## COURSE GOAL

To educate students on the basic concepts and components of operating systems, the relevant characteristics of hardware, and the tradeoffs between conflicting objectives faced by the OS.

## OBJECTIVES

As much as operating systems are an essential part of computer systems, a course on operating systems is an essential part of any computer science or computer engineering program. This course is oriented towards exposing students to the essential concepts and issues that underly operating systems and their design.

### Technical

Make students understand the key concepts and mechanisms of modern operating systems:

- processes and process management,
- memory management techniques aimed at optimising system utilisation and responsiveness,
- on-line storage methods (file systems),
- security and protection,
- concurrency issues,
- some of the issues arising from distribution.

### Educational

Make students understand the reasons why operating systems are built the way they are, and what the implications and lessons are for other software systems. Specific learning objectives are:

- appreciation of design trade-offs and design decisions and their dependence on the target environment;
- appreciation of the distinction between mechanisms and policies, and why this is important;
- exposure to low-level code;
- exposure to current trends in operating systems research and development.

### Professional

The tutorial formats will give students practice in the presentation of solutions to an audience of peers, and will challenge them to critique peer technical presentations. Furthermore, the whole course encourages critical examination and analysis of "standard" solutions.

## PREREQUISITES

### General

It is assumed that the student is familiar with the general organisation of a general-purpose computer (in particular, CPU, memory, bus, registers, machine instructions, interrupts/exceptions).

Students are also assumed to be competent in the C programming language.

### Course Prerequisites

DEFINITION OF COURSE_CODE BROKEN!

## Constituents

### Lectures

Lectures will cover roughly the following topics (although not in this order):

- **Introduction and Overview**
  Introduction to operating systems, computer systems structures, operating system structures
- **Threads**
  execution abstraction, state, context, preemption, scheduling, synchronisation, deadlocks
- **Process Management**
  resources, address spaces, multi-threading
- **Storage Management**
  memory management, virtual memory, file systems
- **Protection and Security**
  goals of protection, access matrix and its implementations, capabilities, authentication, encryption, design for security, security awareness

- **Case Studies**
  drawn from UNIX, Linux, Windows-2000 and OS/161 will be used throughout the course.

PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 166 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 170

**Tutorials**

Tutorials start in Week Two. Part of the allocated tutorial time will be devoted towards solving problems and discussing solutions related to lecture topics, while the remaining time will be used to discuss topics relevant to assignments.

Students will be required to present tutorial problems (approx. 10 min/student) under the guidance of the tutor. Written submissions are required for some portion of the tutorials. Individual student contribution (presentation, critiquing solutions and suggesting alternatives) in tutorials will be assessed and forms part of the final mark for the course. To obtain the tutorial marks, you have to attend your registered tute. Other tutors cannot award participation marks.

*Note that tutorial marks will be based on participation only, there are no marks allocated just for attending.*

**Assignments**

There will be four assignments, due approximately in weeks 3, 6, 9 and 12. Assignments will use the OS/161 educational operating system running on a simulated MIPS R3000 computer called System/161. Both the operating system and the simulator were developed at by the Systems Research at Harvard group. The simulator is relatively platform independent, it runs on Unix as well as Windows systems. Assignment solutions will be submitted in C code. Details will be released in due course.

Assignments will generally have *advanced* options which provide a significantly higher level of challenge than the standard assignments (which most students find fairly challenging already). There will be bonus marks available for students solving the advanced parts. However, we want students to do the advanced assignments for the challenge rather than the marks, therefore the number of bonus marks is intentionally kept small compared to the amount of extra work required.

In order to minimise cheating we do not re-run unmodified assignments from earlier years. This implies that there are new assignment specifications. Even though we will not release the specifications until we have a prototype solution, this does not guarantee that there are no problems with the specification. In order to minimise the impact of specification problems to average students we will offer 5% bonus marks to students solving the assignment within 48 hours of release. Such submissions will have to be demonstrated and discussed with a tutor. This procedure should then help to iron out all problems with the spec before most students start on the assignment. These bonus marks can be used to make up lost marks in the same or other assignments (and are in addition to the early submission bonus below).

Students have, unfortunately, a tendency to underestimate the time required to do the assignments. Experience from the last few years show that the assignments in this course are considered challenging, and they consume a fair amount of time. It is important to start early. If you start on them on the weekend before the deadline you'll almost certainly miss the deadline.

In order to encourage students to start early, we are also offering 10% bonus marks for submitting *their final solution one week before the deadline* (i.e., you lose the bonus marks if you re-submit less than a week before the deadline). Such bonus marks will allow you to make up for missed marks in the same assignments (i.e., you cannot get above the full marks for the assignment that way).

Penalty for late submission of assignments will be 4% (of the worth of the assignment) subtracted from the raw mark per day of being late. In other words, earned marks will be lost. For example, assume an assignment worth 25 marks is marked as 20, but had been submitted two days late. *The late penalty will be 2 marks, resulting in a mark of 18 being awarded.* **No assignments will be accepted later than one week after the deadline.**

**Cheating**

We *always* check for cheating, and while we will never catch all cheaters, we *always* catch a significant number. We deal with them severely. Don't take chances!

**Penalties for cheating:**

- Copying of code: 0 FL for the course

  This includes submitting another students solution to the assignment in full or part and with or without modifications as one's own solution. This includes code that was derived from pseudo code or a detailed algorithmic description.

- Help with coding: negative marks (ie, minus half the max mark)

  Submission of code that was developed with the help of another student or a third party. This includes receiving pseudo code or discussing details of the solution, or having a third party debug your code.

