CSCI 4061
Introduction to Operating Systems

Instructor: Abhishek Chandra

Outline
- Course Overview
- Operating System Definition
- OS Evolution
- OS Structure

General Information
- Lectures: Tu/Th 9.45-11 am
- Course Web page: [http://www-users.cselabs.umn.edu/classes/Fall-2015/csci4061](http://www-users.cselabs.umn.edu/classes/Fall-2015/csci4061)
  - Also a Moodle page (for forum, HW submissions, etc.)
- Office Hours:
  - Location: KHKH 4-209, Time: Tu Th 11.30-12.30
- Contact: email preferred
- Help email: csci4061-help@cs.umn.edu
  - Goes to instructor and all TAs
  - Use this as far as possible – likely to get faster response

TA Information
- 4 TAs for the class
- Will conduct recitation sections and office hrs
  - Details on course website
  - You can go to any TA’s office hrs
Course Objectives

- Learn Operating System Concepts
- A "Programmer's view" of Operating Systems
- Focus on external OS interfaces and services
- Strong emphasis on systems programming
- Concepts applicable to non-Unix OS's
  - Processes, File Systems, Virtual Memory
  - Concurrency and Synchronization
  - Networking and Sockets

Course Objectives (contd.)

- At the end of the course, you should be able to:
  - Understand OS structure, interfaces, utilities
  - Write readable and efficient programs
  - Use several OS APIs, tools, and libraries
  - Understand performance tradeoffs in user programs and the OS
  - Build better programs and systems

Course Non-Objectives

- What you won't be taught in this course:
  - C programming: You'll have to pick it up yourself, though you would get some help in the initial labs
  - OS internals: This would be covered in 5103
  - Unix Tools: Some tools would be covered, but this is not a Unix tutorial course.

Course Non-Objectives (contd.)

- This course is NOT:
  - Kernel hacking course
  - User's guide to Unix (though you will get some exposure to Unix tools)
Pre-Requisites

- CSCI 2021 (Computer Architecture): Requires good understanding of computer organization and hardware concepts
- Familiarity with Unix environment
- Good programming skills: C would be used in the course, but familiarity with another language such as Java would be helpful too
- Good understanding of data structures and algorithm fundamentals

Course Work and Mechanics

Recitation Sections

- Must attend recitation in addition to lectures
- TAs would conduct the recitations
- Discussion of course material
- Hands-on exercises
- Clarifications on Assignments

Class Discussion Forum

- On Moodle class site
- You can post questions, clarifications, discuss ideas, course material
- Try responding to each other as far as possible
- Instructor, TAs will regularly monitor the forum
- However:
  - No irrelevant, abusive mails
  - No posts that break the rules/spirit of honesty
  - Don’t ask for solutions or post parts of your solution
Textbooks

Required:
- "Unix Systems Programming", 2nd Ed. by Robbins & Robbins

Optional:
- "Operating System Concepts", 9th Ed. by Silberschatz et al.
- 8th edition will also suffice

Readings and Lecture Notes

- Weekly readings from textbook/external sources on website
  - Must keep up to follow lectures/recitations
- Lecture Notes would be available online before lecture (most of the time)
  - Print your own copies if you need hard copies
- Additional reading material would be online: Links on the class web-page

Course Requirements

- Readings from the book and assigned lecture notes and additional material
- 4 Programming Assignments (40%)
  - To be done in teams of 2
  - Each assignment due in 2 weeks
- Exams (60%)
  - 2 Quizzes: In-class (5% each)
  - 2 Mid-term Exams: In-class (15% each)
  - Final Exam: 2 hrs (20%)

Programming Assignments

- You will be given a set of functional specifications
  - Implement a program to satisfy these specs
- Programs must be written in C
- Why C?
  - C provides a closer interface to the OS compared to many other languages (like Java, Python, etc.)
  - C allows more control over program state and performance (e.g., pointers, memory management)
  - C has traditionally been used for systems programming and for building OSes (e.g., Linux)
**Programming Assignments (2/3)**
- The programs should be well-documented
- Provide full code, header files, makefiles, test-files, README file
- Online submission by 11:59 pm on due date (via Moodle site)
  - One submission per team
- Late submission policy:
  - 10% penalty for <24 hrs late
  - No submission allowed beyond that

**Programming Assignments (3/3)**
- The code must be original
  - Not copied or derived from the Web, from past offerings, other students, programmer friend, ...
- No sharing of code across teams
  - Team members should work together
  - Ask questions and clarifications on class forum, from TAs or instructor
- Grading: Points for
  - Functionality and correctness
  - Code readability and documentation
  - Read specifications very carefully!

