mount royal college
department of computer science and information systems

WINTER 2008 COURSE OUTLINE

course title COMP 4403: Object-Oriented Design And Development

credits 4 (3 hours lecture + 1 hour lab + 1 hour tutorial per week)

prerequisites COMP 1276 and COMP 1281 with minimum grades of "C"

instructor

Randy Connolly
rconnolly@mtroyal.ca
B233o
440-6061
http://www.randyconnolly.com

office hours As posted outside the Department. Feel free to drop in at anytime that is not crossed out on my schedule. If I am not available, an appointment can be arranged.

lectures	001	T	14:30 – 15:50	B-220
		R	13:00 – 14:20	B-217
labs	501	M	15:00 – 15:50	B-215
	502	M	16:00 – 16:50	B-215
tutorials	401	W	15:00 – 15:50	B-215
	402	W	16:00 – 16:50	B-215

description This course covers the principles of object-oriented analysis (OOA), design (OOD), implementation and testing. This year we will be using C# and the .NET Framework as our implementation environment for system development using OOA and OOD.

educational outcomes

Mount Royal College had identified six college-wide learning outcomes that it believes are critical in order to prepare its graduates for workplace success and a life of continuous learning. Generally speaking, "outcomes" are goals, results, objectives that you should derive from the College, from a program of study, and from a particular course. This course will emphasize the following outcomes:

College-Wide Learning Outcomes

This course covers many of the College-wide learning outcomes but emphasizes Thinking and Communication Skills such as:

- problem solving (assignments and project)
- creative thinking (alternative solutions for assignments)
- analytical thinking appropriate to the discipline of information systems (assignments and project)
- understand and use vocabulary and concepts appropriate to the discipline (assignments and project)
- interpret and evaluate meaning (assignments and project)
- communicate clearly and concisely using visual and written formats appropriate to the audience (assignments and project)

Skills in Group Effectiveness will also be developed.

Course and Discipline Outcomes

- retain a detailed knowledge of object-oriented principles
- Use OO approaches to requirements gathering
- refine the OOA model to include design factors
- refine the OOA/OOD models to reflect implementation details
- architect systems using design patterns
- design object systems suitable for enterprise contexts
- use UML to visualize the design of a software system
- implement the design using an object-oriented programming language

course format

There are five hours of scheduled instruction per week. New material will be formally presented in lectures. Tutorials and labs will be used to reinforce the lectures by providing an opportunity to study examples and practice the application of concepts. Tutorials and labs are not just for students who require extra help; they are an integral part of the course and attendance is required. You will also have to do a lot of work outside of scheduled class time.

grading

The final grade for this course will be determined as follows:

Tutorial/Lab Exercises	10%
Milestones	40%
Midterm (February 14, 2008)	20%
Final Exam	30%

Percentage grades will be converted to letter grades as follows:

95-100	A+	67-69	C+
85-95	A	63-66	C
80-84	A-	60-62	C-
77-79	B+	55-59	D+
73-76	B	50-54	D
70-72	B-	<50	F

The College's complete grading system is described in the Calendar.

examinations

The midterm date is specified in the marks tentative schedule section. If any changes to this date are necessary, students will be notified well in advance. Students will not normally be permitted to write a missed test at a later date. If alternative arrangements are possible, they must be made with the instructor prior to the date of the test.

The examinations will focus on understanding and applying the concepts taught in class. These tests will be made up of short answer style questions, as well as a limited number of larger questions to test students' abilities.

tutorial/lab work

The tutorial/lab work will be done during the labs, or it will require only a few days to complete, and it will focus on a particular aspect of a problem. Students will be assigned a pass/half/fail grade.

assignments

Assignments will consist of larger applications of the topics covered in lectures. The assignments will normally require several weeks of work and may be broken down into more than one component, e.g. an interface, client scripts, and server scripts.

Note: Students are expected to complete the assignments individually.

Problem solving techniques and skills can only be acquired through practice and through the study of increasingly more difficult problems. The assignments all involve problem solving. It is very important that you understand how you solved the problem, and not just be happy with handing in a computer program that produces the requested results.

Assignments will be considered late if submitted after the time specified on the assignments. Unless an assignment states otherwise, it will be accepted up to one day late; however, 10% will be deducted for being late (even for part of a day late). Assignments will not be accepted more than one day late. This includes weekends, so if an assignment is due Friday at 2:00 p.m. then Saturday at 2:00 p.m. is the latest it can be handed in. Start your work early and schedule adequate time for completion.

facilities

The assignments and the project will be written in C# and will use Microsoft Visual Studio 2008. Students will have access to this software through their CSIS accounts.

You are responsible for the security of your computer account. Choose a password that cannot be easily guessed by others and keep it confidential.

The course library will be used for distributing course-related information.

cheating

It is expected that all work handed in by a student will be original work that has been done by the individual. If it is not, then this act of intellectual dishonesty will be dealt with severely. An *Academic Dishonesty Incident Report* will be filed with the Office of Student Conduct. If a record of previous academic dishonesty is established, the case will be forwarded to the Academic Integrity Board. The complete process is described in Mount Royal's *Code of Student Conduct.*

While students are expected to work reasonably independently, we do not expect you to work in isolation. Often you learn best when working with others on an assignment. So what degree of collaboration is expected and, indeed, encouraged, and what is deemed to be cheating?