- Originator of plagiarised solution: 0 marks (for this assignment)

  Showing one's own solution to another student or giving a copy to another student, or providing help in the form of pseudo code or detailed algorithmic descriptions. (This does **not** apply if there is sufficient evidence that the copying happened without knowledge of the originator.)

- Team work: 0 marks (for this assignment)

  Submission of code that was developed fully or in part with another student. This includes joint development of pseudo code or a detailed algorithmic description. (It does not matter if all parties can explain the code afterwards. The code must not be jointly developed.)

People who copy from previous year's students will be treated just as harshly as those copying from students doing the course with them. In addition, they will drag the suppliers of the solutions into the misconduct case. The general principle is that anything that is someone else's work must **not** be used without giving clear and unambiguous credit to the originator. Failing to do so constitutes cheating.

Cases of repeated cheating (in this or other courses) will be treated as academic misconduct and will result in an application being made to have the offender(s) suspended from the University. Suspension applications will also be made for first offenders in serious cases, e.g., where people involved in cheating cases are found to have made false statements.

**This is no idle threat**. We catch and penalise cheaters to protect the reputation and integrity of the University, and the interests of those who get their marks through honest work.

**COMP3231 Cheating Statistics**

| **Session** | **1998/S1** | **1999/S1** | **2000/S1** | **2001/S1** | **2001/S2** | **2002/S1** | **2002/S2** | **2003/S1** | **2003/S2** |
|---|---|---|---|---|---|---|---|---|---|
| **enrolment** | 178 | 410 | 320 | 300 | 107 | 298 | 156 | 290 | 122 |
| **suspected cheaters** | 10(6%) | 26(6%) | 22(7%) | 26(9%) | 20(19%) | 15(5%) | ???(?%) | 13(4.5%) | ??(??%) |
| **full penalties** | 2* | 6* | 9* | 14* | 10 | 9 | 5 | 2 | 1 |
| **reduced penalties** | 7 | 15 | 7 | 7 | 5 | 4 | 2 | 2 | 9 |
| **cheaters failed** | 4 | 10 | 16 | 16 | 10 | 12 | 5 | 4 | 1 |
| **cheaters suspended** | 0 | 0 | **1** | 0 | 0 | **1** | 0 | 0 | 0 |

* Note: full penalty was not 0FL prior to S1/2002.
+Application pending

While we know that not all cheaters get caught, almost all people caught copying end up failing. Don't take the risk, don't become part of the statistics!

#### Mid-Session Quiz

There will be **no** mid-session quiz.

#### Final Exam

A two hour written final examination will be conducted.

#### Supplementary assessments

Supplementary exams will only be awarded in well justified cases, in accordance with School policy, **not** as a second chance for poorly performing students. In particular, it is unlikely that a supplementary will be awarded to students who have actually sat the proper exam. **Make up your mind whether or not you are sick before attempting the exam!**

Supplementary exams will be **oral**. The supplementary final exam will be held after the written supplementary exams held for other courses.

## Assessment

Tutorial participation PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 424 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 467 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 471 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/04s1/intro-content.php on line 557 and assignments will determine the ``class mark'' component towards the final assessment. (Note that the class mark is capped at 100, irrespective of how many bonus marks you have accumulated.) The exam will be the second component of the final assessment. The harmonic mean of the ``class mark'', $C$, and the final exam mark, $E$, is used to determine the final mark, $M$, according to

$$M = \frac{2\,E\,C}{E + C}$$

if both ***E >= 40*** and ***C >= 40***. Otherwise, the final mark is

$$M = \text{minimum of } 44 \text{ and } \frac{2\,E\,C}{E + C}$$

FIXME

A final mark of 50% is required in order to pass, however, it is furthermore required for a pass that a minimum of 40% is achieved in each of the ***C*** and ***E*** components.

## TEXT and REFERENCE BOOKS

### Textbook

- A. Tannenbaum, *Modern Operating Systems*, 2nd ed., Prentice Hall, 2001

### Reference Books

For Operating Systems:

- A. Silberschatz and P.B. Galvin, *Operating System Concepts*, 6th ed., Addison Wesley (2001)
- William Stallings, *Operating Systems: Internals and Design Principles,* 4th edition, 2001, Prentice Hall.
- A. Tannenbaum, A. Woodhull, *Operating Systems--Design and Implementation*, Prentice Hall, ???
- John O'Gorman, *Operating Systems*, MacMillan, 2000
- Uresh Vahalla, *UNIX Internals: The New Frontiers*, Prentice Hall, 1996

For the C language:

- B Kernighan and D. Ritchie, *The C Programming Language*, 2nd ed, Prentice Hall.
- S. Harbison and G. Steele, *C: A Reference Manual*, Prentice Hall.

Copies of lecture slides and other information can be found under the course's WWW home page at URL http://www.cse.unsw.edu.au/~cs3231/.

## STAFFING

| Name | Function | Consultation | | |
|---|---|---|---|---|
| | | Time | Room | Phone |
| Kevin Elphinstone | Lecturer in Charge | Tue 10:00-11:00 | TBA | 7160 |
| Cameron Stone | Administrator | TBA | TBA | 6533 |

Please direct COURSE_CODE-related questions to the class account, not the individual academics or tutors. We will endeavour to provide reasonably quick response for questions mailed to cs3231, and reserve the right to ignore questions mailed to individuals. Furthermore, in order to be able to maximise the support for relevant questions we need to minimise the time wasted on dealing with repeated questions. Therefore, questions which have already been answered in a relevant FAQ will be ignored.