COMP2521 20T2 ◊ Course Introduction

- People and Website
- More about me ...
- Course Goals
- Course Context
- Revision (material from COMP1511)
- Data Structure Viewpoint
- COMP2521 Themes
- How does the course run?
- "Lectures"
- Tutes and Labs
- Quizzes
- Assignments
- Plagiarism
- Final Exam
- Special Consideration
- Supplementary Exams
- Course Assessment

COMP2521 20T2

Data Structures & Algorithms

❖ People and Website

Convenor: John Shepherd (jas@cse)

Course Admin: Kevin Luxa

Tutors: cast of thousands ...

Course Email: cs2521@cse.unsw.edu.au

Course Website: https://webcms3.cse.unsw.edu.au/COMP2521/20T2/

zID/zPass login is needed for access to most of ...

- course material (slides, videos, tutes, labs, assignments, etc.)
- comments/forums, quizzes, polls, group formation

❖ More about me ...

My home office (video central):

Other things: AFL ... CSE ... HYP ... IPA ... KDr

❖ Course Goals

COMP1511 ...

- gets you thinking like a *programmer*
- developing algorithmic solutions to problems
- expressing your solutions as C programs

COMP2521 ...

- gets you thinking like a *computer scientist*
- knowing fundamental techniques/structures
- able to reason about applicability/effectiveness
- able to analyse behaviour/correctness of programs
- expressing your solutions as (larger) C programs

❖ COMP1511 vs COMP2521

COMP1511 ...

❖ ... COMP1511 vs COMP2521

COMP2521 ...

