COMP-202A: Introduction to Computing 1

McGill University, Fall 2009

Course Details

<table>
<thead>
<tr>
<th>Section 1</th>
<th>Instructor: Mathieu Petitpas (Course Coordinator)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Office: McConnell Engineering Building (ENGMC) 104</td>
</tr>
<tr>
<td></td>
<td>Office hours: MW 14:00 – 15:30 (or by appointment)</td>
</tr>
<tr>
<td></td>
<td>Contact info: <a href="mailto:mathieu.petitpas@cs.mcgill.ca">mathieu.petitpas@cs.mcgill.ca</a></td>
</tr>
<tr>
<td></td>
<td>Lecture room: BURN 1B45</td>
</tr>
<tr>
<td></td>
<td>Class times: MWF 11:35 – 12:25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section 2</th>
<th>Instructor: Kamal Zellag</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Office: McConnell Engineering Building (ENGMC) 105</td>
</tr>
<tr>
<td></td>
<td>Office hours: T 14:30 – 16:00 (or by appointment)</td>
</tr>
<tr>
<td></td>
<td>Contact info: <a href="mailto:zkamal@cs.mcgill.ca">zkamal@cs.mcgill.ca</a></td>
</tr>
<tr>
<td></td>
<td>Lecture room: ENGTR 0100</td>
</tr>
<tr>
<td></td>
<td>Class times: TR 13:05 – 14:25</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section 3</th>
<th>Instructor: Mathieu Petitpas (Course Coordinator)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Office: McConnell Engineering Building (ENGMC) 104</td>
</tr>
<tr>
<td></td>
<td>Office hours: MW 14:00 – 15:30 (or by appointment)</td>
</tr>
<tr>
<td></td>
<td>Contact info: <a href="mailto:mathieu.petitpas@cs.mcgill.ca">mathieu.petitpas@cs.mcgill.ca</a></td>
</tr>
<tr>
<td></td>
<td>Lecture room: BURN 1B45</td>
</tr>
<tr>
<td></td>
<td>Class times: MWF 12:35 – 13:25</td>
</tr>
</tbody>
</table>

Important Links

- myCourses (WebCT Vista): http://www.mcgill.ca/mycourses/

Contacting Instructors and Teaching Assistants

Post all your questions about assignments on the myCourses message boards for all to see both the questions and the answers. You may freely answer other students’ questions as well, with one important exception: you may not provide solution code (although you are permitted to provide one or two lines of code to illustrate a point). Of course, you can send e-mail to a teaching assistant or instructor directly for private matters; to that end, you may use the e-mail facilities provided by McGill or any e-mail account you have with any e-mail provider.

Students are expected to monitor their McGill e-mail account, myCourses, and the course home page for course-related news and information.
Introduction

The School of Computer Science (SOCS) would like to welcome you to COMP-202. We intend on making this course very interesting. The purpose of this document is to provide you with an overview of what lies ahead in this course. We shall begin with a brief introduction of the course contents, followed by some important general information about the course. Please read this document carefully and keep it for reference throughout the term.

Course Description

This course introduces students to computer programming and is intended for those with little or no background in the subject. You also do not need to have any knowledge of computer science in general. On the other hand, basic computer skills such as browsing the Web, sending e-mail, creating documents with a word processor, and other such fundamental tasks will be a valuable asset in this course.

The course uses the Java programming language. As with natural languages, programming languages can be grouped; languages in the same group are conceptually similar, while languages from different groups follow quite different paradigms. Java is an object-oriented language (as are C++ and many others). Examples of other language groups are imperative programming languages and functional programming languages.

Despite these differences, there are some basic building blocks in all languages that are fundamental to programming and software development in general. A large part of this course will naturally focus on these basic building blocks before we move to object-oriented or other language-specific concepts.

Learning how to program is not easy; it is not a set of facts that one can simply memorize. In principle, a computer program is simply a set of instructions that tells a computer to perform a task. However, finding the right set of instructions can be quite challenging. For that, one has to learn how to structure a larger problem into small subsets, and then find the solution to each particular subset. A large part of this course is dedicated to teaching students the way of thinking needed in order to build non-trivial programs.

