

Course Outline

Introduction to Computer Science COMP 250

(Winter 2011 MWF 4:35-5:25 TROTTIER 100)

Instructor: Professor Michael Langer
Office: McConnell Engineering 329
Tel: 398-3740
Email: langer@cim.mcgill.ca
Course Web Page: <http://www.cim.mcgill.ca/~langer/250.html>
Office Hours: after class (in Trottier)

Teaching Assistants (T.A.): See course web page for this information.

Introduction

This course is an introduction to several core topics in computer science, namely data structures and algorithms, and object oriented design.

For the data structures and algorithms component, you will learn about lists, stacks, heaps, trees and graphs and some of the basic algorithms that use these data structures such as searching and sorting. We will also learn how to analyze such algorithms in terms of the amount of computation that is required to carry them out. These analysis tools will be used heavily in subsequent data structures and algorithms courses (COMP 251, 360) and are a central part of computer science.

A second significant part of the course is a more advanced look at object oriented design that goes beyond the individual classes and objects that you learned about in COMP 202. You will learn how classes can be organized into hierarchies, and how variables and methods defined in the classes of the hierarchy are related to each other. These relationships, known as inheritance and polymorphism, will be developed more fully in subsequent courses that follow the programming languages (COMP 302) and software engineering stream.

Prerequisites

The prerequisite for this course is COMP 202 *Introduction to Computing 1* or any one semester course on programming in a high level language such as C or Java. The majority of students in COMP 250 have taken COMP 202 (or equivalent), so will know the basics of Java already.

If you have not taken COMP 202 (or equivalent), you should see the COMP 202 web page for the Fall 2010 semester to make sure you have the appropriate background for COMP 250. You should have some experience with programming in a high level language and you should have written simple programs that use variables, expressions, conditional statements (if-then-else, switch), loops (while, for), arrays, pointers (references), I/O (keyboard inputs or reading from a file, writing to console or to a file). Moreover, COMP 202 uses Java which is an object oriented language, and so students who have taken 202 will also be familiar

with basic concepts in object oriented programming (OOP). Those of you who have only programmed in C (but not C++) will be unfamiliar with the basic concepts of classes and objects. This applies, in particular, to Engineering students who have taken COMP 208. If you want to take COMP 250, *you will need to do some work in the first few weeks of the semester to catch up*. In particular, I strongly suggest that you:

- speak to me (Prof. Langer)
- read through the lecture slides from COMP 202 and make sure you are aware what was covered in that course <http://www.cs.mcgill.ca/~cs202>.
- sign out from the library or buy a basic book on Java programming and read the parts of it that are covered in COMP 202. Two very good introductory texts for Java are:
 - D. Liang, “Introduction to Java Programming,” Brief Edition, Prentice Hall.
 - Lewis and Loftus, “Java software solutions,” Addison-Wesley.

Any recent edition (last five years) should be fine.

- There is also a good free online Java book “How to think like a computer scientist”. What you need from COMP 202 can be found in Chapters 1-10 of that book. (Chapters 11-19 cover many of the topics that you will see in COMP 250).

Lecture Notes

There is no required textbook for the course. Instead, a complete set of lecture notes will be made available on the public course web page, as will the lecture slides, and links to other material including Exercises.

WebCT

There is a WebCT discussion board which is organized into various topics such as lectures and various assignments. Please use this discussion board to clarify any problems you may have. The instructor would very much appreciate it if you would help each other by answering questions on the discussion boards. In the same vein, please do not email the instructor or T.A.s with technical questions about the course material. By posting these questions on WebCT, everyone can benefit from the answers.

Topics Covered in Course

- Preliminaries (2 lectures)
 - familiar algorithms (addition and multiplication), binary number representations
- Linear Data Structures (6 lectures)

- arrays, linked lists, stacks, queues (ADTs)
- insertion sort
- Recursion (3 lectures)
 - factorial, power, fibonacci, binary search, mergesort
- Analysis of Algorithms (5 lectures)
 - running time (big O, big Theta, big Omega)
 - recurrences (factorial, Tower of Hanoi, decimal to binary, fibonacci, mergesort)
- Non-linear Data Structures (12 lectures)
 - trees, binary trees, binary search trees, tree traversal
 - priority queues, heaps (building a heap - slow vs. fast) heapsort
 - maps, hashing
 - graph, graph traversal, depth vs. breadth first search
- Object Oriented Design in Java (6 lectures)
 - inheritance, polymorphism, interfaces, abstract classes, UML, casting, generics

Evaluation

Your final grade will be calculated using the following percentage breakdown.

- **Four Assignments (40 % total)**
 - A1 to be posted in mid-late January
 - A2 to be posted in early February
 - A3 to be posted in early March (after Study Break)
 - A4 to be posted in late March

You will be given at least 10 days to complete each assignment.

- **Two in-class Quizzes (20 % = 2 × 10 %)**
 - Quiz 1 in early February
 - Quiz 2 in early March
- **Final Exam (40 % or 60 %)** held during Final Examination Period.

If your percentage grade on the Final Exam is greater than your percentage grade on the two Quizzes, then your Final Exam grade will count for 60% of your final grade. This policy is intended to allow some flexibility, for example, if you are unable to attend one of the quizzes because you are sick or you have some other obligation.

In accord 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.

Policy on collaboration

We greatly encourage you to discuss the assignment problems with each other. However, this discussion must not go so far that you are revealing the solutions to each other. And it certainly should not go so far that you are sharing code. We realize there is sometimes a fine line between giving hints and revealing solutions, so we ask you try to follow a simple guideline: any discussion you have about an assignment should be *open* in the sense that you would be 100% comfortable if anyone else *including the instructor* were listening in.

McGill policy on academic integrity

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/students/srr/honest/> for more information