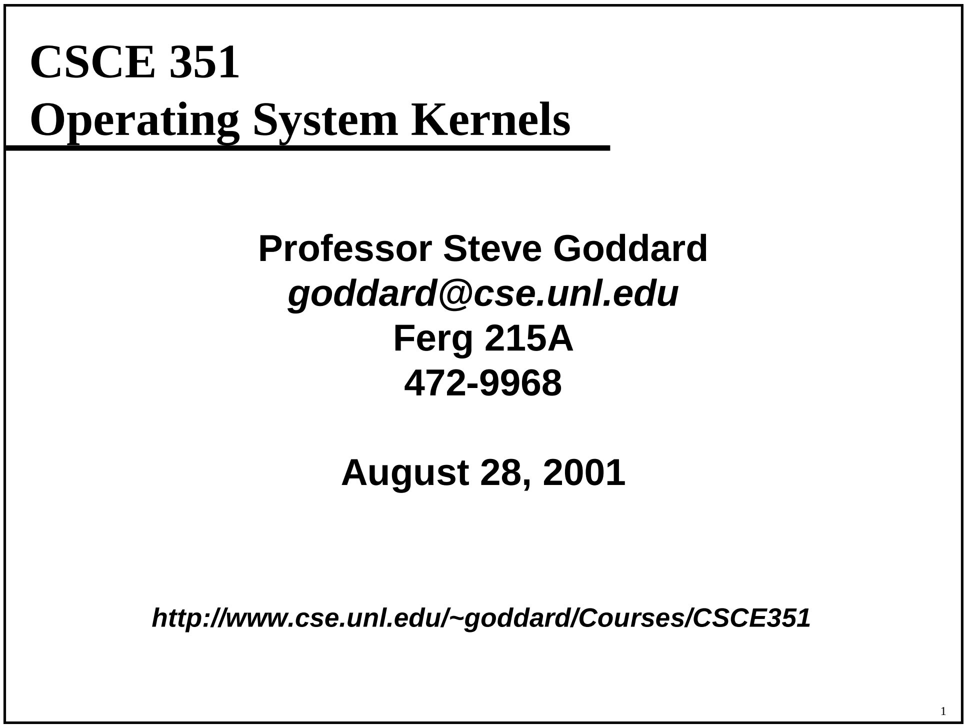


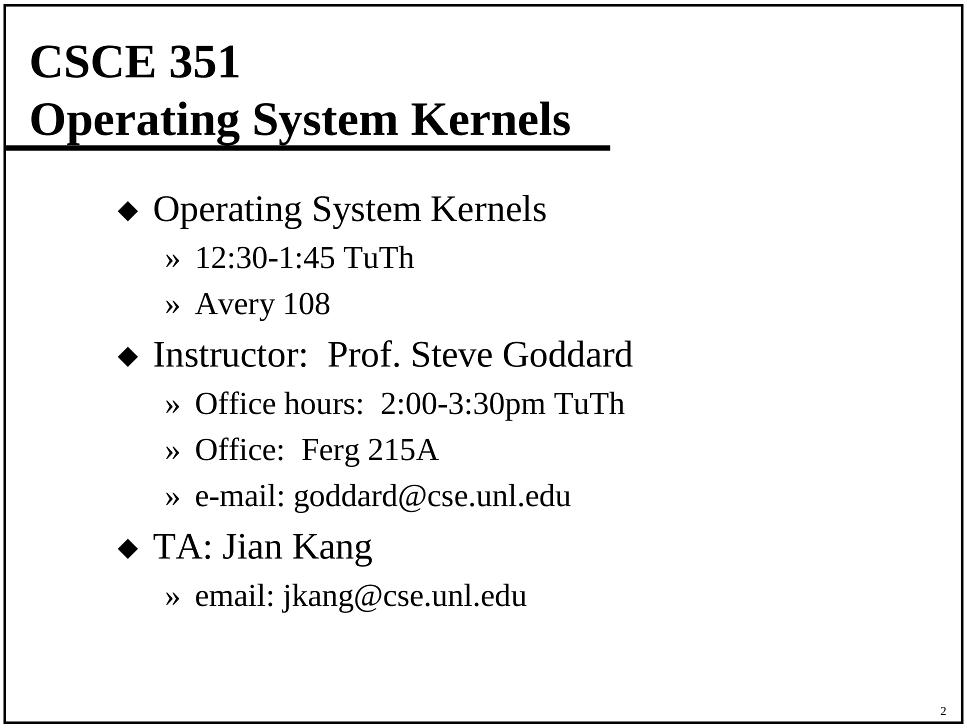
CSCE 351
Operating System Kernels

Professor Steve Goddard
goddard@cse.unl.edu
Ferg 215A
472-9968

August 28, 2001

http://www.cse.unl.edu/~goddard/Courses/CSCE351

1



CSCE 351
Operating System Kernels

- Operating System Kernels
  - » 12:30-1:45 TuTh
  - » Avery 108
- Instructor: Prof. Steve Goddard
  - » Office hours: 2:00-3:30pm TuTh
  - » Office: Ferg 215A
  - » e-mail: goddard@cse.unl.edu
- TA: Jian Kang
  - » email: jkang@cse.unl.edu

2

Textbook

- *Operating Systems: Design and Implementation*, 2nd Ed, by Tanenbaum and Woodhull, Prentice Hall, 1997
- Optional support book:
  - » *The C Programming Language*, ANSI C Edition, by Kernighan and Ritchie, Prentice Hall
  - » or any other suitable C language book.

Course Objectives

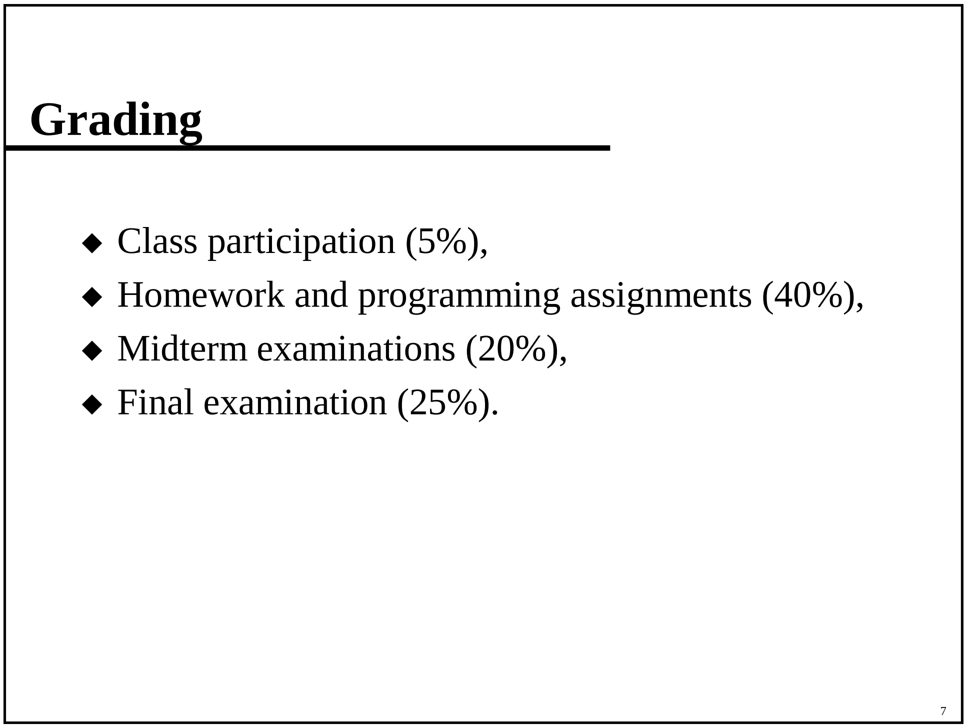
- **Mastery** & **practice** in: system initialization, process context switching, interrupt handlers, device drivers, and clock (timer) management.
- **Familiarity** with OS system calls, OS concepts, and OS structure.
- **Exposure** to processor scheduling, IPC, memory management, file system concepts and structure.
- **Practice** in critical thinking, identifying and evaluating system design tradeoffs, programming via a significant number of programming assignments.

Topics Covered

- Introduction to the organization and structure of operating systems.
- System calls (with hands-on experience using fork, exec, read, and write).
- Introduction to processes and threads.
- Race conditions and critical sections.
- Principles of I/O Hardware: I/O devices, device controllers, DMA.
- Principles of I/O Software: interrupt handlers, device drivers, device-independent I/O software, and user-space I/O software.
- Study of drivers for block devices that use DMA (e.g., drivers for ram disks and hard disks).
- Clock hardware and software including clock (timer) management.
- Terminals: hardware and software including keyboard and display drivers.
- Overview of memory management and file systems.