What this course is not about

This course is not about how to use a computer. It will not teach you how to send e-mail, browse the Web, create word processing documents or spreadsheets, set-up and configure a computer, use specific software applications (except those needed to complete coursework), design Web pages, nor deal with operating system or hardware problems. However, the course offers introductory tutorials that provide instruction in aspects of computer usage necessary to complete coursework.

Course Prerequisites and Textbooks

Prerequisites:

- A CEGEP-level mathematics course or equivalent. For students who did not attend CEGEP, any upper-level mathematics course is sufficient. However, attention to detail, rigor, and the ability to think in an abstract manner is much more important than knowledge of calculus, algebra, or trigonometry.

Course textbook:


You may purchase a copy of this textbook from the McGill bookstore, from Paragraph Books (2220 McGill College Avenue), or from any other vendor. The comprehensive version of the textbook (which includes all the contents of the brief version) is also available on reserve at Schulich Library.
Other references:

- **Java Documentation.** You can browse or download this from Sun Microsystems’ Web site. Use the documentation appropriate for the Java version you are using.
  - Java 5.0 Documentation: http://java.sun.com/j2se/1.5.0/docs/
  - Java 6.0 Documentation: http://java.sun.com/javase/6/docs/
- **The Java Tutorial.** You can also browse or download this from Sun Microsystems’ Web site.
  - http://java.sun.com/docs/books/tutorial/

### Course Grading System and Deadline Policy

Your final grade in the course is calculated as follows:

- **Assignments:** 30%
- **Midterm Examination:** 20%
- **Final Examination:** 50%

In exceptional situations, students may write a supplemental examination. However, ability to do so is not automatic, and depends on your exact situation; contact your Student Affairs Office for further information. The supplemental examination represents 100% of your supplemental grade.

Students who receive unsatisfactory final grades will **NOT** have the option to submit additional work in order to improve their grades.

**Official (human) language policy for graded work:** In accord [sic] with McGill University’s Charter of Students’ Rights, students in this course have the right to submit in English or in French any written work that is to be graded.

### Assignments

There will be 5 assignments. Each of the first 4 assignments is worth 5% of your final grade, while assignment 5 is worth 10% of your final grade. It is important to complete all assignments, as this is the major way in which you learn the material. To receive full grades, assignments (as well as all other course work) must represent your own personal efforts (see the section on Plagiarism Policy and Assignments below).

Late assignments will be deducted 5% each day or fraction thereof for which they are late, including weekend days and holidays; that is, assignments that are between 0 and 24 hours late will be deducted 5%, assignments that are between 24 and 48 hours late will be deducted 10%, and so on. Assignments submitted more than 2 full days (48 hours) after the due date will not be accepted, nor graded, and will therefore receive a grade of 0. The instructors reserve the right to modify the lateness policy for any individual assignments; any such modifications will be clearly indicated at the beginning of the relevant assignment specifications. **Plan appropriately; exceptions will not be made without appropriate justifications supported by documentary evidence.**

### Midterm Examination

The midterm examination will take place on the following date and at the following time:

- **Tuesday, November 3, 2009, 18:30 - 20:30 (TENTATIVE; to be confirmed)**

Students who are absent from the midterm examination **MUST** justify their absence to the satisfaction of their section instructor, supported with documentary evidence. This must be done either before the midterm examination takes place, or within the 7 days that follow, including weekend days and holidays. For students
who satisfy these conditions, the weight of the midterm examination in their final grade will be transferred
to the final examination; that is, for these students, the final examination will be worth 70% of their final
grade. Students who fail to provide satisfactory justification for their absence will receive a grade of 0 for
the midterm examination.

All students who are absent from the midterm examination will have the option of taking the examination
at home and get it graded in order to get feedback on their progress in the course; however, in such cases,
their midterm examination grades will NOT be considered in the calculation of their final grade.

