CISC 324 WINTER 2018: TOPICS, SCHEDULE, MARKING

Instructor: Dorothea Blostein, Goodwin 720, (613) 533-6537, blostein@cs.queensu.ca

Course Website: www.cs.queensu.ca/home/cisc324

TOPICS

The two main topics of this course are concurrent programming and operating systems.

1. Concurrent programs consist of *processes* that execute in parallel. The processes communicate with each other by referencing shared data structures or by sending messages. References to shared data structures have to be synchronized, to ensure consistency and avoid corrupting the data structures. The labs in this course use Java threads to explore issues that arise in concurrent programming: interprocess communication, mutual exclusion, deadlock, and starvation.

   Concurrent programming is useful in a variety of applications. Here are some examples.

   - An operating system can be coded as a set of concurrent processes. This is a convenient way to organize code that lets the operating system keep track of many activities and users at once. A typical operating system contains separate processes for memory management, for managing manage the file system, for receiving input typed by a user, and so on. New processes are created when users log in or launch an application.

   - Simulations often use concurrent programming. For example, suppose we want to simulate traffic flow in order to evaluate the impact of widening a two-lane road into a four-lane road. We can implement this by coding a *car* process to define the behaviour of a car, and then creating many instances of this car process. Intersections are coded as synchronization constraints among the car processes, to make sure that the simulated cars obey the rules of the road at the simulated traffic lights and four-way stops.

   - Concurrent programming arises naturally in distributed applications, such as banking networks, telephone networks, or airline-reservation systems. As long ago as 1990 Bell Northern Research (BNR) had over three million lines of concurrent Pascal code associated with the telephone network.

2. An operating system provides a convenient and efficient interface for the computer user. We study how operating systems bridge the gap between the hardware and the interface to the user. Operating systems must permit several users to share the same hardware, and to cooperate in mutually-agreed upon ways, while ensuring that one user cannot accidentally or maliciously interfere with other users.

   An operating system consists of software written in a mixture of high-level language, assembly-language, and perhaps microcode. An operating system can be organized into layers, with the hardware as the lowest layer, and the user interface as the uppermost layer. Alternatively, an operating system can be organized as a kernel (which implements the concept of *process*), augmented by a collection of processes for memory management, file management, CPU scheduling, etc.

TEXTBOOK

The textbook and course reader are available from Queen's bookstore.

Operating System Concepts, Ninth Edition, by Silberschatz, Galvin and Gagne; published by Wiley. The eighth edition can be used as well.

CISC324 Course Reader, D. Blostein, 2018.

MARKING SCHEME

This course uses mixed marking, with the conversion between percentage and letter grades shown on page 4. The marking scheme is as follows.

Assignments/Labs	24%	Six assignments are worth 2% each. Six labs are worth a total of 12%: Labs 1 and 2 together are 1%; Lab 3 is 4%; Lab 4 is 3%; Lab 5 is 1%; Lab 6 is 3%
Two in-class Exams	26%	Each exam is worth 13% of the course mark. The exams take place on Thursday February 15 and Monday March 19.
Final Exam	50%	

ASSIGNMENTS AND LABS

The six assignments contain written questions to help you learn the course material. The six labs give you practice in concurrent programming using Java**.** This course does not have scheduled lab times; you may complete lab work anytime. Teaching Assistants are available to help with labs, both via email and during office hours. **Assignments are posted on the course website and lab instructions are provided in the course reader.**

Assignments and labs may be completed **individually or in pairs**. If you work as a pair, be sure that both of you participate fully in solving the assignment problems and in designing, implementing and debugging the code for labs.

Assignments and labs are due **at the beginning of class** on the due date. Late assignments are not accepted. If a student has a documented reason for missing an assignment, then the marks from other assignments replace the missed assignment.

Do not copy assignment answers. Doing that wastes your time and your tuition. The only effective way to learn course material is by working through problems on your own. Here is the Arts and Science description of academic integrity.