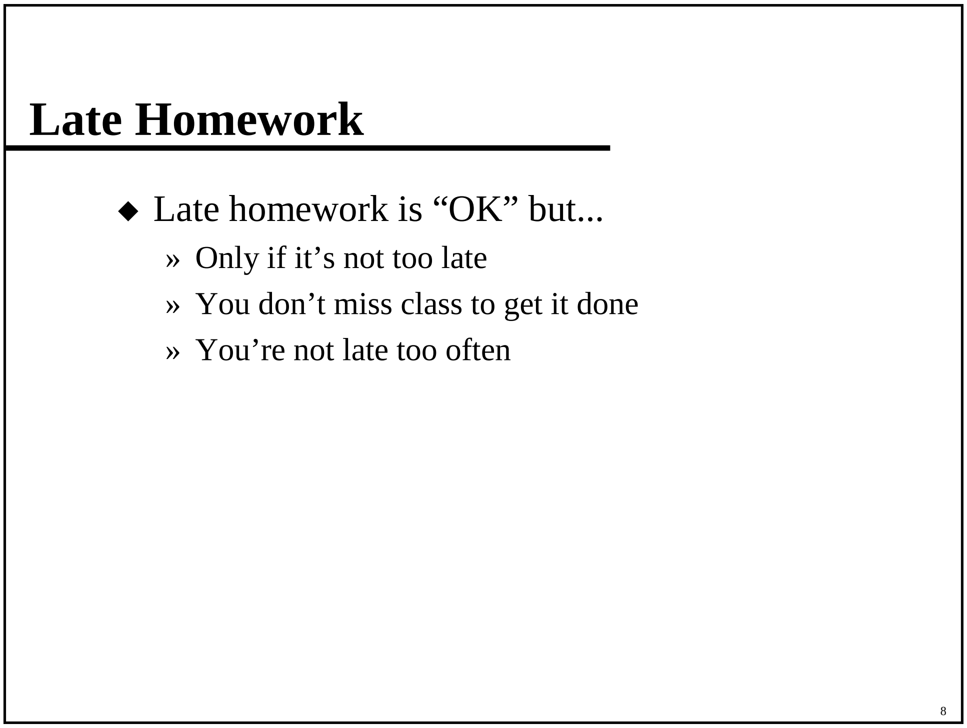
Prerequisites:
230, 230L or 231, and 310

- **Mastery** of computer programming, Boolean algebra, binary numbers, and powers-of-2.
- **Mastery** of stack, list, and queue data structures and algorithms.
- **Familiarity** with computer organization including I/O devices, and instruction set architecture.
- **Familiarity** with assembler language principles and context switching.



