Course Outline

Contents

Course Details

Course Code	COMP3231 COMP9201 COMP3891 COMP9283
Course Title	Operating Systems
Convenor	Kevin Elphinstone (/users/z8990549)
Admin	Kevin Elphinstone (/users/z8990549)
Classes	**Lectures** : Video only, no face-to-face lectures.
Consultations	http://cgi.cse.unsw.edu.au/~cs3231/consultations.php (http://cgi.cse.unsw.edu.au/~cs3231/consultations.php)
Units of Credit	6
Course Website	http://cse.unsw.edu.au/~cs3231/ (http://cse.unsw.edu.au/~cs3231/20T2/)
Handbook Entry	http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP3231.html (http://www.handbook.unsw.edu.au/undergraduate/courses/current/COMP3231.html)

Note: The 21T1 term is a **web stream only** version of the Operating System's course. The main implication of it being a web stream is that there are no face-to-face lectures, even if UNSW returns to campus. Lecture material will be available either via pre-recorded videos or recorded live online lectures.

Tutorials will run online via blackboard collaborate (available through moodle).

Course Summary

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 exposes students to the essential concepts and issues that underlie 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 trade-offs between conflicting objectives faced by the operating systems in efficiently supporting a wide range of applications. Students will apply some of the concepts learnt by implementing parts of a realistic teaching operating system.

This course provides an understanding of the underlying operating systems that students have implicitly relied upon when developing applications in the foundational courses within Computer Science and Engineering. The knowledge gained will continue to be relevant in future careers when developing systems and applications.

Assumed Knowledge

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

Students are assumed to be competent using the C programming language. More specifically, students should understand pointers, function pointers, memory allocation (`malloc()`), and be comfortable navigating around an existing code base. They should be confident implementing data structures and algorithms, and able to debug their implementation within an existing code base.

Students are assumed to be familiar with the `git` revision control system or capable of learning the basics quickly.

The above knowledge is assumed to have been acquired in the following courses...

- For COMP3231:
  - COMP2521 or COMP1927 These are the foundational data structure and algorithms courses. You should be able to implement standard data structures and algorithms such as trees, hashing, heaps, sorting, etc...
  - COMP2121 or ELEC2142 or COMP1521 or DPST1092 These are microprocessor programming courses or systems programming courses. We expect you to be comfortable with low-level microprocessor-based hardware, registers, interrupts, memory, devices, and programming in assembler.
- COMP9201:
  - COMP9024 Data Structures and Algorithms
  - COMP9032 Microprocessors and Interfacing

Student Learning Outcomes

After completing this course, students will:

1. Understand the key concepts and mechanisms of modern operating systems:
   - processes and process management, including threads and concurrency management,
   - physical and virtual memory management techniques ,
   - on-line storage methods (file systems)
2. Able to apply their knowledge to understand/diagnose the behaviour of an operating system
3. Able to implement or extend the functionality of an operating system.

This course contributes to the development of the following graduate capabilities:

Graduate Capability	Acquired in

Scholars capable of independent and collaborative enquiry, rigorous in their analysis, critique and reflection, and able to innovate by applying their knowledge and skills to the solution of novel as well as routine problems	Lectures, tutorials, and especially the assignment projects where students have to collaboratively develop operating system functionality.
Entrepreneurial leaders capable of initiating and embracing innovation and change, as well as engaging and enabling others to contribute to change	The assignments provide the opportunity engage with a partner and bring them along the path to solving the problem at hand.
Professionals capable of ethical, self- directed practice and independent lifelong learning	Lectures, tutorials, and especially the assignment projects where students have to independently develop an understanding of the problem at hand and the solution required.
Global citizens who are culturally adept and capable of respecting diversity and acting in a socially just and responsible way	The assignments where students come to terms with working in groups

Teaching Strategies

Our approach to teaching operating systems is:

- introducing operating system theory in the lecture component of the course, including case studies of real systems
- actively 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, tutorials, and online forums 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 workforce.

Teaching Rationale

This course embraces the opportunity computer science provides to practically apply theory learned in a realistic way without requiring the extensive resources needed to build bridges, generate power, or manufacture machines as would be required for other engineering disciplines.

