CMPT225: Data Structures and Programming
Instructor: Greg Mori
Spring 2012 SYLLABUS

Overview

This course will explore ideas of data and program organization that allow complex tasks to be solved in simple and elegant ways. In order to manage the complexity of programs, we will look at program design and organization ideas such as abstract data types, data structures and object-oriented programming. We will gain practical experience of these ideas by considering their implementations in Java and C++. Once you have successfully completed this course, you will have acquired:

- knowledge of data structures such as lists, stacks, queues, trees, hash tables, heaps and priority queues;
- knowledge of algorithms commonly used in computing science such as searching and sorting, as well as time and space efficiency analysis;
- knowledge of abstract data types and associated concepts such as abstraction, encapsulation and information hiding;
- knowledge of external data storage and sorting;
- object-oriented and modular design and programming (OOD/OOP) skills;
- recursive programming skills;

Administrivia

Lectures: Mondays 14:30-16:20 in AQ 3153, Wednesdays 14:30-15:20 in WMC 3260

Labs: Tuesdays in CSIL (ASB 9838S)

Greg's office hours: Tuesdays 14:30-15:30 and Wednesdays 15:30-16:30 in TASC1 8007

TAs: Hossein Hajimirsadeghi (`hosseinh@sfu.ca`), Tian Lan (`tla58@sfu.ca`), Nataliya Shapovalova (`nshapova@sfu.ca`)

TA office hours: Thursdays 11am-1pm, Fridays 11am-12pm in CSIL (ASB9838_TA_1)

Course website

`http://www.cs.sfu.ca/~mori/courses/cmpt225`

Prerequisites

CMPT 125, 126, or 128. MACM 101.

The course assumes that the students are already familiar with the following basic software development concepts, and basic programming in Java or C++ (you should know both by the end):

- Principles of software design, implementation, and testing;
- Fundamental data types - numbers, characters, boolean;
- Classes, and objects;
- Class fields, and methods, parameter passing, references;
- Control structures - sequence, repetition, decision;
- Simple input and output;
- Data structures - arrays, strings;
- Algorithms - searching, simple sorting (Bubble, Selection and Insertion sort), recursion;

Lecture Schedule (subject to change)

Jan. 9: Introduction

Jan. 11, 16, 18: Stacks and Queues (Ch. 5, 7, 8)

Jan. 23, 25: Object Oriented Programming and Design (Ch. 2, 4)

Jan. 30, Feb. 1: Recursion (Ch. 3)

Feb. 6, 8: O Notation (Ch. 10)

Feb. 13-17: **Reading week, no classes**

Feb. 20, 22: Sorting (Ch. 10)

Feb. 27, 29: Trees (Ch. 11)

Mar. 5: Balanced trees

Mar. 7: Written midterm (in class)

Mar. 12: Programming midterm (in CSIL)

Mar 14: Balanced trees, midterm post-mortem

Mar. 19, 21: Hashing (Ch. 13)

Mar. 26, 28: Heaps and Priority Queues (Ch. 12)

Apr. 2, 4: External Storage (Ch. 15)

Apr. 9: **Easter Monday, no class**

Apr. 11: Final review

Grading

Evaluation will be based on individual assignments and labs, and midterm and final exams.

- 12% Assignments
- 8% Labs
- 15% Written midterm
- 20% Programming midterm
- 45% Final Exam

Students must obtain a passing grade on the programming midterm in order to receive a clear pass (C or better) in the course.

Assignments

Assignment dates:

- Assignment 1 due February 7 worth 3%
- Assignment 2 due February 29 worth 3%
- Assignment 3 due March 16 worth 3%
- Assignment 4 due April 10 worth 3%

All assignments are to be done individually. Assignments 1, 2, and 3 will be done in C++. Assignment 4 will be done in Java.

Important Note: The university policy on academic dishonesty (cheating) will be taken very seriously in this course. You may not provide or use any solution, in whole or in part, to or by another person in the assignments.

You are encouraged to discuss the concepts involved in the assignments with other students. If you are in doubt as to what constitutes acceptable discussion, please ask! Further, please take advantage of office hours offered by the instructor and the TA if you are having difficulties with assignments.

DO NOT:

- Give/receive code to/from other people
- Use Google to find solutions for assignment

DO:

- Meet with other students to discuss assignment away from a computer (do not to take any notes during such meetings, and re-work assignment on your own)
- Assist other students in deciphering compiler error messages, providing debugging hints, helping with linux/IDE commands
- Use online resources (e.g. Wikipedia) to understand the concepts needed to solve the assignment

Assignment Late policy

Students will be permitted 4 grace days to use at their discretion over the trimester. Late days are counted from the time an assignment is due, rounded up to the nearest whole day. For example, if an assignment is due on Friday at 3:30am, and is submitted on Saturday at 5pm, 2 grace days will have been used.