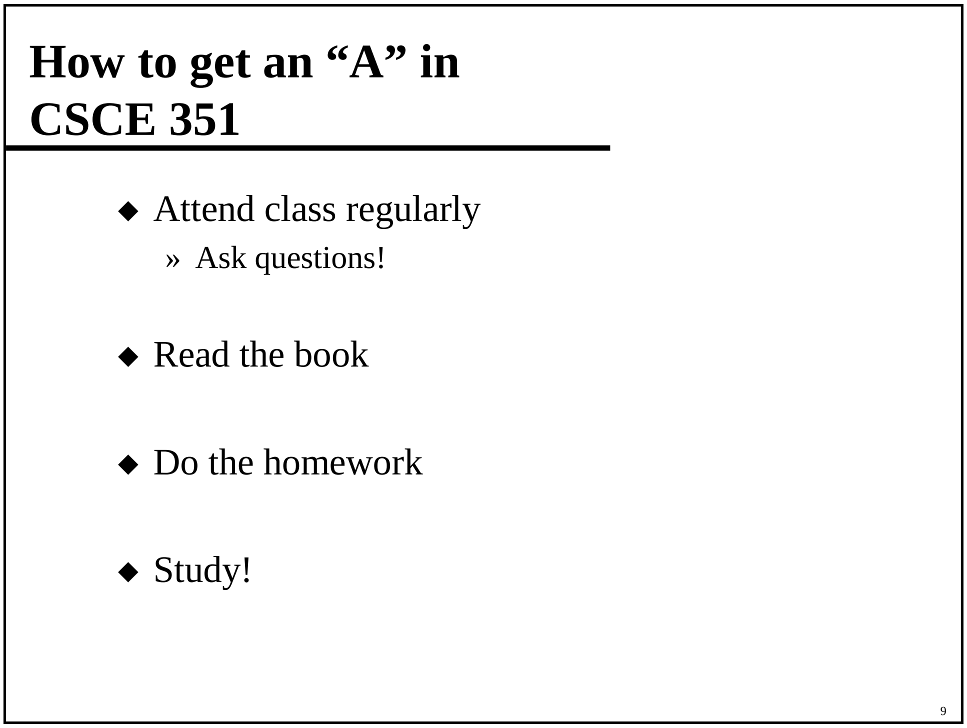
Grading

- Class participation (5%),
- Homework and programming assignments (40%),
- Midterm examinations (20%),
- Final examination (25%).



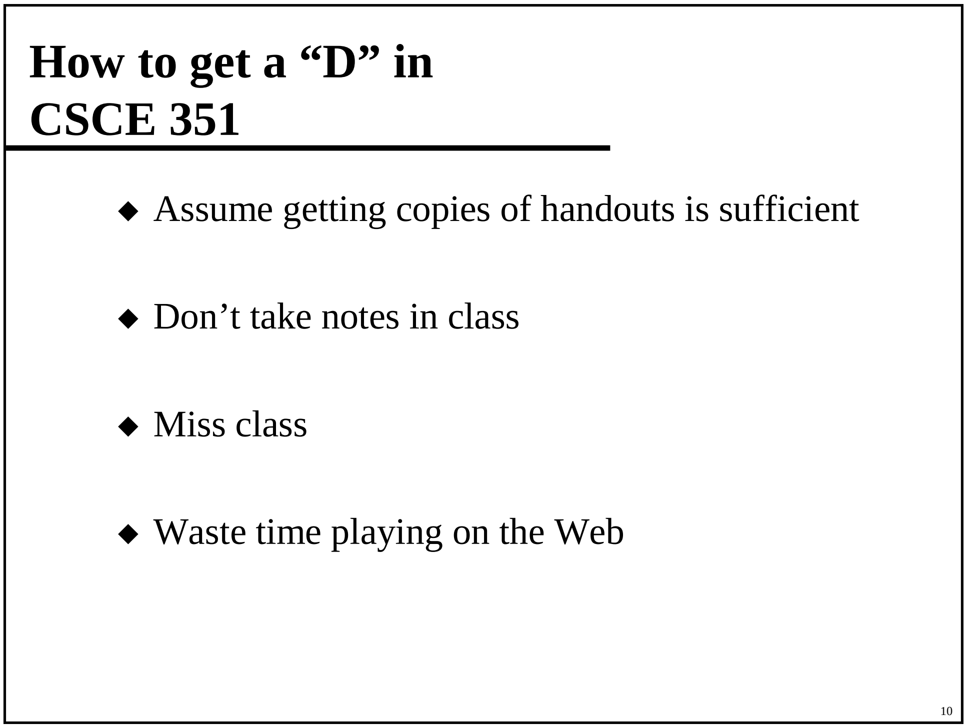
Late Homework

- Late homework is “OK” but...
  - Only if it’s not too late
  - You don’t miss class to get it done
  - You’re not late too often



How to get an “A” in CSCE 351

- Attend class regularly
  - » Ask questions!
- Read the book
- Do the homework
- Study!



How to get a “D” in CSCE 351

- Assume getting copies of handouts is sufficient
- Don’t take notes in class
- Miss class
- Waste time playing on the Web

Course Conduct

- You may work in groups in *understanding* assignments,
- developing *approaches* and *strategies*
- *learning* to use the UNIX/Minux tools
- You may not
  - » develop joint solutions
  - » share code
  - » copy anything
- All assignment solutions must be authored in full by you!

Summary

- We will study the design and implementation of operating system kernels.
  - » Use Tannenbaum’s book for concepts
  - » Use MINIX for concrete examples and hands on experience!
- There will be both written assignments and programming assignments.
- This course will be a lot of work.
- Hopefully, it will also be fun!