ELEC 278

Fundamentals of Information Structures and Software Engineering

Fall 2016

David Athersych, P.Eng.
da49@queensu.ca
david@cynosurecomputer.ca

Data Structures and Algorithms

- Probably the most important course to help understand how to effectively design good computer software.
- Computers used to be slow, memory used to be expensive – software had to be as small as possible and as fast as possible.
- Computers are much, much faster and memory is vast.
- BUT, user demands keep growing
- Still required to design code that will run efficiently and effectively – and will scale up.

Instructor

- Dave Athersych, B.Sc., M.Sc., P.Eng.
- EE graduate from Queen's (before there was CE)
- MSc from Queen's in Computer Science
- MBA studies
- Founded 3 computer-related businesses
- Currently run Cynosure Computer Technologies
- Real-time control, distributed processing, networking, renewable energy, energy control systems.
- Likes to sail (this is Kingston!)


34219

Data Structures and Algorithms - History

- 1950s – machine or assembly language. Roll everything yourself.
- Early 1960s - high level languages emerge – Fortran, Basic, Cobol. Arrays are language feature; Cobol has something akin to a structure. Variables have types.
- 1960s – lots more languages – all adding various data structures and/or specializing in a particular type of processing.
- OS/360 – first real operating system – coded in assembler.

History, continued

- Experiments in using high-level languages (like PL/1) to write system software (Multics)
- 1970s – Pascal, C, LISP, etc. Justification for development of Ada was that US Military used 6000 different languages.
- UNIX and C. Skunk-works project that moved OS research forward.
- 1980s – Object Oriented Programming – Smalltalk, Modula, etc. C++ could be listed (more later).
- 1990 – languages target a high-level base – browser.
- 2000 – C# competes with Java

History - Summary

- The hardest parts of designing computer software are:
  - Figuring out how to organize the data, and
  - Figuring out how to process huge quantities of data.
- Example:
  - Think of how to display data on a computer screen.
  - Now think of refreshing the screen at least 30 times a second, by showing the image that one would expect to see from the window of a moving vehicle.
  - Consider that there are two front windows and two side windows – and the people there expect to see a consistent view of the outside world.
- This course will help you understand the two hardest parts.

Some Terms

- Assembly language, machine language – basic instructions that processor understands
- Hard to see the process
- Hard to see the structure of data
- Extremely error prone

```
L26:
MOV    EAX, 12(ESP)
CMP    EAX, 10
BGT    L27
MOV    EAX, 16(ESP)
ADD    EAX, 42
MOV    16(ESP), EAX
MOV    EAX, 12(ESP)
INC    EAX
MOV    12(ESP), EAX
BR     L26

L27:
```

Terms

- Higher Level Language
- Includes:
  - Variable types – identify what memory used for
  - Means to describe collections of variables as a unit of data (structures)
  - Control structures – express standard ways to process data
  - Perhaps, control structures that deal with particular collections of data
- Some “high level” languages allow programmers a lot of latitude (C is a good example)
- Others enforce rules strictly.

Object Oriented Programming

- Software Engineering – encapsulate data structure ***and*** the allowed processing on that data structure
- Code reuse, localized modification,
- OOP can be done in assembly language
- Designers use a variety of languages that incorporate OOP principles – C++, C#, Java, etc.
- This course is not about using an OOP environment to develop code
- We will use C to develop fundamental data structures and algorithms – which will help you understand what has been built into OOP languages.

Are Software Development Skills Useful to Electrical Engineering?

- YES
- Every system you work on, every system you design – will have electronic hardware and some computer component
- Every design team will need some folks focussed more on developing the electronic side and some concerned more with the software side – BUT both sides need to be aware of what is going on in the other domain.
- Question – how many computers are in a modern car? In a coffee maker?

Course Objectives

- Become familiar with standard data structures and algorithms (DS&A)
- Learn techniques for comparing alternate DS&A for a particular problem.
- Gain experience implementing DS&A solutions
- At completion, you will be able to:
- Analyse a problem
- Choose an appropriate DS&A solution and justify your choice
- Implement your solution, based on having developed skills to do so.

Course Organization

- Classes: Stirling Hall – Lecture Room B
  - Monday 1630-1730
  - Wednesday 1530-1630
  - Friday 1430-1530
- Tutorial: Stirling Hall – Lecture Room C. Work through suggested problems; answer student questions
  - Tuesday 1530-1630
- Lab: BM213 – 6 Lab exercises through the semester – 2 lab periods to complete each one.
  - Thursday 1830-2130

Quizzes and Midterm

- Quizzes will be held during the tutorial period in weeks 3, 6, 9, and 12. Basically, short questions based on the contents of labs and lectures.
- Midterm test – planned for week 6 (subject to change).

Marking Scheme

- Four parts:
  - Labs – 15 marks (6 x 2.5 marks)
  - Quizzes – 20 marks (4 x 5 marks)
  - Midterm – 20 marks
  - Final - 45 marks
- Assignments:
  - There will be some assignments but they will not be marked. They will be discussed in tutorials and you can get help from the TAs and the instructor. Questions on the quizzes, the midterm and the final may come from assignment material.

How to Succeed in this Course

- Keep up with the material.
- Use the resources:
  - OnQ will have copies of class slides, lab exercises, and suggested readings.
  - The TAs will post office hours.
  - The instructor will be available before and after class, by email and by appointment.
  - Read the class slides before class and reread them afterwards
- When sitting in the class, be present.
- There is a lot to cover, but steady pace is the way to do it.

Course Administration

- The class needs a class representative.
- Volunteer(s)?
- The key responsibility is the administration of the class evaluation (week 9 or 10).
- You may be asked by the Department Chair to give an opinion about the class, and/or to attend one or two meetings of class reps.
- You may be asked by your colleagues to bring an issue to my attention.

C Programming