In general, we encourage things like bouncing ideas off one another, discussing which of two alternate solutions might be better (and why), and getting another's ideas on how to resolve a difficulty that you have already spent time on. You should not be working so closely together that someone else's solution becomes incorporated into your product. These general guidelines apply to **any type** of assignment. Some more detailed guidelines for programming assignments follow.

There are two main steps in creating a computer program—the first is to determine how to solve the problem (by understanding all the nuances of the problem and then designing a solution), the second is to write the program and thoroughly test it. You need to become proficient in both areas.

In the first step, you need to understand the problem. First, analyze the problem yourself and try to understand it as much as possible. Then, if you like, discuss your ideas with others—this way you can learn from each other. Once you feel you understand the problem, decide how to approach solving it. First come up with a high-level design yourself. After you have the first cut at a design, you may want to compare notes with others to see if you've missed anything or to help you resolve a difficulty you have with your design. However, you **MUST** come up with your own design first; if not, you will never acquire the skills required of a good programmer.

When you get to the point of writing the actual program, you must work independently. It is **NOT** acceptable to be coding with others. Write your program by taking your design, possibly fleshing out some details, and writing the code in the appropriate programming language. If you have difficulty with a certain statement, check your notes or your text. If you are still having problems with the code, then you can ask others for help. However, you must master the language as quickly as possible, and not rely on others. When testing the program, if you are having problems with a section and have spent some time trying to find the problem yourself, it is a good idea to ask others if they can help you. Other people will often see errors that you cannot see because you are too close to your solution – that is, you no longer see it clearly. Don't forget that if you need help at any time, the Instructional Assistant and your Instructor are available.

When complete, two students' programs should be essentially independent of one another. Each student should have attempted each step of the problem solving method alone before discussing it with others. In this way you can develop your own skills, while still learning from (and helping) others.

general department policy

Students are responsible for attendance at all lectures and labs, for completion of assignments in open lab time, and for requesting assistance from their instructor or from the instructional assistant when they are having difficulty with the course material.

If this course is a prerequisite for other courses, the minimum grade required in order to take the subsequent course is stated elsewhere in this course outline.

The midterm test dates are indicated in the Assessment section. Should changes become necessary, students will be notified well in advance. Students will not normally be permitted to write a missed test at a later date. If alternative arrangements are possible, they must be made with the instructor prior to the date of the test.

The final examinations will be scheduled by the Registrar during the period from April 16 - April 26, 2008. Do not make any plans for that period until the final examination schedule has been posted.

Programs will be graded for documentation and style, as well as for correctness. All files must be left in the student's directory until the marked program has been returned.

As a rule, the deadline for assignments will not be extended for computer downtime of less than 24 hours; however, this will be at the instructor's discretion. Any exception will be communicated to the class as quickly as possible.

In general, assignments are due at 16:00 and will be considered late if submitted after that time. Assignments will be accepted up to one day late; however, a penalty of 10% will be deducted, even for a partial day late. Assignments will not be accepted more than one day late. This includes weekends!

Students should familiarize themselves with the College policy on the integrity of student work as described in the Calendar and with the departmental policy on cheating detailed on the attached sheet. Cheating of any form is a serious matter and will be dealt with severely.

The last day for withdrawal from this course is March 20, 2008.

Students should familiarize themselves with the Statement of Student Rights and Responsibilities contained in the College Calendar.

topic outline

The course topics are listed below. Not all topics will be covered in the same degree of detail and the sequence may differ somewhat from the list. Lectures will provide more material than that covered in the text – you are responsible for all material covered in the course, not just the content of the text.

- Introduction to Object-Orientation
  - What does it mean to be object-oriented?
  - What are objects?
  - Basic object concepts.
- Lifecycles, Processes and Models
  - System development lifecycles and methodologies.
  - What are models and their role in system design and development.
  - The Unified Process (UP)
  - Overview of UML.
- Understanding and Documenting Requirements
  - Capturing requirements with Use Cases.
- Object-Oriented Analysis
  - Identifying real-world things with class diagrams.
  - Expressing how things work together with sequence and collaboration diagrams.
- Object-Oriented Design
  - Refining the structure of things with class diagrams.
  - Describing flows with activity diagrams.
  - Showing how groups of things work together with collaboration diagrams.
  - Making use of design patterns.
- Object-Oriented Implementation
  - Describing how things will be built using component, deployment, and physical diagrams.
  - Moving from design to code.
  - Using Java for object-oriented programming.

4403 TENTATIVE SCHEDULE

Week	Tues	Thurs	Due
Jan 7-11	Intro to Objects	Lifecycles + Methodologies	
Jan 14-18	Object Concepts	Object Concepts	
Jan 21-25	Test-Driven Development	Test-Driven Development	
Jan 28-Feb 1	Object Development	Object Development	
Feb 4-8	Analysis	Analysis	Milestone 1
Feb 11-15	Review	Midterm	
Feb 18-22	Reading Week		
Feb 25-29	Midterm Evaluation	Analysis	Milestone 2
Mar 3-7	Design	Design	
Mar 10-14	Design	Design	
Mar 17-21	Design	Design	Milestone 3
Mar 24-28	Design	Design	
Mar 31-Apr 4	Design	Design	
Apr 7-11	Design	Review	Milestone 4