Course Staff

Name	Function
Kevin Elphinstone	Lecturer in Charge

Note: Consultation times are available on the the course web page.

Course Information

Units of Credit: 6

Course Delivery

Note that COURSE_CODE are taught in parallel. The lectures are common for all courses. We cater for the differing needs of postgraduate and undergraduate students by weighting assessment components more flexibly in the case of postgraduate students. See the section on assessment below for details.

Course Aims

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.

In general terms, the course aims to educate students in the basic concepts and components of operating systems, the relevant characteristics of hardware, and the tradeoffs between conflicting objectives faced by the operating systems in efficiently supporting a wide range of applications.

This course provides an understanding of the underlying operating systems which students have implicitly relied upon in developing applications in foundational courses within Computer Science and Engineering, and will rely on in their future careers when developing systems and applications. The course also builds upon COMP2121/9032 *Microprocessors and Interfacing* by illustrating how more complex software systems are supported on microprocessor-based systems.

Specific Aims and Outcomes

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 (optional),
- concurrency issues,

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.

The assignments give students an opportunity to develop skills required to work as a team on a technical project, and the opportunity to work with a substantial body of code created by a third-party.

Teaching Approach

Our approach to teaching operating systems is

- introducing operating system theory in the lecture component of the course, including case studies of real systems
- and active engaging students in the practical application of that theory through challenging assignments, including assessment and feedback on their efforts.

- We also seek engagement of students by encouraging dialogue between staff and students both in lectures and tutorials to re-enforce concepts being taught.
- Students also are encouraged to learn collaboratively through group assignments, which also hones their skills as "team players" for future entry into the works force.

Course Development

We strive to continually improve this course, and actively seek feedback both during session, and via various evaluation surveys (CATEI and my own) at the end of the course. The results of surveys and any feedback are taken seriously and are actively used as input to further improvements in the course. See the survery archive (http://cgi.cse.unsw.edu.au/~cs3231/surveys.html) for what students thought about the course, and how the course is evolving based on their feedback.

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. More specifically, students should understand pointers, function pointers, memory allocation (`malloc()`), and be comfortable navigating around an existing code base.

Course Prerequisites

> PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 162 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 182 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 188 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 195 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 260 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 273 PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 282

Course Exclusions

1. COMP9201/9283 Operating Systems (for COMP3231/3891)
2. COMP3231/3891 Operating Systems (for COMP9201/9283)

Constituents

Broadly, the course consists of lectures, tutorials, assignments, and a final exam.

Lectures

The lecture delivery (3 hours per week) and material covered is common for all operating system students, independent of the course code your enrolled under.

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
  - *Note: I'm trying to get this topic back into the course this semester, but it depends on how succesful I am in compressing other material*
- **Case Studies**
  - drawn from UNIX, Linux, Windows and OS/161 will be used throughout the course.

A detailed list of lecture topics and the slides used for the lectures will be posted on the course website as session progresses.

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 under the guidance of the tutor. Individual student participation (presentation, critiquing solutions and suggesting alternatives) in tutorials will be assessed and forms part of the final mark for the course (COMP3231 only). To obtain the tutorial marks, you have to attend your registered tutorial. 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 4,6,9, and 13. Note that the first assignment is a really trivial familiarisation exercise, and should not be used to judge the difficulty of the other assignments. Except for the first familiarisation assignment, the three main assignments are done in groups of two. 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.

The last 2 assignments have *advanced* components which provide a significantly higher level of challenge than the standard assignments (which most students find fairly challenging already). We want students to do the advanced assignments for the challenge rather than the marks, therefore the opportunity to earn marks towards you assessment is intentionally kept small compared to the amount of extra work required. The marks are limited to a max of 10% of the class mark component of the course, see the Assessment section below for details.

PHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in /tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 329

Students have, unfortunately, a tendency to underestimate the time required to do the assignments. Experience from the last few years shows 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. As a tip, the majority of the assignment is learning how to solve the problem (approx. 75% of the effort). The programming component is actually the minority of the work required (approx 25%).

In order to encourage students to start early, we are also offering 10% bonus marks (10% of the raw mark awarded) 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 assignment (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.** The late penalty is purposely lenient due to the tendency of students to underestimate the work required.

- Assignments will be marked as a group, i.e. groups of one will be marked the same as groups of two or three.
- All group members are expect to contribute equally to EACH assignment.. In submitting your assignment, you are agreeing you have contributed equally unless we recieve an agreed-to statement from all group members identifying differing levels of contribution.

Plagiarism

What is Plagiarism?

Plagiarism is the presentation of the thoughts or work of another as one's own.

Examples include:

- direct duplication of the thoughts or work of another, including by copying material, ideas or concepts from a book, article, report or other written document (whether published or unpublished), composition, artwork, design, drawing, circuitry, computer program or software, web site, Internet, otherelectronic resource, or another person's assignment without appropriate acknowledgement;
- paraphrasing another person's work with very minor changes keeping the meaning, form and/or progression of ideas of the original;
- piecing together sections of the work of others into a new whole;
- presenting an assessment item as independent work when it has been produced in whole or part in collusion with other people, for example, another student or a tutor; and
- claiming credit for a proportion a work contributed to a group assessment item that is greater than that actually contributed.

