ECE 650
Systems Programming & Engineering
Spring 2018

Introduction

Tyler Bletsch
Duke University

Slides are adapted from Brian Rogers (Duke)
Welcome to **ECE 650: Systems Programming & Engineering**

**Instructor**
- Prof. Tyler Bletsch
  - Can call me “Tyler” or “Dr. Bletsch” or whatever
  - Office hours will be on course page, and I also accept appointments

**TAs**
- Siyang Chen, Yuchen Zhou
  - Office hours will be posted on class page
Getting Info

- **Course Web Page**: static info
  - [http://people.duke.edu/~tkb13/courses/ece650/](http://people.duke.edu/~tkb13/courses/ece650/)
    - Syllabus, schedule, slides, assignments, rules/policies, prof/TA info, office hour info
    - Links to useful resources

- **Piazza**: questions/answers
  - Post all of your questions here
  - Questions should be “public” unless good reason otherwise
  - **No code** in public posts!

- **Sakai**: just assignment submission and gradebook
Overview of this Class

- You have a foundation of knowledge in:
  - Computer architecture
  - Basics of OS and networking
  - Programming (strong C and C++ skills)
  - Unix development tools

- Using these skills, we will learn to:
  - Write programs that relate to the design of operating systems
  - Write programs for system-level software
What is Systems Programming?

- Of course we should consult Wikipedia!
- **Applications programming**: services for users
- **Systems programming**: services for other software
- “System programming requires a great degree of hardware awareness.”
Course Topics

- Operating systems
  - Concurrency and inter-process communication (IPC)
  - Interactions between user space and kernel space
  - Process management and scheduling
  - Security, malware, exploits
  - System boot process
  - I/O systems
  - File systems
  - Virtual memory management for the OS
  - Hypervisors: virtualizing the OS

- Networking
  - Link layer (hubs, switches, etc.)
  - IP, routing review, BGP routing, UDP, TCP
  - Flow and congestion control
  - DNS, HTTP, ICMP

- Databases
  - Tuples, tables, schema, relational algebra
  - SQL basics, C programming, ACID, transactions, isolation
  - Data organization, including B-trees and indexing
  - Distributed hash tables
  - Distributed files systems
  - MapReduce and Hadoop
### Grading Breakdown

<table>
<thead>
<tr>
<th>Assignment</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homeworks</td>
<td>55%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>15%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>30%</td>
</tr>
</tbody>
</table>

Partial credit is available – provide detail in your answers to seek it!

Late homework submissions incur penalties as follows:
- Submission is 0-24 hours late: total score is multiplied by 0.9
- Submission is 24-48 hours late: total score is multiplied by 0.8
- Submission is more than 48 hours late: total score is multiplied by the Planck constant (in J·s)

NOTE: If you feel *in advance* that you may need an extension, contact the instructor.

~6.6×10⁻³⁴

These assignments are looooooooolong. START EARLY.
Grade Appeals

• All regrade requests must be in writing
  • Email the UTA who graded the question (we’ll indicate who graded what)

• After speaking with the TA, if you still have concerns, contact the instructor

• All regrade requests must be submitted no later than 1 week after the assignment was returned to you.
<table>
<thead>
<tr>
<th>Score Range</th>
<th>Grade</th>
<th>Score Range</th>
<th>Grade</th>
<th>Score Range</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt; 97</td>
<td>A+</td>
<td>92 - 96.9</td>
<td>A</td>
<td>90 - 91.9</td>
<td>A-</td>
</tr>
<tr>
<td>87 - 89.9</td>
<td>B+</td>
<td>82 - 86.9</td>
<td>B</td>
<td>80 - 81.9</td>
<td>B-</td>
</tr>
<tr>
<td>77 - 79.9</td>
<td>C+</td>
<td>72 - 76.9</td>
<td>C</td>
<td>70 - 71.9</td>
<td>C-</td>
</tr>
<tr>
<td>67 - 69.9</td>
<td>D+</td>
<td>62 - 66.9</td>
<td>D</td>
<td>60 - 61.9</td>
<td>D-</td>
</tr>
<tr>
<td>&lt; 70</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Example** Assignments

- **Security**: develop a portion of a rootkit to hide the presence of malicious activity on a system
- **Concurrency**: implement a thread-safe system library (e.g. memory allocation)
- **Networking**: simulate an IP routing system
- **IPC**: build an application consisting of multiple processes that must communicate and coordinate via an OS-supported IPC mechanism
- **Databases**: create an SQL database and interact with it from a program
Academic Misconduct

- Academic Misconduct
  - Refer to Duke Community Standard
  - Homework is individual – you do your own work
  - Common examples of cheating:
    - Running out of time and using someone else's output
    - Borrowing code from someone who took course before
    - Using solutions found on the Web
    - Having a friend help you to debug your program

- **I will not tolerate any academic misconduct!**
  - Software for detecting cheating is very, very good ... and I use it
  - I’ve referred over a dozen cases to the Office of Student Conduct; don’t be one of them!

- “But I didn’t know that was cheating” is not a valid excuse
Academic Integrity for Assignments

• Your work is expected to be your own
• If you are unsure whether a certain course of action is permissible or not, please ask.
  • If you think that asking is a bad idea because I would probably say “no,” you can be fairly certain it is not permissible.
Our Responsibilities

• The instructor and TAs will...
  • Provide lectures at the stated times
  • Set clear policies on grading
  • Provide timely feedback on assignments
  • Be available out of class to provide reasonable assistance
  • Respond to comments or complaints about the instruction provided

• Students are expected to...
  • Receive lectures at the stated times
  • Turn in assignments on time
  • Seek out of class assistance in a timely manner if needed
  • Provide frank comments about the instruction or grading as soon as possible if there are issues
  • Assist each other within the bounds of academic integrity
The course website again

http://people.duke.edu/~tkb13/courses/ece650/
Let’s get started!