## Campus Computer Laboratories

### Using the SOCS computer laboratory facilities: All students registered in COMP-202 may use the
SOCS computer laboratory facilities to do their work regardless of the program in which they are registered.
These facilities are located on the third floor of Trottier building. All computers are physically accessible
on weekdays from 10:00 until 20:00, and on weekends from 12:00 until 20:00; a consultant will be on duty
during these times. Outside of these hours, only the computers located in the open area will be physically
accessible, and no consultant will be on duty. Please note that these hours may vary for the first week of
lectures.

In order to enter Trottier building before 7:00 or after 21:00 from Monday to Friday, or at anytime during
weekends, your McGill ID must be added to the building access list. This will enable card readers located
at the entrances of the building to recognize your McGill ID card. Your McGill ID will automatically be
added to the building access list if you are officially registered in a computer science course. However, if you
registered late or had other registration problems, this might not have been done in your case. If you are
officially registered in the course but unable to enter Trottier building using your McGill ID card outside the
building’s opening hours, contact the SOCS Systems Staff by e-mail at help@cs.mcgill.ca, and request
that your McGill ID be added to the building access list.

Students who wish to use the SOCS computer laboratory facilities must first create an account; this can
be accomplished by going to any computer on the third floor of Trottier building, logging in as newuser,
and supplying newuser as the password. You will then be invited to fill out a Web form. Upon completion
of this form, you will be provided with the user ID and password with which you will be able to use the
SOCS computer systems. Note that if you are not officially registered in this course, you will not be able to
create an account for use with the SOCS computer systems. You only need to perform the account creation
procedure once.

Almost all computers in the SOCS laboratory facilities run Ubuntu GNU/Linux, which is a Unix-like oper-
ating system. Members of the SOCS Systems Staff will hold Unix seminars at the beginning of the semester
for those who are new to Unix. Information regarding these seminars will be given during the first lectures.
If you are only familiar with a Windows (95/98/Me/NT/2000/XP/Vista) or Mac OS X environment, it is
strongly recommended you attend these seminars.

Refer to [http://socsinfo.cs.mcgill.ca/wiki/Main_Page](http://socsinfo.cs.mcgill.ca/wiki/Main_Page) for more information on the SOCS computer
laboratory facilities.

### Other computer laboratory facilities: You may also use other computer laboratory facilities on campus
to do your work. Most facilities are available to all McGill students, but there are facilities which grant
usage privileges only to students registered in a course or program offered by the faculty or department
which manages the facility.

Students should contact the work area of their choice to enquire about access requirements, opening hours,
or any further information such as software availability.
Personal Computers and Required Software

You will use the Java compiler on personal computers to compile the programs you are required to write for the assignments. The Java compiler is included in a larger software package called the Java Development Kit (JDK). You can use any text editor of your choice to write your programs, and then use the tools included with the JDK to compile and run them. During the first half of the course, we will be using a text editor called RText in the lectures and tutorials.

During the second half of the course, we will see how to use an integrated development environment (IDE) to write programs. IDEs provide an editor that allows you to type your program, commands to compile and run it, and to many other useful tools, all in one application. After IDEs are introduced, we will be using Eclipse, a very powerful IDE, during lectures and tutorials.

The JDK is installed on the computers in the SOCS laboratory, as are RText and Eclipse. On your own computer or in other laboratories, you may need to install these yourself or have the laboratory administrator(s) do so for you. Installing any of these is fairly straightforward.

You are encouraged to install the JDK, as well as an IDE if you wish to use one, on your own computer so you do not have to depend on the SOCS computer laboratory facilities to do your work.

- **Required:** The JDK.
  - Windows users: You may download the JDK installation program from the following Web site: [http://java.sun.com/javase/downloads](http://java.sun.com/javase/downloads) (choose *Java SE Development Kit (JDK) 6 Update 16*, with no additional software such as Java EE or NetBeans). The JDK is available at no cost, and there is no time limit on its use. You should install the JDK before any IDE.
  - Mac users: JDK 5.0 is installed by default on most Mac computers.
  - GNU/Linux users: JDK 5.0 or 6.0 is available in the software repositories of most of the major GNU/Linux distributions like Ubuntu, Fedora, and OpenSUSE; you can install it through your package manager.

