

Course Objectives

Welcome to CSC207H, an introduction to software design. One major goal of this course is to introduce you to large-scale software design and development concepts and tools that become useful as you work on projects in teams. We will discuss effective team behaviours and communication skills, practise agile methods for designing software, and use tools such as a fully-featured IDE and a version control system. This course is thus somewhat of a software engineering course, with a lot of emphasis on processes and standards, not just strict functional correctness of code. Please note this very carefully, as a significant number of marks will be lost if you do not follow the requirements.

The other primary goal of this course is to help you practise how to learn a new language, so instead of focusing on syntax as in CSC108H, we will compare salient features of Python and Java, expecting you to fill in details outside of class, and we will investigate Java’s memory model, scoping facilities, and object-oriented structures in depth. By the end of this course, we believe that you will pick up new languages and tools with little guidance and expect that you will be prepared to develop code in small teams.

Contact Information

<b>Instructor</b>	Philip Patchin
<b>Lectures</b>	L0101: WF 1-2 (BA 1210) Lab: M1-2 (BA 3175)
<b>Website</b>	<a href="http://www.cdf.toronto.edu/~csc207h/winter/">http://www.cdf.toronto.edu/~csc207h/winter/</a>
<b>Office Hours</b>	W2-3, F2-3 in BA2200
<b>Email</b>	philip [at] cs [dot] toronto.edu

Resources

The required text for this course is *Developing Java Software* (3ed) by Winder and Roberts. We will also assign required readings from online sources; links to those readings will be posted on the course webpage. The Java API is also available online.

Email and Discussion Board

The course has a discussion board that should be your first stop for CSC207H information. Everyone can post questions – and answers – so the discussion board will typically be the fastest way to get help with course material or to obtain an answer to an administrative question. Please become active on the board; it will work best if everyone posts questions and replies. The board will also benefit from use of good thread names and adherence to the one-topic-per-thread rule.

Please use email for personal issues and the discussion board for general course-related questions. Include “207” in all email subject lines lest your message accidentally be filed as spam. An informative subject line like “207: intending to break my leg before the quiz” really helps. I try to answer email by the end of the next day. However, it may occasionally take longer, especially on weekends and near due dates. I expect emails to be written in a professional manner. This means that they should be clear and should use good English.

**The discussion board is required daily reading, as is email.** You are responsible for all announcements made in lecture and on the discussion board. We will also occasionally send important course announcements to either your CDF account or your official university email address, so please make sure that you know how to access these and check them regularly.

Syllabus and Term Due Dates

M-F Dates	Topic	Course Work Due
10–14 Jan	Subversion, Intro to Java	
17–21 Jan	Java Syntax and Memory Model	A0: SVN and Java basics (5%)
24–28 Jan	Java OO	A1: More Java practice (5%)
31 Jan–4 Feb	Interfaces and the Development Cycle	A2: Yet more Java (5%)
7–11 Feb	The Scrum Process and Effective Teamwork	Java quiz (in tutorial) (5%)
14–18 Feb	Design Case Study	Project Introduction (10%)
21–25 Feb		Reading Week
28 Feb–4 Mar	Design Case Study (Parsing and Swing)	
7–11 Mar	Documentation and Testing	Project Milestone (10%)
14–18 Mar	Design Case Study (Applets) and Usability	
21–25 Mar	Reflection and Understanding Programs	Project Due (10%)
28 Mar–1 Apr	Memory and the Machine	
4–8 Apr	Wrap-up and Review	A3: Project Code Review (10%)
12–29 Apr		Final exam (40%). <u>You must achieve 40% on the final exam to pass the course.</u>

**Assignments**

The assignments are individual coding projects that combine several concepts or Java features from class and the tutorials. Handouts for assignments will be available on the course website, and they will generally be due on Thursdays at 11:59 p.m. Late assignments will not be accepted.

Assignments must pass all of the basic tests that we post with the handout to be marked. Functionality counts! If your assignment passes the basic tests, we will mark it for style and design and will apply a more advanced set of tests. Much of the emphasis in this course is on developing appropriate procedures for designing software, so expect half or more of the assignment mark to be based on your program's design and on evidence that you've followed the development procedures we learn, e.g. correct use of SVN, good code style, and appropriate testing. Process counts!

**Project**

The project is a significant, team-based design and programming assignment. It will require you to work effectively as a team, to communicate your project goals in written and oral form, to use project management tools, and to demonstrate familiarity with Java.

The project will be marked in three phases. The first phase will be a proposal presentation and document. The second and third phases require increasingly complete projects and test suites. Marking schemes will be published in advance for the project phases. This means that failure to meet the required standards will be treated harshly. After the third phase, there will be an individual assignment in which you will each do a code critique of another team's project. This means that anything you submit may be seen by other teams (anonymized).

**Quiz and Final Exam**

The course has a quiz in tutorial during week 5 and a three-hour final exam in the final exam period. Both test the material from lectures, tutorials, and assignments. The final exam is comprehensive, and you must obtain a mark of at least 40% to pass the course. If your mark is less than 40%, your final mark will be set to be no higher than 47%.

**Labs**

There are two kinds of labs: optional ones and project ones. You **must** attend the project labs, because they are for team meetings. If you miss a project lab without a valid excuse, your project mark will be strongly penalized. There will be more details as the project approaches.

**Missed Work and Re-marks**

In case of an emergency that will cause you to miss a test or other deadline, please contact the instructor within 24 hours of the due date. It is far easier to find a solution well before the due date, so when possible, please inform the instructor as soon as you know accommodation will be required. In case of illness, have your doctor complete an official U of T medical certificate. For other emergencies, be prepared to provide other documentation requested by the instructor.

If a piece of work has been mis-marked or if you believe the rubric used to evaluate the work is not appropriate, you may request a re-mark. For a re-mark to succeed, you must clearly and concisely express what you believe was mis-marked or unfairly marked. To request a re-mark, set up an appointment with the instructor within two weeks of the work being returned. Be prepared for the entire work to be re-evaluated and for the mark to be adjusted up or down after the re-evaluation.

**Academic Offences**

All of the work you submit must be completed by you alone, and your work must not be submitted by anyone else. Sharing your work or using outside resources without prior permission and appropriate citation is academic fraud and is taken very seriously. The department uses software that compares programs for evidence of similar code. Please read subsections B.i. and B.ii. from the "Code of Behaviour on Academic Matters" here:

<http://www.governingcouncil.utoronto.ca/policies/behaveac.htm>

Here are a couple of guidelines to help you avoid an academic offense:

- Only discuss marked work with course and Help Centre TAs or the instructor. Studying concepts with other students is fine, but anything that relates to the project or an assignment is off limits.
- Never look at another student's solution (or in the case of group work, another group's solution), whether it is on paper, a board, or on a computer screen.
- Never show another student your assignment solution. This applies to all drafts of a solution, to incomplete solutions, and to pseudocode and diagrams.
- Do not use code that you find online without discussing the issue with your instructor.