IMPORTANT: Other than the 4 grace days, late assignments will not be accepted, and will receive zero marks.

Lab Schedule (not so subject to change)

Jan. 10: Hello world (C++)

Jan. 17: Dynamic arrays (C++)

Jan. 24: A Simple Class (C++)

Jan. 31: Copy Construtors and Destructors (C++)

Feb. 7: **No assigned lab – TA office hours in labs. Labs 0-3 can be submitted for 1/2 credit.**

Feb. 14: **No lab – Reading Week**

Feb. 21: Overloaded operators (C++)

Feb. 28: Templates (C++)

Mar. 6: **No assigned lab – TA office hours in labs. Labs 4-5 can be submitted for 1/2 credit.**

Mar. 13: **No assigned lab – TA office hours in labs.**

Mar. 20: **No assigned lab – TA office hours in labs.**

Mar. 27: Mergesort (Java)

Apr. 3: Heaps (Java)

Apr. 10: **No assigned lab – TA office hours in labs. Labs 6-7 can be submitted for 1/2 credit.**

Labs will be held in CSIL (ASB 9838S, linux). Please check in advance that your SFU Campus Computing account is active so that you will be able to log in and perform the labs.

Each lab is graded out of 1 mark. Labs will be made available before the day of the lab, via the course webpage and announcement to the mailing list. Full credit (1 mark) will be given if the lab is completed by the end of your lab session. If you fail to complete a lab in time, you can see a TA during one of the make-up sessions noted above and receive 1/2 credit.

Software

We will use Linux with the command-line interface and a basic text editor for the C++ labs and programming midterm. It is recommended that you do the same for the C++ assignments. However, you may use any operating system and development environment you choose as long as the code conforms to specifications provided with the assignment to compile and execute. We will use the Eclipse IDE for the Java labs, and you may choose your favourite OS/IDE for the Java assignment.

CourSys

This course will use CourSys, an online grade-tracking system, for recording grades. The URL of CourSys is `https://courses.cs.sfu.ca/`.

CourSys uses your Campus Computing id and password (CAS authentication).

Exams

There will be one written midterm, one C++ programming midterm, and a final examination at the end of the trimester. Each exam will be closed-book.

The final examination has been scheduled for Wednesday, April 18, from 12:00 to 15:00, location TBA. It will cover the entire course.

Do not write an examination if there is a medical factor which might impair your performance. If you have been unable to write an exam due to illness, you must provide a Health Care Provider Statement, available from the Registrar's Forms page: `http://students.sfu.ca/forms.html` under "Appeals". Any appeals regarding medical excuses for assignments or labs must also be accompanied by this form.

Policies and procedures regarding Grading Practices and Grade Appeals can be found @ `http://www.sfu.ca/policies/teaching/t20-01.htm`. Policies and procedures regarding Final Examinations can be found @ `http://www.sfu.ca/policies/teaching/t20-02.htm`.

Textbooks

There are no required textbooks for this course. The lecture slides will be made available, and the books below provide additional descriptions of the material.

REFERENCE:

• Data Abstraction & Problem Solving with C++ - Walls and Mirrors, F. M. Carrano, Addison Wesley, 5th edition, 2007

• Data Abstraction & Problem Solving with Java - Walls and Mirrors, F. M. Carrano & J. J. Prichard, Addison Wesley, 2nd edition, 2006.

These two books essentially cover the same material (hence the similar titles), but illustrate the material in different programming languages. I would recommend buying the version that uses the programming language that you are least familiar with.

RECOMMENDED:

• Introduction to Algorithms - 3rd Edition, T.H. Cormen, C.E. Leiserson, R.L. Rivest, C, Stein, MIT Press, 2009

• C++ for Java Programmers, Mark Allen Weiss, Prentice-Hall, 2003, 9780139194245, Recommended for former CMPT 125/126 students who have no past C/C++ experience.

See course webpage for links to online materials.

Academic Honesty

Academic Honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty are subject to disciplinary action by the School; serious infractions are dealt with in accordance with the Code of Academic Honesty (S10.01) (`http://www.sfu.ca/policies/Students/`). Students are encouraged to read the School's policy information (`http://www.cs.sfu.ca/undergrad/Policies/`).