COMP-202A: Introduction to Computing 1

McGill University, Fall 2007

Course Details

Section 1	Instructor:	Mathieu Petitpas
	Office:	McConnell Engineering Building 104
	Office hours:	MF 13:30–14:30 (or by appointment)
	Contact info:	`mathieu.petitpas@mail.mcgill.ca`
	Room:	ENGTR 0100
	Class times:	MWF 11:35–12:25
Section 2	Instructor:	Clark Verbrugge (Course coordinator)
	Office:	McConnell Engineering Building 230
	Office hours:	WF 10:00–11:00
	Contact info:	`clump@cs.mcgill.ca`
	Room:	ENGTR 0100
	Class times:	TTh 13:05–14:25
Section 3	Instructor:	Ladan Mahabadi
	Office:	McConnell Engineering Building 104
	Office hours:	TTh 8:30–9:30
	Contact info:	`ladan.mahabadi@cs.mcgill.ca`
	Room:	RPHYS 118
	Class times:	MWF 12:35–13:25

Contacting Instructors and TAs

Post all your questions about assignments on the myCourses (WebCT Vista) 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-mailing facilities by McGill or an e-mail account you may have with any e-mail provider. myCourses is located at `http://www.mcgill.ca/mycourses/`.

Students are expected to monitor their McGill e-mail account, myCourses, and the course home page (`http://www.sable.mcgill.ca/~clump/comp202`) 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 (except for knowing how to use e-mail, browse the Web, etc.).

The course uses the Java programming language. As with natural languages, programming languages can be grouped; languages in the same group are conceptually quite similar, while languages from different groups follow quite different

paradigms. Java is an *object-oriented* language (as are C++ and many others). 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 could 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 recipe to solve 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, set-up and configure a computer, use specific software applications, design Web pages, nor deal with operating systems problems. However, the course does provide introductory labs that provide some help in this regard.

Course Pre-requisites and Textbooks

Pre-requisites:

- A CEGEP-level math course or equivalent. For students who did not attend CEGEP, any upper-level mathematics course is sufficient.

Course textbook:

- *Java Software Solutions: Foundations of Program Design*, 5th Edition. John Lewis and William Loftus. Addison-Wesley. 2006. ISBN: 0321409493.

  You may purchase a copy of this textbook from the McGill bookstore. It is also on reserve in the Schulich Library.

Other references:

- *Java 5.0 Documentation.* You can browse or download this from the Sun website (`http://java.sun.com/j2se/1.5.0/docs/index.html`). Use Java 6.0 documentation if you use Java 6.0.

Course Grading System and Deadline Policy

Your final grade in the course is allocated as follows:

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

In exceptional situations students may write the supplemental exam. Ability to do so is not automatic though, and depends on your exact situation; contact the student affairs office for further information.

- **Supplemental Exam:** The supplemental exam represents 100% of your final grade.

There will be **5** assignments. Each of the first 4 assignments are worth 5% of your grade, while assignment 5 is worth 10% of your total 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 and all course work must represent your own personal efforts (see the section on Assignments and Plagiarism Policy 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%, etc. Assignments submitted more than 2 days after the deadline will not be accepted, nor marked. 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 a medical note.**

Campus Computer Labs

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 they are registered in. These facilities are located on the third floor of the 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 and in room 3120 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 the Trottier building before 7:00 and after 21:00 on weekdays, 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 the Trottier building using your McGill ID outside the building's opening hours, contact the SOCS Systems Staff, either by e-mail (preferred) at `help@cs.mcgill.ca`, by phone at (514) 398-7087, or in person in McConnell engineering building 209N, 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 the 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.

All computers in the SOCS laboratory facilities run Gentoo GNU/Linux or FreeBSD, which are Unix-like operating systems. Members of the systems staff will hold Unix tutorials at the beginning of the semester for those who are new to Unix. Information regarding these tutorials will be given during the first lectures. If you are only familiar with a Windows (95/98/Me/2000/XP/Vista) environment, it is strongly recommended you attend these tutorials.

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

Other computer laboratory facilities: The information in this subsection is to be used as a general guideline only. We suggest that students contact the work area of their choice to enquire about the hours and to obtain any further information needed. Most facilities are available to all McGill students but there are locations with restricted usage permitting access only to those students within the faculty or department indicated. For example, students in the Faculty of Science may want to use the lab in Burnside Hall, and Engineering Students may want to use the Engineering Labs.

A usage fee is required by some facilities and it may be a little more for students not from the faculty or department indicated. A student may pay the fee on a per day basis or become a member, that is, obtain a pass by paying for a full semester or academic year. Becoming a member may have certain advantages such as, for instance, computer reservation privileges. Most areas provide printing, but may have a charge per page.

Information about other labs can be obtained via the Computing Center's web pages. Please visit the labs for time changes.

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). We highly recommend that you use an integrated development environment (IDE). It provides an editor so that you can type your program as well as commands to compile and run it. During the lectures and tutorials, we will be using DrJava, a simple but easy-to-use IDE. You can also simply use any text editor of your choice, and then use the JDK tools directly to compile and run the programs.

The JDK is pre-installed on the SOCS lab machines, as is DrJava. On your own machine or in other labs, you may need to install these yourself or have the lab administrator(s) do so for you. Installing both of these is extremely easy.