- **Optional:** RText and Eclipse. In the first half of the course, we suggest that you use RText (for Windows, Mac OS X, or Unix), a free text editor suitable for writing programs. In the second half of the course, we suggest that you use Eclipse (for Windows, Mac OS X, or Unix), a free and powerful IDE. You may download these software packages from the following Web sites:

You should install these packages only after you have installed the JDK, as this will enable you to avoid several configuration problems.

**Teaching Assistants (TAs)**

Each TA will be available for at least one office hour per week, on the third floor of Trottier building, to help you with your assignments and answer questions about the course material. You can also contact TAs by e-mail; an e-mail address for each TA will be posted on the course home page.

**Submitting Assignments**

In this course, you will be using myCourses for assignment submission; instructors and TAs will discuss how to use it during the lectures and tutorials. Each assignment will contain directions on what to submit. Assignments will also be graded via myCourses.
In addition, myCourses will have a discussion board that you will be able to use to ask questions about assignments. Students, instructors, and TAs will scan and post answers to questions. Please do not post assignment answers or code (although you may post one or two-line long code snippets to illustrate a point)!

**Posting of Course Marks**

The course marks will be posted on myCourses. The marks will be updated after each assignment, as well as after the midterm. It is your responsibility to check that the marks are correct and to notify your section instructor of any errors or missing marks.

**Plagiarism Policy**

**Official policy:** McGill University values academic integrity. Therefore all students must understand the meaning and consequences of cheating, plagiarism, and other academic offenses under the Code of Student Conduct and Disciplinary Procedures (see [www.mcgill.ca/integrity/](http://www.mcgill.ca/integrity/) for more information).

**Plagiarism Policy and Assignments**

You **MUST** include your name and McGill ID number at the top of each program or module that you implement and submit. By doing so, you are certifying that the program or module is entirely your own, and represents only the result of your own efforts.

Work submitted for this course **MUST represent your own efforts.** Assignments **MUST** be done INDIVIDUALLY; you **MUST NOT** work in groups. Do not rely on friends or tutors to do your work for you. You **MUST NOT** copy any other person’s work in any manner (electronically or otherwise), even if this work is in the public domain or you have permission from its author to use it and/or modify it in your own work (obviously, this prohibition does not apply to source code supplied by instructors explicitly for this purpose). Furthermore, you **MUST NOT** give a copy of your work to any other person.

We will be using automated software similarity detection tools to compare your assignment submissions to that of all other students registered in the course, and these tools are very effective at what they have been designed for (note, however, that the main use of these tools is to determine which submissions should be manually checked for similarity by an instructor or TA; we will not accuse anyone of copying or working in groups based solely on the output of these tools). You may also be asked to present and explain your submissions to an instructor at any time. Finally, students who put their name on programs or modules that are not entirely their own work will be referred to the appropriate university official who will assess the need for disciplinary action.

Students who require assistance with their assignments should see a TA or instructor during their office hours. If you have only partially finished an assignment, document the program parts that do not work, and submit what you managed to complete for partial credit.

**Course Content**

Note that minor changes in content, reading material, and times for tutorials and assignments may occur. It is your responsibility to attend class and generally be aware of what content is being covered.
Scheduled Tutorials

There will be several tutorials spread throughout the term to provide guidance and help you fully understand the material and techniques. They are not mandatory, but they are highly recommended. Tutorials T1 through T7 will help with basic material while tutorials T8 through T10 show how to combine these concepts with more advanced features using larger and more complex examples. Tutorials M and F will help you review for the midterm and final examinations, respectively.