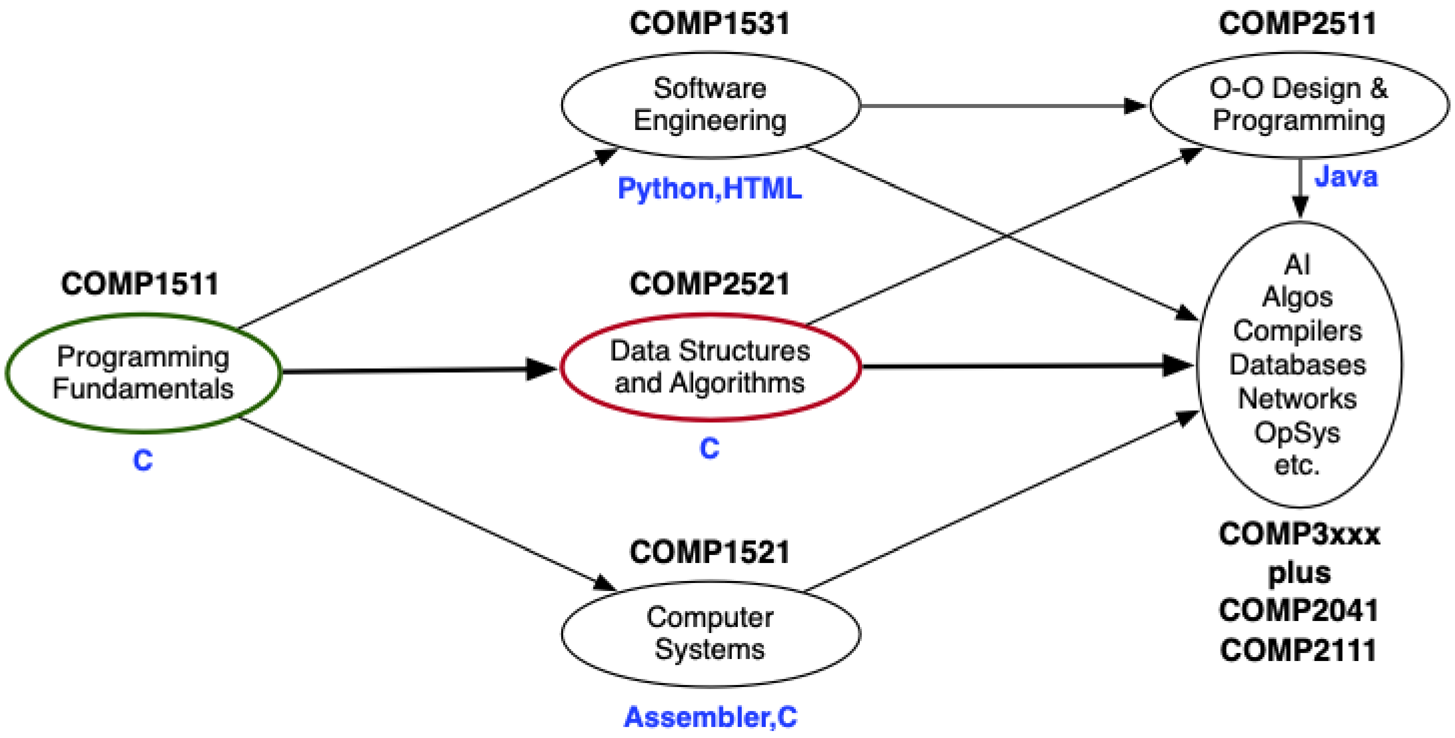
❖ Thinking like a Scientist

How to think like a (natural) scientist ...

observe → hypothesize → experiment → analyse → repeat

(In fact, the above process is precisely what we do for debugging)

❖ Course Context



❖ Pre-conditions

At the *start* of this course you should be able to:

- produce a correct C program from a specification
- understand the state-based model of computation (variables, assignment, addresses, parameters, scope)
- use fundamental C data structures (`char`, `int`, `float`, arrays, structs, pointers, linked lists)
- use fundamental control structures (`if`, `while`)
- implement abstraction via function declarations, ADTs
- implement a C program as a collection of `.c` and `.h` files
- fix simple bugs in incorrect programs

❖ Revision (material from COMP1511)

Important: Make sure you understand the following topics:

- Structs (see videos on Structs)
- Pointers (see videos on Pointers)
- Malloc (see videos on Malloc)
- Linked Lists (see videos on Linked Lists)

The above are used *extensively* in COMP2521

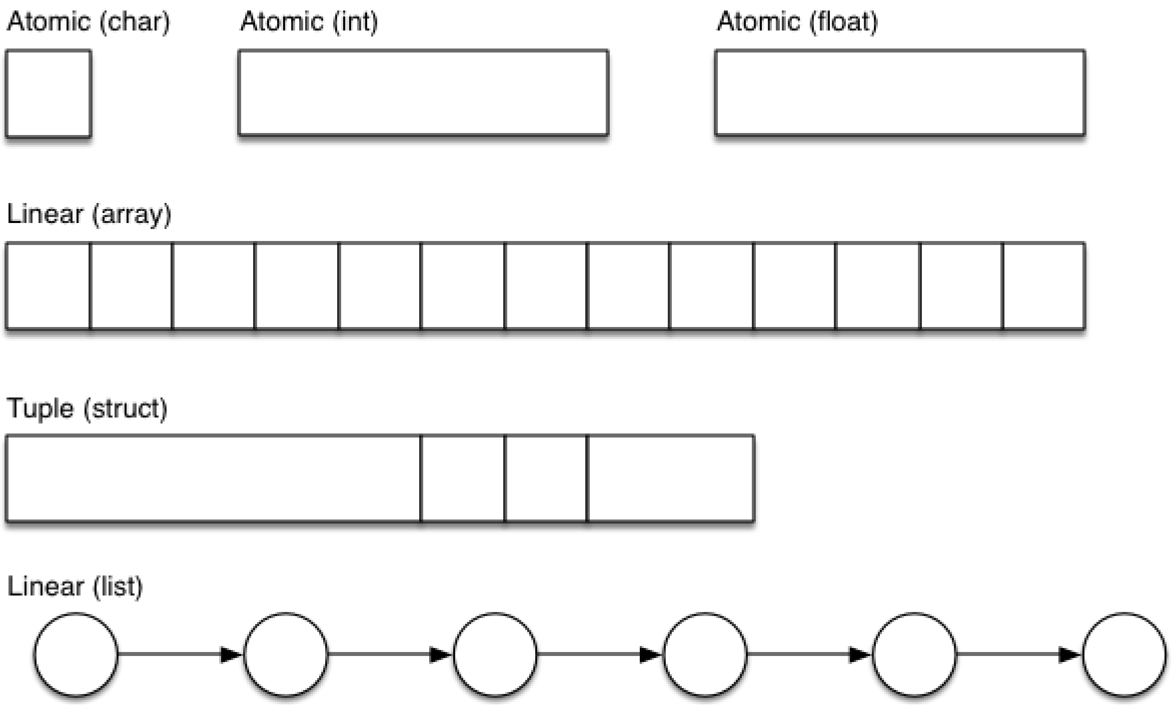
❖ Post-conditions

At the *end* of this course you should be able to:

- analyse performance characteristics of algorithms
- measure performance behaviour of programs
- choose/develop effective data structures (DS)
- choose/develop algorithms (A) on these DS
- package a set of DS+A as an abstract data type
- develop and maintain 9999-line C programs

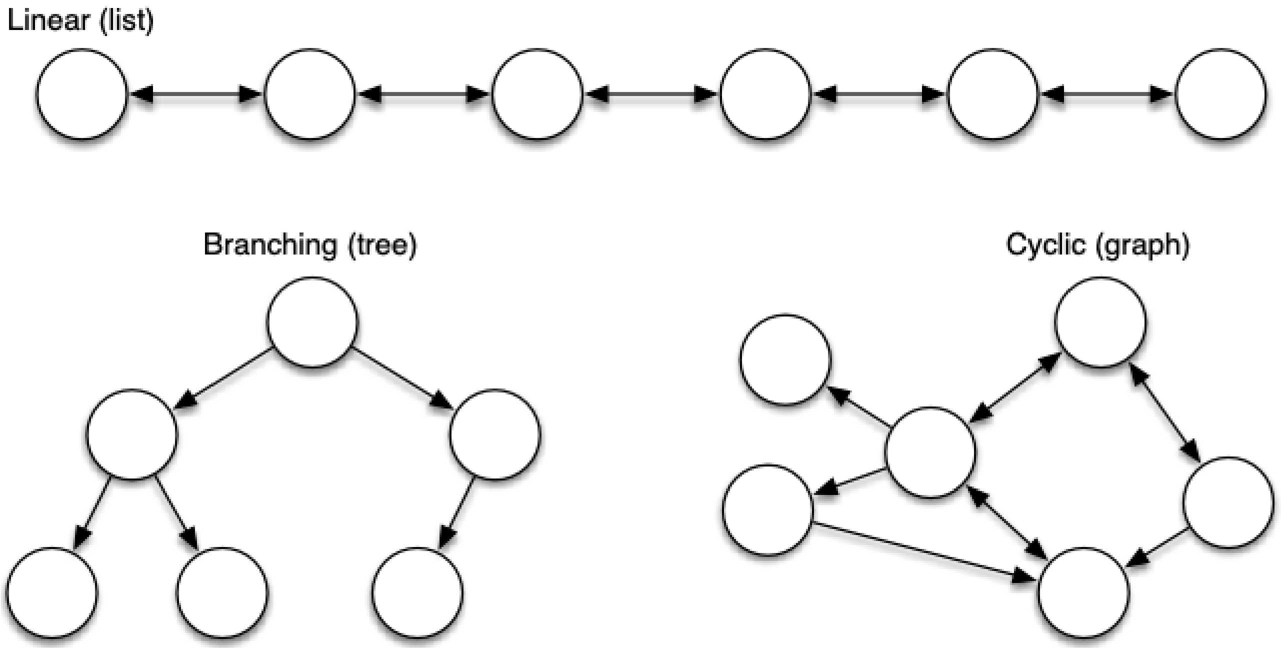
❖ Data Structure Viewpoint

COMP1511 looked at ...



❖ ... Data Structure Viewpoint

COMP2521 also looks at ...



❖ COMP2521 Themes

Major themes ...

1. Analysis: correctness, performance, *style*
2. ADTs: sets, lists, trees, graphs, dictionaries
3. Operations: building, sorting, searching, traversing

For data types: alternative implementation of operations

For algorithms: complexity analysis, performance analysis

❖ Credits for Material

Always give credit if you use someone else's work.