> Academic integrity is constituted by the five core fundamental values of honesty, trust, fairness, respect and responsibility (see www.academicintegrity.org). These values are central to the building, nurturing and sustaining of an academic community in which all members of the community will thrive. Adherence to the values expressed through academic integrity forms a foundation for the "freedom of inquiry and exchange of ideas" essential to the intellectual life of the University (see the Senate Report on Principles and Priorities www.queensu.ca/secretariat/policies/senateandtrustees/principlespriorities.html).
>
> Students are responsible for familiarizing themselves with the regulations concerning academic integrity and for ensuring that their assignments conform to the principles of academic integrity. Information on academic integrity is available in the Arts and Science Calendar, on the Arts and Science website (see http://www.queensu.ca/artsci/academics/undergraduate/academic-integrity), and from the instructor of this course. Departures from academic integrity include plagiarism, use of unauthorized materials, facilitation, forgery and falsification, and are antithetical to the development of an academic community at Queen's. Given the seriousness of these matters, actions which contravene the regulation on academic integrity carry sanctions that can range from a warning or the loss of grades on an assignment to the failure of a course to a requirement to withdraw from the university.

SCHEDULE AND TOPICS

This table shows the due dates for assignments and labs. The schedule of lecture topics is approximate, and may be adjusted slightly during the term. The *Textbook Sections* column is a rough guide; the assignments provide detailed information about the required readings in the textbook and course reader.

Exam 1 covers assignments 1-3 and labs 1-3. Exam 2 covers assignments 4-5 and lab 4. The final exam is comprehensive.

	Date	Due	Topics	Textbook Sections
1	Jan 8 Jan 9 Jan 11		Introduction, overview, review. • Review of instruction execution, interrupts, DMA. • Concurrent processes. Process Control Blocks. Context switching. • Operating system organized as kernel and processes	1, 2, 3.1-3.3
2	Jan 15 Jan 16			
	Jan 18	Assig 1	Process synchronization. • The mutual exclusion problem (software and hardware solutions). • Language constructs for process creation, synchronization and communication. • Use of semaphores. • Classic synchronization problems (readers/writers, dining philosophers)	5
3	Jan 22 Jan 23 Jan 25	Lab 1&2		
4	Jan 29 Jan 30 Feb 1	Assig 2		
5	Feb 5 Feb 6	Lab 3	Deadlock. • prevention, avoidance, detection, recovery	7
	Feb 8		Memory management • Memory pyramid. • Address spaces and address translation. • Review of linking and loading.	8, 9
6	Feb 12 Feb 13 Feb 15	Assig 3 **Exam 1**		
			READING WEEK	
7	Feb 26 Feb 27 Mar 1	Lab 4	Memory management, continued • Review of memory partitions, single-level paging. • Segmentation, without and with paging. • Multi-level paging. • Page replacement algorithms; local and global frame allocation. • Coding considerations for reducing page faults • Effective memory access time.	8, 9
8	Mar 5 Mar 6 Mar 8	Assig 4		
9	Mar 12 Mar 13			
	Mar 15	Assig 5	Monitors. Message Passing.	5.8, 3.4
10	Mar 19	**Exam 2**		
	Mar 20 Mar 22	Lab 5	CPU scheduling Threads: lightweight processes	6.1-6.3 4.1
11	Mar 26 Mar 27 Mar 29	Lab 6	Protection and security. • Example: buffer overflow attack. • Access lists and Capabilities. Creation and revocation of access rights.	14
12	Apr 2			
	Apr 3 Apr 5	Assig 6	OS structure: Layers, Kernel, Virtual machines. Review of course material: discuss aspects of Linux, Windows. For students who present posters at Creative Computing on April 5, the assignment 6 due date is extended to April 9.	2.7, 16.3 21, 22

MIXED MARKING

In this course, some components will be graded using numerical percentage marks. Other components will receive letter grades, which for purposes of calculating your course average will be translated into numerical equivalents using the Faculty of Arts and Science approved scale: the Arts & Science Letter Grade Input Scheme. Your course average will then be converted to a final letter grade according to Queen's Official Grade Conversion Scale.

Arts & Science Letter Grade Input Scheme		***Queen's Official Grade Conversion Scale***	
Assignment mark	**Numerical value for calculation of final mark**	**Grade**	**Numerical Course Average (Range)**
A+	93	A+	90-100
A	87	A	85-89
A-	82	A-	80-84
B+	78	B+	77-79
B	75	B	73-76
B-	72	B-	70-72
C+	68	C+	67-69
C	65	C	63-66
C-	62	C-	60-62
D+	58	D+	57-59
D	55	D	53-56
D-	52	D-	50-52
F48 (F+)	48	F	49 and below
F24 (F)	24		
F0 (0)	0		

COPYRIGHT OF COURSE MATERIALS