ECSE 427/COMP 310 -- Operating Systems

Fall 2018

General Information

Instructor:	Muthucumaru Maheswaran (“Mahesh”)
Tel:	398-1465
Email:	**maheswar@cs.mcgill.ca**
Office:	Room 754, McConnell Engineering Building
Office hours:	TuTh 12:00-1:00pm. Appointments can be made for meetings at other times.
Class:	MC 304
Tutorial:	TBA
Prerequisites:	see Calendar
Class web page:	No class web page. My Courses will be used for assignments drop off and class discussions.
TAs:	TBA
TA office hours:	TBA

“McGill University values academic integrity. Therefore, all students must understand the meaning and consequences of cheating, plagiarism, and other academic offences under the Code of Student Conduct and Disciplinary Procedures (see www.mcgill.ca/integrity for more information).”

Course Description and Learning Outcomes

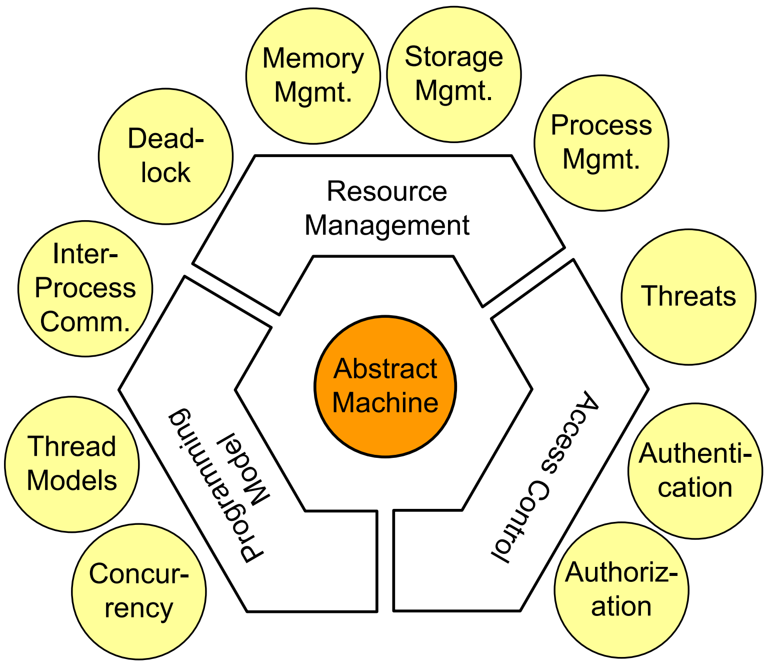
Description: This is an introductory course in computer operating systems. In this course we will study the theoretical and practical concepts behind modern operating systems. In particular, we will study the basic structure of an operating system, its components, design strategies, algorithms and schemes used to design and implement different components of an operating system. Major components to be studied include: processes, inter-process communication, scheduling, memory management, virtual memory, storage management, network management, and security.

Primary learning outcome: To get a clear ***understanding*** of the major principles/algorithms that underlie an operating system and how they interplay within it.

Secondary learning outcomes: After taking this course, you should be able to: (i) **identify** the core functions of operating systems and how they are architected to support these functions, (ii) **explain** the algorithms and principles on which the core functions are built on, (iii) **explain** the major performance issues with regard to each core function, and (iv) **discuss** the operating system features required for particular target applications.

Course Content

CONCEPT MAP



Course Schedule

The table below shows a tentative course schedule (**it is likely I will be rearranging the material a bit as the semester progresses**).