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. You will be able to download a copy of JDK from the following Web site: `http://java.sun.com/javase/downloads`. You need version 5.0 or 6.0. There is no time limit on the free use of the JDK. You should install the JDK before any IDE.
- **Optional:** DrJava. We suggest you use DrJava (for Windows/Mac/Unix) as a very simple IDE for Java development, free and with a very small learning curve. You may download a personal copy of DrJava from the following WebSite: `http://drjava.org`. You should install this environment only after you have installed the JDK, as this will avoid several configuration headaches.

Teaching Assistants

Each TA will be available for 1 office hour per week, on the third floor of the Trottier building, to help you with your assignments and answer questions about the course material. TA office hours will be posted on the course home page when available.

Submitting Assignments

Each assignment will contain directions on what to submit and how to submit it. This course will predominantly be using myCourses for assignment submission and grading, and instructors and TAs will discuss how to use it in class and during labs. myCourses will also 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 and midterm. It is your responsibility to check that the marks are correct and to notify your instructor of any errors or missing marks.

Assignments and 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 `http://www.mcgill.ca/integrity/` for more information).

Politique officielle: L'université McGill attache une haute importance à l'honnêteté académique. Il incombe par conséquent à tous les étudiants de comprendre ce que l'on entend par tricherie, plagiat et autres infractions académiques, ainsi que les conséquences que peuvent avoir de telles actions, selon le Code de conduite de l'étudiant et des procédures disciplinaires (pour de plus amples renseignements, veuillez consulter le site `www.mcgill.ca/integrity`).

You must include your name and student number at the top of each program or module that you have implemented. By doing so you are certifying that the work is entirely your own, and represent only the result of your own efforts.

Work submitted for this course must represent your own efforts. Assignments **must** be done **individually**; you may **not** work in groups. Do not rely on friends or tutors to do your work for you. You may **not** copy another person's work in any manner (electronically or otherwise). Furthermore, you must **not** give a copy of your work to another person.

Work found not to be the result of your own efforts will receive a grade of 0. We will be using automated plagiarism 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 plagiarism by an instructor or TA; we will not accuse anyone of plagiarism based solely on the output of these tools). You may also be asked to present and explain your program to an instructor or TA at any time.

Students who require assistance with their assignments should see a TA or their instructors during their office hours. If you have only partially finished an assignment, submit it anyways for partial credit. You must document what works and what does not work in your program, and hand it in. Students who put their name on programs or modules that are not entirely their own work may also be referred to the appropriate Associate Dean who will assess the need for further disciplinary action.

Course Content

Note that the course schedule below is preliminary. Changes in content, reading material, and times for labs and assignments may occur. Please do not bother TAs or instructors with questions as to what material was covered or not; it is your responsibility to attend class and generally be aware of what content is being covered.

Scheduled Tutorials

There will be several tutorials (or "labs") 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 1 through 4 will help with basic material while tutorials 5, 7, and 8 show how to combine these concepts with more advanced features using larger and more complex examples. Tutorials 6 and 9 will help you review for the midterm and final examinations, respectively.

Lab	Title	Contents
1	How to Use the Lab Computers	Creating your SOCS account Login/logout procedures Available applications Overview of Unix commands Accessing myCourses Using DrJava
2	Simple Java Programming	Expressions and data types Conversions Java libraries
3	Control Flow	Conditional programming and iteration constructs in Java
4	Classes and Objects	Creating your own classes Instance variables and methods Constructors Class relationships
5	Arrays	Single and multidimensional arrays `ArrayList`s
6	Midterm Review	
7	Working With Files	Reading from and writing to files Handling exceptions
8	Recursion	Recursion and run-time stack examples
9	Final Exam Review	

Approximate Schedule of Topics

	Date	Material	Readings	Events
Introduction	Week 1	*Introduction* What is programming How does a computer work? Our first programs	1.2, 1.4–1.5 2.2, 2.6 (Note: The book is more detailed than the lectures)	Placement Quiz
	Week 2	*Java in 1 week* Basics of programming Variables and assignment Control-flow Object-orientated programming: the concept of a *class* Example: Implementing a calculator	2.2–2.4, 2.6 5.1–5.2, 5.5 (Note: The book is more detailed than the lectures)	Tutorial 1
Fundamentals	Week 3	*The basic building blocks* More data types Advanced expressions Data conversion Using classes and objects Class libraries	2.1–2.6 3.1–3.6 (Note: Much more detail on material presented earlier)	Tutorial 2 A1 due
	Week 4	*Conditional programming* `if` and `switch`	5.1–5.4 (Note: chapter 4 skipped)	
	Week 5	*Programming with iterations* `while`, `do`, and `for`	5.5–5.8	Tutorial 3 A2 due
Methods & Objects	Week 6	*Building your own classes* Anatomy of a class Constructors and methods	4.1–4.5 6.1–6.2	
	Week 7	*Using objects* Class relationships Objects as parameters Overloading Encapsulation	6.4, 6.7–6.9	Tutorial 4 A3 due
In-depth	Week 8	*Programming with arrays* Single and multi-dimensional The `ArrayList` Linear search and selection sort	7.1–7.3, 7.6–7.7	Tutorial 5
	Week 9	*Review*		Tutorial 6 Midterm
	Week 10	*Static artifacts and the command line* Static variables and methods Command-line arguments	6.3 7.4	A4 due
	Week 11	*Using files* What are files? I/O streams and exceptions Writing files	10.1–10.3, 10.6	Tutorial 7
	Week 12	*Advanced iteration* Recursion and run-time stacks	11.1–11.3	Tutorial 8
	Week 13	*Linked lists (if time permits) and review* Aliasing and pointers Self-referencing objects Review of course material	12.1–12.3	Tutorial 9 A5 due