Computer Science 2SC3

Imperative Programming and Basic Data Structures

and

Software Engineering 2S03

Principles of Programming

Fall 2008

Course Outline

Revised: 19 September 2008

Note: This course outline contains important information that may affect your grade. You should retain it throughout the semester as you will be assumed to be familiar with the rules specified in this document.

Instructor

Dr. William M. Farmer
Office: ITB 163
Extension: 27039
E-mail: `wmfarmer@mcmaster.ca`
Web: `http://imps.mcmaster.ca/wmfarmer/`
Office hours: by appointment

Teaching Assistants

Pouya Larjani	graduate	`larjanp@mcmaster.ca`
Hong Ni	graduate	`nih2@univmail.cis.mcmaster.ca`
Jessica Cao	undergraduate	`jessica.cao@gmail.com`
Rebecca Dreezer	undergraduate	`dreezerj@mcmaster.ca`

Course Web Site

`http://imps.mcmaster.ca/courses/SE-2S03-08/`

Students should be aware that, when they access the electronic components of this course, private information such as first and last names, user names for the McMaster e-mail accounts, and program affiliation may become apparent to all other students in the same course. The available information is dependent on the technology used. Continuation in this course will be deemed consent to this disclosure. If you have any questions or concerns about such disclosure please discuss this with the course instructor.

Schedule

Lectures:	MWR	10:30–11:20	JHE 264
Tutorial 1:	F	13:30–14:20	BSB 119
Tutorial 2:	M	14:30–15:20	BSB 137

CS 2SC3 Calendar Description

"Disciplined programming in the C language; problem decomposition; iteration and recursion; dynamic memory allocation; design, use and implementation of elementary fixed-size and dynamic data structures."

SE 2S03 Calendar Description

"Fundamental concepts of imperative programming languages; (Assertion, Assignment; Control flow, Iteration, recursion, exceptions); Data representations; Basic concepts of operating systems; Composing and analyzing small programs."

Mission

Imperative programming is the oldest, and still one of the most popular, programming paradigms. The mission of the course is to teach students the underlying principles of imperative programming and basic data structures. By the end of the course the student should:

1. Understand the imperative programming paradigm and how it is different from other programming paradigms, particularly object-oriented programming and functional programming.

2. Be able to program in the imperative style in C, a traditional, intermediate-level, single-paradigm programming language.

3. Be able to program in the imperative style in Objective Caml (OCaml), an advanced, high-level, multiparadigm programming language.

4. Have a working knowledge of Unix-style operating systems, command-line interfaces, and version control systems such as Subversion.

5. Have a sense of what are the professional responsibilities of computing professionals such as computer scientists, software engineers, and mechatronics engineers.

Required Text

H. Deitel and P. Deitel, *C How to Program (5th Edition)*, Prentice Hall, 2006. ISBN: 978-0132404167.

Work Plan

There will be lectures, tutorials, programming exercises, quizzes, a midterm test, and a final exam. The lectures will be given by the instructor during regular class sessions. The tutorials will usually be conducted by the teaching assistants during the regular tutorial sessions. There will be a written, 10-minute quiz given each Thursday at the beginning of the lecture session.

The programming exercises will be done by the students during their own time. There will be five exercises each consisting of two parts. The first part will be performed using OCaml and the second using C. The exercises will be submitted using Subversion. Details concerning the exercises will be provided later.

The midterm test will be held on Thursday, October 30, 2008 during the regular lecture time at 10:30–11:20. The final exam will be 3 hours long. It will take place on the date scheduled by the University.

The class will pick a *class representative* who will serve as a liaison between the students and the instructor.

Log Book

Each student is expected to keep a detailed, up-to-date log book that records all the steps performed on the programming exercises. Sources of information, consultations with instructors, teaching assistants, and fellow students; successful and failed experiments; discovered errors; and lessons learned should be recorded. The log book should be written as a text file, and the entries in the log book should be listed chronologically with dates and times. *A copy of the student's log book must be submitted with each programming exercise that is submitted.*

Academic Dishonesty

You are expected to exhibit honesty and use ethical behavior in all aspects of the learning process. Academic credentials you earn are rooted in principles of honesty and academic integrity.

Academic dishonesty is to knowingly act or fail to act in a way that results or could result in unearned academic credit or advantage. This behavior can result in serious consequences, e.g., the grade of zero on an assignment, loss of credit with a notation on the transcript (notation reads: "Grade of F assigned for academic dishonesty"), and/or suspension or expulsion from the university.

It is your responsibility to understand what constitutes academic dishonesty. For information on the various types of academic dishonesty please refer to the Academic Integrity Policy, located at

`http://www.mcmaster.ca/academicintegrity/`

The following illustrates only three forms of academic dishonesty:

1. Plagiarism, e.g., the submission of work that is not one's own or for which other credit has been obtained.
2. Improper collaboration in group work.
3. Copying or using unauthorized aids in tests and examinations.

Your work must be your own. Plagiarism and copying will not be tolerated! If it is discovered that you plagiarized or copied, it will be considered as academic dishonesty.

Students may be asked to defend their written work orally.

Other Policy Statements

1. Significant study and reading outside of class is required.
2. *Students are required to attend the lectures and tutorials.* Attendance will be recorded via an attendance sheet passed around the class. Absences will be excused only in highly exceptional cases.
3. The student is expected to ask questions during class.
4. You may want to discuss the assignments with your fellow students. *If you do that, you must record a summary of your discussions in your log book including a list of all those with whom you had discussions and a description of what information you received.* It is part of your professional responsibility to give credit to all who have contributed to your work.
5. A student may use his or her texts and notes during the midterm test and final exam but not during the quizzes.
6. Programming exercises may not be submitted late and the quizzes and midterm test may not be taken later without *prior* approval from the instructor.
7. Calculators and electronic devices are *not* permitted during the quizzes, midterm test, and final exam.
8. The instructor reserves the right to require a deferred final exam to be oral.
9. The Faculty of Engineering is concerned with ensuring an environment that is free of all adverse discrimination. If there is a problem, that cannot be resolved by discussion among the persons concerned, individuals are reminded that they should contact their Department Chair and the Human Rights and Equity Services (HRES) office as soon as possible.

10. Suggestions on how to improve the course and the instructor's teaching methods are always welcomed.

Marking Scheme

The course grade will be based on the student's performance on the quizzes, programming exercises, midterm test, and final exam as follows:

Quizzes (11)	20%
Programming exercises (5)	20%
Midterm test	20%
Final exam	40%
Total	**100%**

Notes:

1. A student's final score will be reduced by one half point for each missed lecture (there is no penalty for the first *six* missed lectures).
2. The instructor reserves the right to adjust the marks for a quiz, programming exercise, midterm test, or final exam by increasing or decreasing every score by a fixed number of points.

Syllabus

00 Preliminaries

01 Programming Languages

02 Simple Procedures

03 Control Structures

04 Records and Arrays

05 Linked Data Structures

06 Advanced Procedures