<table>
<thead>
<tr>
<th>Tutorial</th>
<th>Title</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>How to Use the Laboratory Computers</td>
<td>Creating your SOCS account, Login/logout procedures, The desktop environment and applications, Accessing and using myCourses, RText and the JDK: creating, loading, editing, saving, compiling, and running programs</td>
</tr>
<tr>
<td>T2</td>
<td>Simple Java Programming</td>
<td>Syntactic and semantic errors, Variables and primitive data types, Expressions and the assignment statement, Primitive type conversions, Writing simple programs, Assignment submission via myCourses</td>
</tr>
<tr>
<td>T3</td>
<td>Conditional Programming</td>
<td>Boolean expressions, Conditional programming constructs in Java</td>
</tr>
<tr>
<td>T4</td>
<td>Iterative Programming</td>
<td>Iterative programming constructs in Java, Basic debugging techniques</td>
</tr>
<tr>
<td>T5</td>
<td>Methods</td>
<td>Defining and calling methods, Using the Eclipse IDE</td>
</tr>
<tr>
<td>T6</td>
<td>Arrays</td>
<td>Allocating and processing arrays, Reference types and aliases</td>
</tr>
<tr>
<td>M</td>
<td>Midterm Review</td>
<td></td>
</tr>
<tr>
<td>T7</td>
<td>Records</td>
<td>Defining record types, Creating and using records, Using the Eclipse debugger</td>
</tr>
<tr>
<td>T8</td>
<td>Using Classes and Objects</td>
<td>Object-orientation concepts, Creating and using objects, Java Standard Library classes</td>
</tr>
<tr>
<td>T9</td>
<td>Creating Classes and Objects</td>
<td>Creating your own classes, Instance variables and methods, Constructors</td>
</tr>
<tr>
<td>T10</td>
<td>Files and Recursion</td>
<td>Reading from and writing to files, Basic exception handling, Recursive methods</td>
</tr>
<tr>
<td>F</td>
<td>Final Exam Review</td>
<td></td>
</tr>
</tbody>
</table>
## Schedule of Topics

<table>
<thead>
<tr>
<th>Date</th>
<th>Material</th>
<th>Readings</th>
<th>Events</th>
</tr>
</thead>
</table>
| **Procedural Programming Basics** | **Week 1** | *Introduction*  
What is programming  
How does a computer work? | 1.1–1.3 | Placement  
Quiz  
A1 released |
| | **Week 2** | *The basic building blocks*  
Variables and primitive data types  
Expressions and the assignment statement  
Data conversion  
Basic input and output | 1.6–1.9, 2.1–2.15 | Tutorial T1  
A1 released |
| | **Week 3** | *Conditional programming*  
Boolean expressions  
*if* and *if-else* statements | 3.1–3.3, 3.7 | Tutorial T2  
A1 due  
A2 released |
| | **Week 4** | *Programming with iterations*  
*while* and *for* statements | 4.1–4.2, 4.4–4.6, 4.8 | Tutorial T3 |
| | **Week 5** | *Methods*  
Defining and calling methods  
Parameters and return values  
Method libraries | 5.1–5.11 | Tutorial T4  
A2 due  
A3 released |
| | **Week 6** | *Arrays*  
Allocating and processing arrays  
Reference types and aliases | 6.1–6.5, 6.10–6.11 | Tutorial T5 |
| | **Week 7** | *Records*  
Defining record types  
Fields and static variables  
Creating and using records | 7.1–7.4, 7.6, 7.9–7.10 | Tutorial T6  
A3 due |
| | **Week 8** | *Review* | | Tutorial M  
Midterm  
A4 released |
| | **Week 9** | *Using classes and objects*  
Object-orientation concepts: classes and objects  
Creating and using objects  
Java Standard Library classes | 7.5, 8.1–8.2, 8.5, 10.9–10.10 | Tutorial T7 |
| | **Week 10** | *Building your own classes*  
Anatomy of a class  
Instance variables and methods, constructors  
Encapsulation  
Class relationships | 7.7–7.8, 9.1–9.10 | Tutorial T8  
A4 due  
A5 released |
| | **Week 11** | *Using files*  
Files and file paths  
I/O streams  
Reading from and writing to files  
Basic exception handling | 8.6–8.7, 18.1–18.5, 18.7 | Tutorial T9 |
| | **Week 12** | *Recursion*  
Recursive thinking and programming | 20.1–20.4 | Tutorial T10 |
| | **Week 13** | *Advanced topics (if time permits) and review* | | Tutorial F  
A5 due |