CMSC 321 – Operating Systems, Spring 2003

Professor
Lewis Barnett          Jepson Hall 202
email: lbarnett@richmond.edu (804) 289-8091
Office hours: Monday – Thursday, 2:30 p.m. – 3:30 p.m., or by appointment

Course Description
By itself, computer hardware is a lump of so much metal, plastic, and silicon. An Operating System is a set of programs that turn this “bare” hardware into a useful computer system. The Operating System’s primary goal is providing and managing shared access to the resources of the system. These resources include the CPU, memory, disks, terminals or displays, printers, and other peripherals. Operating Systems range in complexity from simple systems for single-user personal computers through large multi-user time sharing systems to distributed systems managing cooperation among many connected computers. One of the important tasks of an operating system is creating various levels of abstraction to hide the details of the structure and operation of the various hardware components that make up the computer, thus making the job of the applications programmer simpler.

The design of an Operating System is an exercise in compromise rivaling the most delicate diplomatic negotiation. For any parameter of system performance that you seek to optimize in a design, there is a penalty to be paid in some other aspect of system performance. We will consider many of the tradeoffs involved in OS design and investigate many alternative methods of dealing with them. The course is arranged as a survey of these general design problems, but will also include case studies of how real systems have solved them.

Prerequisites
CMSC 222, CMSC 301

Readings
Textbook: Operating System Concepts, 6e, Abraham Silberschatz, Peter Galvin, and Greg Gagne, Addison-Wesley, 2002. A-W has recently released an “updated” version of the sixth edition of the text with additional information about Windows XP, and this is the version you’ll find in the bookstore. Don’t worry if you bought a used copy of the earlier, un-updated sixth edition; it has everything you’ll need.

In addition, the following books (among others) are available at the Science Library for further reference:

   Unix Internals: the new frontiers, Vahalia

   Design and Implementation of the 4.3BSD Unix Operating System, Leffler, McKusick, Karels, and Quarterman.

   Design of the Unix Operating System, Bach. (Describes System V.)

Readings from the original literature will be assigned when appropriate.
Web links

http://www.mathcs.richmond.edu/~lbarnett/cs321/

This URL gives access to general course information, including links to assignments, old tests, example code, and links to other web sites of interest. I’ll call this the “class web page.”

Grading Policy

Homework/Projects  35%
Tests (2) 40%
Final Exam 25%

Special Information Concerning Class Meetings

The class will meet Wednesday and Friday from 1:15 pm to 2:30 pm. In addition, there will be consulting lab sessions one or two times in most weeks. The times for these lab sessions will be determined during the first week of classes.

This is a 4-credit class. Hence the additional time in consulting lab. These sessions are intended to allow you to work on the programming assignments and homework assignments for the course in a situation where I am immediately available to answer questions. Both the programming assignments and the homeworks are substantial in this course, so you will want to take advantage of these opportunities.

Assignments

There will be two types of outside assignments this semester, homework problem sets and programming projects. Homework must be done in your own words. You may ask others for clarification of questions, but the solutions/answers must be your own.

The level of acceptable cooperation on the projects will be specified in the project assignments. Unless cooperation is explicitly allowed, I will expect you to work individually on the projects. In such circumstances, it is acceptable to ask classmates about the meaning of syntax errors, and about the operation of the computer systems or software development software you are using. What is not acceptable is looking at another student’s program or using code written by someone else unless it is code that I have provided to everyone. If you are having trouble with your program’s logic, I will be happy to look at it and answer questions.

The projects will primarily be exercises in using the systems calls in the Standard C Library which give access to the Unix operating system. We will be using the C++ programming language on the department’s Unix systems. If you have not used C++ or Unix before, please see me after the first class meeting.

I expect assignments to be turned in on time. Assignments not received by the stated due date will receive a grade of 0. I do award partial credit for partial solutions.

Honor System and Collaboration Policy

I assume that each of you is honorable and will abide by the Honor Code. Tests and exams must be completed without help from others. Projects and homework can be done in accordance with the following collaboration rules:

- Programming projects are to be completed individually. You may discuss compiler errors and issues relating to the computing environment with your classmates. It is
acceptable to discuss the assignment with classmates, but not to use code written by another student in your solution to a problem. I will be happy to discuss problems in the logic of your program with you.

- Homework must be done in your own words. You may ask others for clarification of questions, but the solutions/answers must be your own.

**Attendance Policy**

Each unexcused absence will result in a 1 point deduction from your final average. Excuses need not be written. When possible, please inform me of absences beforehand. For students who have five or more unexcused absences, I reserve the right to submit a grade of V, failure due to excessive absences.

**Course Outline**

In the following list of topics, the numbers in parentheses indicate sections in the text book which cover the topic.

- Introduction to Operating Systems (Chapters 1 through 3)
- Introduction to Processes (4.1 - 4.3)
- Interprocess Communication (4.4 - 4.7)
- Process scheduling (6)
  - Concurrency and synchronization (7)
- Deadlock (8)
- Memory Management, Swapping (9)
- Virtual Memory (10)
  - Memory Page Replacement Algorithms (10.4 – 10.9)
- File System Overview (11)
- File System Implementation (12)
- Input/Output Hardware and Software (13)
- Disk systems (14)
- Protection and security (18, 19)
- Case studies (20, 21)

**Test dates**

Test 1: Friday, Feb 14
Test 2: Wednesday, April 2
Final exam: Wednesday, April 23, 2:00 pm – 5:00 pm