We use a teaching operating system (OS/161 from Harvard) with functionality not dissimilar to the a BSD-based or Linux-based UNIX operating system to provide a realistic environment to both observe and understand the inner working of an operating system and to enable student to write real OS code to extend its functionality.

The rationale is that the practical nature of the course provides a much deeper understanding of operating systems than what can be obtained solely from theory, theoretical analysis, and writing applications alone.

Student Conduct

The **Student Code of Conduct** (Information (https://student.unsw.edu.au/conduct) , Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)) sets out what the University expects from students as members of the UNSW community. As well as the learning, teaching and research environment, the University aims to provide an environment that enables students to achieve their full potential and to provide an experience consistent with the University's values and guiding principles. A condition of enrolment is that students *inform themselves* of the University's rules and policies affecting them, and conduct themselves accordingly.

In particular, students have the responsibility to observe standards of equity and respect in dealing with every member of the University community. This applies to all activities on UNSW premises and all external activities related to study and research. This includes behaviour in person as well as behaviour on social media, for example Facebook groups set up for the purpose of discussing UNSW courses or course work. Behaviour that is considered in breach of the Student Code Policy as discriminatory, sexually inappropriate, bullying, harassing, invading another's privacy or causing any person to fear for their personal safety is serious misconduct and can lead to severe penalties, including suspension or exclusion from UNSW.

If you have any concerns, you may raise them with your lecturer, or approach the School Ethics Officer (mailto:ethics-officer@cse.unsw.edu.au) , Grievance Officer (mailto:grievance-officer@cse.unsw.edu.au) , or one of the student representatives.

Plagiarism is defined as (https://student.unsw.edu.au/plagiarism) using the words or ideas of others and presenting them as your own. UNSW and CSE treat plagiarism as academic misconduct, which means that it carries penalties as severe as being excluded from further study at UNSW. There are several on-line sources to help you understand what plagiarism is and how it is dealt with at UNSW:

- Plagiarism and Academic Integrity (https://student.unsw.edu.au/plagiarism)
- UNSW Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

Make sure that you read and understand these. Ignorance is not accepted as an excuse for plagiarism. In particular, you are also responsible that your assignment files are not accessible by anyone but you by setting the correct permissions in your CSE directory and code repository, if using. Note also that plagiarism includes paying or asking another person to do a piece of work for you and then submitting it as your own work.

UNSW has an ongoing commitment to fostering a culture of learning informed by academic integrity. All UNSW staff and students have a responsibility to adhere to this principle of academic integrity. Plagiarism undermines academic integrity and is not tolerated at UNSW. Plagiarism at UNSW is defined as using the words or ideas of others and passing them off as your own.

If you haven't done so yet, please take the time to read the full text of

- UNSW's policy regarding academic honesty and plagiarism (https://student.unsw.edu.au/plagiarism)

The pages below describe the policies and procedures in more detail:

- Student Code Policy (https://www.gs.unsw.edu.au/policy/documents/studentcodepolicy.pdf)
- Student Misconduct Procedure (https://www.gs.unsw.edu.au/policy/documents/studentmisconductprocedures.pdf)
- Plagiarism Policy Statement (https://www.gs.unsw.edu.au/policy/documents/plagiarismpolicy.pdf)
- Plagiarism Procedure (https://www.gs.unsw.edu.au/policy/documents/plagiarismprocedure.pdf)

You should also read the following page which describes your rights and responsibilities in the CSE context:

- Essential Advice for CSE Students (https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students)

Assessment

Class work

There will be three main assignments plus a warm up assignment to encourage you to set up you development environment - ASST0 - ASST3. They are due approximately in weeks 2, 4, 7 and 10. Note that the ASST0 assignment is a really trivial familiarisation exercise, and should not be used to judge the difficulty of the other assignments. ASST0 and ASST1 are completed as individuals, ASST2 and ASST3 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 (http://www.eecs.harvard.edu/~syrah/index.shtml) group. The simulator is relatively platform independent. Assignment solutions in C code will be submitted as git repositories. 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 is intentionally kept small compared to the amount of extra work required. The marks are limited to 10% of the class mark component of the course, see the Assessment section below for details.

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 2% per day bonus marks (2% of the raw mark awarded) for submitting *their final solution before the deadline* (i.e., the bonus marks are reduced accordingly if the assignment is re-submitted before the deadline). Such bonus marks are capped at 10% per assignment and will allow students to make up for missed marks in the same assignment (i.e., one cannot get above the full marks for the assignment).

