Computer Science 331

Introduction to CPSC 331

Mike Jacobson

Department of Computer Science
University of Calgary

Lecture #1

Outline

1. Course Information
   - Contact Information
   - References
   - Assessment
2. Learning Goals
   - Programming by Contract
   - Algorithm Analysis and Testing
   - Java Implementation
3. Expected Background
4. How to Succeed
5. Reading Assignment #1

Course Information

Instructor: Mike Jacobson

- Phone: 210-9410
- email: `jacobs@cpsc.ucalgary.ca`
- URL: `http://pages.cpsc.ucalgary.ca/~jacobs/`
- Office hours: M 11:00-13:00 or by appointment *only*

Contact Times:

- Lectures: MWF 9–9:50 in CHC 119
- Tutorial Section #1: M/W 10:00-10:50 in CHE 202
- Tutorial Section #2: M/W 13:00-13:50 in EDC 152

First labs: Wednesday!

Textbook and Recommended Reference

Required Textbook:

- Elliot B. Koffman and Paul A. T. Wolfgang
  Objects, Abstraction, Data Structures, and Design using Java Version 5.0
  Wiley, 2005

Recommended Reference:

- Justin Zobel
  Writing for Computer Science
  Springer-Verlag, 2004

This writing reference will be useful in future courses too.

Other Resources

Course web site: *lots* of information here!

- Available from the instructor's home page
- Blackboard page should be used for assignment submission and access to grades

Lectures: students are expected to attend *all* classes

- Partial notes (outline) will be made available online
- Additional material on topics on course web site
- Even more material in textbook. Yes, there will be required reading in this course.

Tutorials: participation in these is expected too!

- Exercises will be posted on the web site ahead of time

Assessment

Components:

- 25% — five assignments (written and programming questions)
- 15% — term test 1 (Oct 15, 18-19:30, ICT 121)
- 15% — term test 2 (Nov 19, 18-19:30, ICT 121)
- 45% — final exam

Take note of term test dates/times: let me know of conflicts as soon as possible (no make up tests)

Submission procedures and guidelines:

- information available on course web site

NOTE: a grade of **C-** or better is required to use this course as a prerequisite for any course offered by Computer Science

Programming by Contract

Programming by Contract:

- A methodology for developing computer software
- Key idea: software developers should define and use *precise checkable* specifications for software components
- A useful approach when software is developed and maintained over a long period of time by a group whose members can change
- Many modern programming languages, including Java, include facilities to support this approach. You will learn about and use these in this course

Specifying and Implementing a Procedure

A **specification of requirements** for a procedure includes the following (along with the procedure's name and the names and types of its inputs), to define the *problem to be solved:*

- **Responsibilities:** Purpose of the procedure
- **Pre-Conditions:** Conditions assumed to be true on entry if the procedure is to execute successfully
- **Post-Conditions:** What the procedure guarantees on exit
- **Returns:** Type of value(s), if any, returned
- **Exceptions:** Description of exceptions that be generated and circumstances when they arise (Section 2.4)

See pages 18–19 of the textbook for more details.

Algorithms

An **Algorithm**

- is *a finite sequence of steps that solves some well-defined problem* (as defined in the textbook)
- is often given either by several paragraphs in carefully written English or using *pseudocode*. Such a description is (largely) "implementation independent"
- can be *implemented* as (part of) a program using some programming language

Note: This course will focus at least as much on *algorithms* as on the computer programs generated from them.

$\Longrightarrow$ CPSC 331 is *not a programming course.*

More About Algorithms

Many computer science applications rely on solutions to a small number of *fundamental problems*

Resource requirements and limitations may also be important — and may differ from application to application

Consequence: It is often useful to know about *several* algorithms for the same problem — because there will be situations in which each is a better choice than the others

In this course we will learn about algorithms for several fundamental problems, including *searching* and *sorting*

Abstract Data Types

As described in the textbook a **data type** is defined by

- Data values and their representation
- Operations defined on the data values and the implementation of these operations as executable statements

A specification of *requirements* for a data type is given by an **abstract data type (ADT)**

See pages 13–14 of the textbook for more about ADTs

Data Structures

A **data structure** provides a representation of the data values specified by an ADT

Together with **algorithms** for an ADT's operations, this provides an *implementation-independent* description of a data type

We will study several fundamental ADTs, along with data structures and algorithms for their operations, in this course

Implementation

Modern programming languages may include ways to specify and use *both* ADTs *and* data structures

In Java:

- an ADT is generally specified using an **interface** (p.14-17)
- a data structure (and the algorithms for the ADT operations) is specified using a **class** (and its **objects**) that *implements* the interface for the corresponding ADT

Algorithm Analysis

Correctness and **efficiency** of algorithms are both important!

In this course you will

- design and implement *tests* in order to look for errors and use the results of tests to debug your programs
- see numerous proofs of correctness of algorithms, and you will become familiar with the structure of a proof of correctness as a result
- learn ways to measure
  - the time an algorithm requires
  - the amount of storage space

In this course we will generally test *programs* but we will prove the correctness and efficiency of *algorithms*

Java Implementation

Assignments will involve both the algorithms and data structures discussed in lectures and Java programming. You will

- implement algorithms and data structures on your own
- use implementations in a standard Java library (the “Java Collections Framework”) to solve problems

Use the computer science undergraduate laboratory (1st floor, MS)

- you can pick up your account from the help desk (engineers, too!)

Java will *not* be taught (much) during the lectures. However, sources of help with Java include

- lots of material on the course web site, Appendix A
- tutorials, which will include more material about Java programming (some of the time)

Expected Background

Object-Oriented Programming Language Syntax:

- either C++ or Java should have been introduced in a prerequisite course
- see Java resources on the course web site or Chapter 3
- work through Tutorial Exercise #1 as soon as you can! It will be discussed in the first tutorial, this Wednesday

Recursion: (Sections 7.1, 7.2, 7.4)

- you should understand how recursive programs can be used to solve problems
- recursive **definitions** of various structures and properties will be used in this course as well

Expected Background: Other Areas

Discrete Mathematics:

- has numerous applications in CPSC 331 (especially proofs and analysis)

⟹ MATH 271 is now a prerequisite or co-requisite

Technical Reading and Writing:

- this course will include **reading assignments**
- your writing **will be assessed** in this course

How to Succeed

Prepare for and attend **lectures**

- obtain/read notes and other reading material ahead of time

Prepare for and attend **tutorials**

- read and work through exercises ahead of time
- the more you do *on your own* the better

Take **assignments** seriously

- start early (not last minute!)
- make sure that you understand what you are — and what you are *not* — allowed to do when working on these

Make use of my **office hours** if you need more help

Reading Assignment #1

Please Read:

- Text, Chapter 1
  Course introduction (with more details)
- Text, Chapter 3
  Object-Oriented Programming and Java
  (mostly review for students who passed CPSC 233)
- Tutorial 1 (course web page) — will be covered in the labs on Wednesday and Thursday

Please Browse Through the course web site (you may find things there that are helpful)