Week	Topics	Reading Material	Comments
1	**OS Introduction**: OS concepts	1.5, 1.7	1.8 highly recommended
2	**Process Concept:** processes, introduction to process scheduling, IPC, Threads – multithread programming, issues	1.6, 2.1, 2.2	
3	**Synchronization**: Critical section problem, Peterson's solution, synchronization hardware, mutex, semaphores, monitors, well-known problems in synchronization	2.3, 2.5	1st Mini Assignment out (tentative)
4	**Deadlocks:** models, characterization, prevention, avoidance, detection, recovery	Chap. 6	
5	**File Systems:** file concept, file system structure, implementation, allocation	4.1-4.5	**Midterm #1**

	methods, free-space management, efficiency & performance, access methods, directories, disk structure, disk scheduling, etc		
6	**Advanced File Systems:** Log-structured file systems, fault tolerance, examples of modern file systems, SSD issues	Same as week 6	2nd Mini Assignment out (tentative)
7	**CPU Scheduling:** basic concepts, scheduling criteria, algorithms, thread scheduling, real-time scheduling, lottery scheduling, stride scheduling, performance of scheduling algorithms	2.4, 10.3.4	
8	**Main Memory:** swapping, memory allocation, segmentation, paging	3.1-3.7	Programming Assignment out (tentative)
9	**Virtual Memory:** demand paging, copy-on-write, page replacement, allocation of frames, thrashing	Same as week 9	
10	**Virtualization and Clouds:** Virtualization requirements, hypervisors, memory virtualization, I/O virtualization, clouds	7.1-7.11	
11	**Security:** goals of protection, domain of protection, access matrix, implementation, access control, capability-based systems	9.1-9.4	Midterm #2
12	**Security:** security problem, program threats, system & network threats, user authentication	9.7-9.10	

The course will consist of three hours of instructor led classes per week together with a *maximum* of one hour of tutorial per week taken by the TAs. The class time will be devoted to the presentation and development of new concepts and the application of these concepts to examples and problems, while the tutorials will discuss solutions to the programming projects and written assignments. The primary focus of the tutorials is to provide sufficient "how-to" knowledge through the discussion of the assignments to help in the development of the programming project series.

Instructional Method

The course will consist of three hours of instructor led classes per week together with a *maximum* of one hour of tutorial per week taken by the TAs. The class time will be devoted to the presentation and development of new concepts and the application of these

concepts to examples and problems, while the tutorials will discuss solutions to the programming assignments/projects and written assignments. The primary focus of the tutorials is to provide sufficient "how-to" knowledge through the discussion of the assignments to help in the development of the programming project series.

Students are strongly encouraged to use the My Courses discussion groups to talk about the programming and written assignments. These discussion groups will be monitored by the TAs and by the instructor for providing the necessary answers.

Course Materials

Required Textbook

A. Tenenbaum and H. Bos, Modern Operating Systems, 4th Edition, Pearson, 2015.

Evaluation

Activity	Weight (in percent)
Mini Programming Assignment 1:	7%
Mini Programming Assignment 2:	8%
Programming Assignment:	15%
Midterm #1	10%
Midterm #2	10%
Final (comprehensive)	50%
Total	**100%**

NOTE 1: Final letter grades could be assigned on a curve.
NOTE 2: The official programming language of this course is C.

Double Grading Policy: This course has a significant portion of the grade allocated for the programming component. You are expected to submit ***only*** your work in these assignments. You can receive advise or tips from others (instructor, teaching assistants, or peers), but the final submission should be yours. You are expected to know all the design decisions in the program and explain all aspects of the program handed in as part of the assignment. To test this condition, we will randomly select some students and ask them to explain their programming assignments. The eventual marks for an assignment will be the minimum of the two marks. For example, if 85 is the marks obtained in the first (normal) evaluation of the programming assignment and 50 is the marks obtained in the second evaluation, then effectively you have 50.

Late Assignment Policy: There will be two deadlines for each assignment: proper deadline and cut-off date. After the proper deadline, there will be a penalty of 10% for each day the assignment is late until the cut-off date. After the cut-off date, the assignment cannot be handed in. No individual requests for extensions will be granted unless they are for medical reasons.

The deadlines will be set for 11:55 pm or 11:59pm. Please observe the time and date very carefully. It is your responsibility to make sure that the assignment is properly submitted via the WebCT.

Regrading Policy: If you find your assignments or exams are not marked according to the marking scheme, you are encouraged to consult me or the TAs. When you resubmit your assignment or exam for regarding, we reserve the right to regrade the full exam or assignment without restricting the attention to the disputed portion.