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 5 days after the deadline.** The late penalty is purposely lenient due to the tendency of students to underestimate the work required.

- Group 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 expected to contribute equally to EACH group assignment.. In submitting an assignment, the group is agreeing that all members have contributed equally unless we receive an agreed-to statement from all group members identifying differing levels of contribution.
- Submissions are required to have significant contributions attributable to each individual group member that can be verified using the revision control system.

Assignment	**Topic**	**Due**	**Marks**
ASST0	Warm-up	Week 2	10
ASST1	Concurrency	Week 4	30
ASST2	System calls and file systems	Week 7	30
ASST3	Memory management	Week 10	30
Class Total			**100**

Final Exam

A two hour final examination will be conducted.

Supplementary assessments

Supplementary exams will only be awarded in well justified and documented cases of illness or misadventure, in accordance with UNSW's policy for Special Consideration, not as a second chance for poorly performing students. You should familiarise yourself with UNSW special consideration policy, especially "Fit-to-sit". Make up your mind whether or not you are sick before attempting the exam.

- Special Consideration (https://student.unsw.edu.au/special-consideration)

Final Assessment

Item	Topics	Due	Weight
Class Work	All topics	Weeks 2,4,7,10	40%
Final Exam	All topics	Exam period	60%

The final assessment is based on a "class work" (C) and a final exam (E). The class mark is based on the assignment work and advanced bonuses accrued during the semester.

The class mark (C) is formed as follows: (asst0 + asst1 + asst2 + asst3) + min(10, adv_bonus)

As described above, any early bonuses contribute to the applicable individual assignment (which is capped at max marks for that assignment) and are not acrued.

(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 geometric mean of the ``class mark'', ***C*** , and the final exam mark, ***E*** , is used to determine the final mark, ***M*** , accordingly a 40/60 weighted geometric mean (40 class mark, 60 exam) as shown below.

M = C^0.4 * E^0.6

A final mark of 50% is required in order to pass.

If either the class (*C*) or exam (*E*) component is less than 40%, the final mark (*M*) is capped at 45.

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

Course Schedule

Subject to change

Week	Lectures	Tutes	Assignments	Notes
1	OS Overview, processes, threads, concurrency	No tutorial	-	-
2	Concurency, deadlock,	Operating Systems Overview, Processes and Threads, Critical Sections	ASST0 due	-
3	Process & thread implementation, system calls, R3000	ASST1 related, concurrency and deadlock	-	-

4	File systems	R3000, Processes and Threads, Kernel Entry/Exit	ASST1 due	-
5	File systems	Memory Hierarchy and Caching, Files and file systems	-	-
6	Flexibility week	ASST2 consult	-	-
7	Memory management, Virtual memory	File Systems	ASST2 due	-
8	Virtual memory, Multiprocessors	Memory Management, Virtual Memory	-	-
9	Scheduling, I/O	ASST3 related	-	-
10	No lecture	Virtual Memory, Multiprocessors, Scheduling, I/O	ASST3 due	-

Resources for Students

Textbook

- A. Tannenbaum and H. Bos, *Modern Operating Systems* , 4th ed.

Reference Books

For Operating Systems:

- A. Silberschatz, P.B. Galvin and Greg Gagne, *Operating System Concepts*
- William Stallings, *Operating Systems: Internals and Design Principles*
- A. Tannenbaum, A. Woodhull, *Operating Systems--Design and Implementation*
- 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, manuals, and other information can be found under the course's WWW home page at URL http://www.cse.unsw.edu.au/~cs3231/.

Course Evaluation and Development

This course is evaluated each session using the myExperience system **and my own surveys.**

The surveys, a detailed analysis, and developments as a result are documented on the course website http://cgi.cse.unsw.edu.au/~cs3231/surveys.html (http://cgi.cse.unsw.edu.au/~cs3231/surveys.html)

Resource created 4 months ago (Tuesday 09 February 2021, 10:04:13 AM), last modified 4 months ago (Thursday 18 February 2021, 08:09:06 AM).

Comments

(/COMP3231/21T1/forums/search?forum_choice=resource/58036)

(/COMP3231/21T1/forums/resource/58036)

There are no comments yet.