V22.0202-001: Computer Systems Organization II (Honors)
Spring 2000

Course Description and Guidelines

General Information

Instructor: Vijay Karamcheti (vijayk@cs.nyu.edu)
Teaching Assistant: Ilya Lipkind (lipkind@cs.nyu.edu)
Lectures: Mondays and Wednesdays, 2:00pm – 3:15pm
101 CIWW
Office Hours: Instructor: Mondays and Wednesdays, 3:15pm – 4:30pm
TA: Thursdays, TBD
Mailing List: To subscribe send email to majordomo@cs.nyu.edu.
In the body of the message, write: subscribe v22_0202_001_sp00
To post to the mailing list, send email to v22_0202_001_sp00@cs.nyu.edu
Web Page: http://www.cs.nyu.edu/courses/spring00/V22.0202-001/index.htm
All course materials, lecture slides, and announcements will be available here. You should plan on visiting this often, as we will rarely hand out printed materials in class.

Prerequisites

The only prerequisite for this course is V22.0201: Computer Systems Organization I.

Although not an official prerequisite, you will find the programming assignments easier to handle if you have some familiarity with C/C++ and UNIX program development tools (text editors, make, etc.). Note that since this is an Honors section, there are no recitations scheduled. Therefore, I will expect you to pick up requisite knowledge of the above tools in the first few weeks of class on your own. See the course resources page for several online tutorials on these topics.

Description

This course is an introductory undergraduate course in operating systems (OS), focusing on the core concepts of uniprocessor operating systems. The emphasis will be on understanding general concepts that are applicable to a wide range of operating systems, rather than a discussion of the features of any one specific system. Topics that will be covered include: protected kernels, processes and threads, concurrency and synchronization, memory management, virtual memory, and file systems. Time permitting, we shall also briefly discuss additional topics such as secondary storage, protection, and security.

A major component of the course is a series of programming projects using the Nachos instructional OS. These projects emphasize the implementation of the core concepts listed above in the context of a small, yet non-trivial OS. Starting from minimal baseline implementations, these projects will develop relatively sophisticated implementations of different OS modules such as support for thread synchronization, multiprogramming, and virtual memory.
A note about the Nachos projects: The programming projects in this course borrow heavily from the Nachos projects and accompanying resources developed for the CPS 110 course at Duke University by Jeff Chase (chase@cs.duke.edu) and his colleagues (Thomas Narten, Carla Ellis). They are being used here with permission.

Textbook(s)

The required textbook is:

**Operating System Concepts**, Fifth Edition  
Author(s): Silberschatz & Galvin  
Publisher: Addison Wesley

The course lectures will closely follow the organization of material in this text, focusing on Chapters 1-13, and 19-20.

The following text is also recommended:

**Modern Operating Systems**  
Author(s): Tannenbaum  
Publisher: Prentice Hall

Workload

- **Class:** You are expected to attend all of the lectures.
- **Readings:** You are responsible for completing suggested readings; this would help both in understanding the lecture material, and in conducting meaningful discussions.
- **Labs:** There will be six (6) required programming assignments using the Nachos OS, due approximately two weeks after they are handed out. You will work through these assignments in groups of 3-4 students and present your results in “demo sessions” to either the TA or me. Although a fair bit of work, these labs will be fun and valuable if you form your teams quickly, get an early start on the projects, and make your work in this course a priority this semester. The [Nachos Project Guide](#) contains additional details about the projects and policies.
- **Writeups:** There will be four (4) 2-page writeups that each of you will be required to hand in. In these writeups, you will explore broad OS issues that are currently relevant. For example, I might ask you to comment upon the open source software development philosophy and its implications on the eventual outcome of the Linux vs. Windows NT debate. The goal here is to have you go beyond the textbook and explore other sources of information.
- **Exams:** There will be two exams: a midterm and a final. The final will only include the material not covered in the midterm exam. You will be provided with two optional problem sets approximately a week before each exam; the exam questions will be derived from these problem sets.

This is a challenging course, requiring you to put in a minimum of 10-15 hours of effort every week. Since this is likely to be more difficult and time consuming than any of the courses you have taken so far, please budget your time accordingly. What you will get for your effort will be (1) knowledge of core uniprocessor OS concepts common to most current-day OSes, (2) hands-on experience implementing these concepts in a non-trivial OS, and (3) broad familiarity with a number of issues affecting wide-scale deployment of operating systems in the real world.

See the [course schedule](#) for tentative information about when the labs and writeups are due.
Grading

Final grades will be computed based upon the following weights: Nachos Labs: 50% (Labs 1-5: 8% each, Lab 6: 10%), Writeups: 10%, Midterm Exam: 20%, and Final Exam: 20%. See the Nachos Project Guide for additional information about how the programming assignments will be graded.

I will grade on the curve by looking for natural break points in the distribution of the overall scores. This means that your overall grade in the course is a function of how well or badly your peers are doing. For those students falling slightly below a cutoff, I will rely on subjective factors such as trends of improvement or decline, the nature of exam errors, and project group dynamics in deciding whether or not to assign the next higher grade.

The above being said, I think if you are worrying about grades, you are missing all the fun! You should focus on working hard in this course and learning the material. I have every expectation that all of you will do very well in this course.

Policy on Collaboration for V22.0202-001

You are encouraged to collaborate with others in the class on all aspects of the course except the exams. The only requirement is that what you hand in should represent your own effort (or your group’s, in the case of the labs). To make this more concrete, you can discuss high-level approaches that you might take to solve a particular lab problem, but the actual coding and evaluation of this approach should be your own. For the lab assignments, I will expect every member of the group to be familiar with the approaches taken for each project, whether or not they were the ones who wrote the code. The class will be more productive and fun for everybody if you try and help each other out, while taking responsibility for your share of the work.