**Exams**
- Each quiz will be ~25 mins long: cover material from previous 3-4 weeks
- Mid-term exams would be the length of a lecture: cover material from previous 5-6 weeks
- Final exam would be comprehensive 2-hour exam
- Rules (unless specified otherwise):
  - Open notes/open book
  - No electronic devices allowed
  - No sharing of books or other material

**Grading Policy**
- Absolute scale (not grading on a curve)
  - [93-100] A
  - [90-93) A-
  - [87-90) B+
  - [83-87) B
  - [80-83) B-
  - [75-80) C+
  - [70-75) C
  - [65-70) C-
  - [60-65) D+
  - [50-60) D
  - [0-50) F
Regrading Policy

- Any issues with grading must be resolved within a week from getting back the graded material
  - Except Final exam where there will be a specific time/day for it
- For Prog. Assignments, contact the help email (or talk to the TAs)
- For exams, can talk to me

Expectations from you

- Attend lectures and recitations regularly
  - Very important for success in course
- Keep up with weekly readings, exercises
- Start on assignments early!
  - Not going to be trivial, one-day affairs
  - Will run into bugs, problems, questions
- Class etiquette
  - Be attentive, respectful to others
  - Be involved: ask and respond to questions, participate in discussions

Academic Dishonesty

- What does it include?
  - Copying assignments, cheating on exams, plagiarism
  - Programming assignments: Code should be original (not copied or derived from the web, other teams or external sources)
- Can result in serious consequences:
  - Can range from 0 on assignment to F in class or worse
  - U requires report to Office of Student Affairs
- Take this issue very seriously
  - All parties involved in cheating (helper and helpee) will be considered equally culpable
- If unsure, just ask!

Disability Statement

- If you have, or think you have, a disability, contact Disability Services
- Please get a letter from DS for any special accomodation request on course work
- I will try my best to make the required accomodations
What is an Operating System?

Consider an Example Scenario

- Suppose you have got a Math homework:
  - Multiply two large matrices
- You are also writing an essay for your English class
- Also, you want to surf the Web:
  - Read news
  - Watch a youtube video
  - Follow live scores of a football match

Computer System

<table>
<thead>
<tr>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
</tr>
<tr>
<td>Hardware (CPU, Memory, Disks, Devices)</td>
</tr>
</tbody>
</table>
What happens on your computer?

- Your matrix multiplication program is running
  - Reads/writes input and output files
  - Does number crunching (perhaps on multiple CPUs)
- You are editing a Word document
  - Typing characters from time-to-time
  - Saving file on the disk
- Your browser has multiple tabs open
  - Downloading and displaying Web pages, videos
  - Getting notifications of live score

What services does the OS provide?

- Allows different applications to execute concurrently
  - Processes, Memory management
- Allows access to multiple files, user input, display
  - File system, File I/O
- Allows parallelism and data sharing
  - Threads and synchronization
- Enables communication across machines
  - Networking and sockets

What is an Operating System?

User's View: Extended Machine
- Simple abstraction of hardware resources
  - CPU -> Processes, Threads
  - Memory -> Virtual Memory
  - Disks -> Files
  - Network interfaces -> Sockets

Goal: Simple, easy to use
- Less important: System performance
Programmer's View

- Resource Manager
- Efficient division of resources among multiple users, programs
  - Multiple processes on same CPU
  - Multiple files on the same disk
  - Multiple connections on same network link
- Arbitrate conflicting demands
- **Goal:** Maximum system performance

System View

- Control Program
- Handle different events, user inputs, etc.
  - User typing commands on keyboard
  - Bytes being read from the disk
  - Packets arriving on the network interface
- Multiple concurrent and asynchronous events
- **Goal:** Ensure correctness and fairness

Course Road Map

- Understand different OS components
  - Processes
  - File System
  - Memory Management
  - Threads
- Concurrency and Communication
  - Thread Synchronization
  - Networking and IPC
  - Signals