For the purposes of this policy, submitting an assessment item that has already been submitted for academic credit elsewhere may be considered plagiarism.

Knowingly permitting your work to be copied by another student may also be considered to be plagiarism.

Note that an assessment item produced in oral, not written, form, or involving live presentation, may similarly contain plagiarised material.

The inclusion of the thoughts or work of another with attribution appropriate to the academic discipline does not amount to plagiarism.

The Learning Centre website is main repository for resources for staff and students on plagiarism and academic honesty. These resources can be located via:

www.lc.unsw.edu.au/plagiarism

The Learning Centre also provides substantial educational written materials, workshops, and tutorials to aid students, for example, in:

- correct referencing practices;
- paraphrasing, summarising, essay writing, and time management;
- appropriate use of, and attribution for, a range of materials including text, images, formulae and concepts.

Individual assistance is available on request from The Learning Centre.

Students are also reminded that careful time management is an important part of study and one of the identified causes of plagiarism is poor time management. Students should allow sufficient time for research, drafting, and the proper referencing of sources in preparing all assessment items.

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

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. A sample from past years is included below.

COMP3231 Cheating Statistics

Session	**1998/S1**	**1999/S1**	**2000/S1**	**2001/S1**	**2001/S2**	**2002/S1**	**2002/S2**	**2003/S1**	**2003/S2**	**2013/S1**
enrolment	178	410	320	300	107	298	156	333	133	108
suspected cheaters	10(6%)	26(6%)	22(7%)	26(9%)	20(19%)	15(5%)	???(?%)	13(4.5%)	??(??%)	18(16%)
full penalties	2*	6*	9*	14*	10	9	5	2	1	0
reduced penalties	7	15	7	7	5	4	2	2	9	16
cheaters failed	4	10	16	16	10	12	5	4	?	1
cheaters suspended	0	0	**1**	0	0	**1**	0	0	0	0

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

Mid-Session Quiz

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

Final Exam

A two hour written final examination will be conducted. Note: Extended and normal OS have different exams with approximately 75% common content.

Supplementary assessments

Supplementary exams will only be awarded in well justified cases, in accordance with School policy for *Special Consideration*, **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 participationPHP Notice: Use of undefined constant COURSE_CODE - assumed 'COURSE_CODE' in

/tmp_amd/adams/export/adams/1/cs3231/public_html/14s1/intro-content.php on line 607 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 weighted harmonic mean of the ``class mark'', $\boldsymbol{C}$, and the final exam mark, $\boldsymbol{E}$, is used to determine the final mark, $\boldsymbol{M}$, according to

- For COMP3231/3891:

  The harmonic mean is weighted 50/50.

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

- For COMP9201/9283:

  The final mark will be $\boldsymbol{M} = \max(\boldsymbol{M1}, \boldsymbol{M2})$, where $\boldsymbol{M1}$ is a 50/50 weighted harmonic mean as above, and $\boldsymbol{M2}$ is a 20/80 weighted harmonic mean (20 class, 80 exam) as shown below.

$$\boldsymbol{M2} = \frac{5\,\boldsymbol{E}\,\boldsymbol{C}}{\boldsymbol{E} + 4\boldsymbol{C}}$$

  This method of weighting is to cater for postgraduates working full time who are unable to invest significantly in assignment work (hence 20/80 weighting), while not penalising postgraduates who do commit to the assignment component of the course (50/50 weighting).

For COMP3231 students the class mark is determined as the sum of tutorial participation plus any advanced assignment bonuses (max 10%) and the base assignments (max 90%).

For 3891/9283 students the class mark is determined as the sum of the assignment advanced assignments (max 10%) and the base assignments (max 90%).

COMP9201 students who wish to participate in tutorials can arrange with their tutor *at the start of semester* to recieve a participation mark, in which case, the class component of their mark will be calculated as for COMP3231. Otherwise, the class mark will be determined solely from the base assignments scaled to 100%.

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 the exam $\boldsymbol{E}$ component.

We reserve the right to scale mark components of the individual courses (COMP3231/9201/3891/9283) separately where appropriate.

More information on harmonic means in general is available here.

TEXT and REFERENCE BOOKS

Textbook

- A. Tannenbaum, *Modern Operating Systems*, 3rd ed., Prentice Hall, 2008

Reference Books

For Operating Systems:

- A. Silberschatz, P.B. Galvin and Greg Gagne, *Operating System Concepts*, 7th ed. (2004)
- William Stallings, *Operating Systems: Internals and Design Principles,* 5th edition, 2005, Prentice Hall.
- A. Tannenbaum, A. Woodhull, *Operating Systems--Design and Implementation*,3rd ed., Prentice Hall, 2006
- 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/.