 2400 can count as an outside-ECE technical elective
  ▶ ECE 2400 satisfies the ECE advanced programming requirement

▶ CS Majors – Can I use ECE 2400 in place of CS 2110?
  ▶ Kind of but you should probably take CS 2110

▶ ECE/CS Dual Majors – Can I use ECE 2400 in place of CS 2110?
  ▶ Absolutely! (NEW)

▶ CS Minors – Can I use ECE 2400 in place of CS 2110?
  ▶ Absolutely! (NEW)
Frequently Asked Questions

- **Other Majors** – How does ECE 2400 satisfy degree requirements?
  - ECE 2400 can count as one of your two required ENGRD courses

- **ECE MEngs** – How does ECE 2400 satisfy degree requirements?
  - ECE 2400 can count as a technical elective, but you remember you can only use one ECE undergraduate course as a technical elective

- Should I take both ECE 2400 and CS 2110?
  - Sure! (recall popularity and performance data)
Take-Away Points

- Computer systems programming involves developing software to **connect** the low-level computer hardware to high-level, user-facing application software and usually **requires careful consideration of performance and resource constraints**.

- We are entering an **exciting era** where computer systems programming will play a **critical role in enabling both cloud computing and the internet-of-things**.