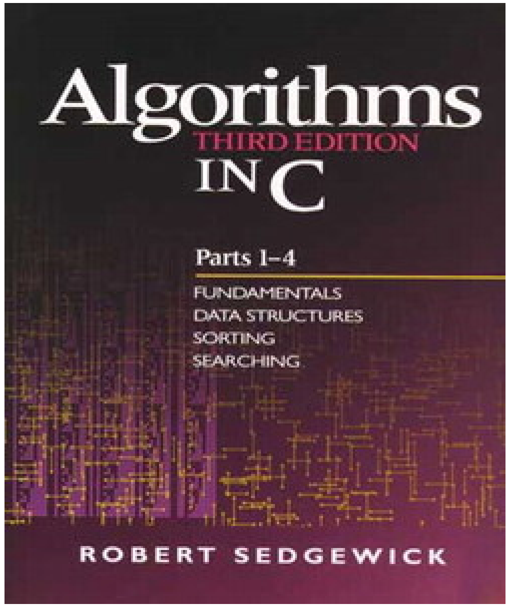
Most material was prepared by me, using ideas drawn from

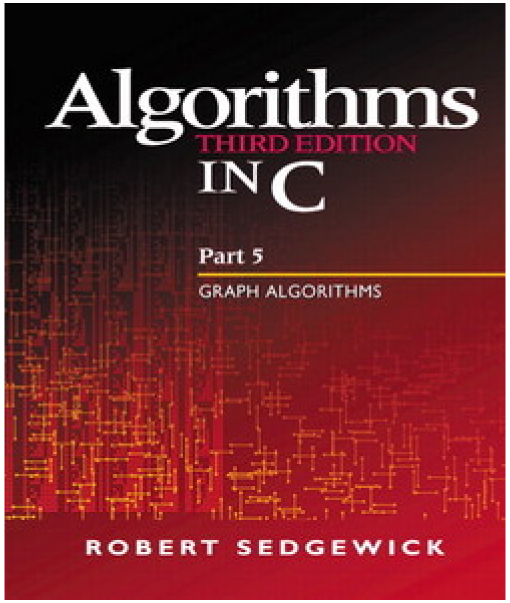
- notes by Aleks Ignjatovic (COMP2011 2005)
- slides by Manuel Chakravarty (COMP1927 08s1)
- lectures by Richard Buckland (COMP1927 09s2)
- slides by Gabrielle Keller (COMP1927 12s2)
- slides by Michael Thielscher (COMP9024 17s2)
- slides by Ashesh Mahidadia (COMP2521 2018-2020)
- slides and books by Robert Sedgewick

❖ Textbook

Textbook is a "double-header"

- Algorithms in C, Parts 1-4, Robert Sedgewick
- Algorithms in C, Part 5, Robert Sedgewick





Good books, useful beyond COMP2521, but code style .....

❖ How does the course run?

We provide the following

- content ... via slides and videos
- review ... via tutes and quizzes
- practice ... via labs and assignments
- summation ... final exam

And, this term, all delivered online ... thank you, COVID-19

❖ "Lectures"

No lectures in 20T2.

Instead ...

- topic-based videos 20-50 mins in length.
- available the week before covered in tutes/labs

augmented by

- on-line problem-solving sessions in lecture time-slots

Details available at the start of each week.

❖ Tutes and Labs

Tutorials ...

- as in COMP1511
- practise analysis/design; clarify lecture material

Labs ...

- small(ish) implementation tasks, done in pairs
- give skills practice (leading on to assignments/exam)

Tutes/labs will run from Weeks 1 to 10 (but not 6)

Exercises for Week X available at end of Week X-1

❖ ... Tutes and Labs

There are 9 lab exericses (weeks 1-5,7-10).

Lab exercises contribute 18% to overall mark.

The lab exercises for Week X must be

- submitted before Sunday at end of week X
- demonstrated to tutor *during* Week X lab
  OR, demonstrated *at the start of* Week X+1 lab

We take marks for best 7, BUT you should do them all.

Total mark for labs is greater than 18 (but they are scaled to 18).

❖ Quizzes

There are 8 online quizzes (weeks 2-5,7-10)

Quizzes contribute 12% to overall mark.

Using Webcms3 quiz module (m/c, numeric, fill-in-the-blank)

Done in your own time; resubmission is allowed.

Quiz timeline ...

- released on Sunday at start of Week X
- due before midnight Friday of Week X

We take marks for best 6, BUT you should do them all.

Total mark for quizzes is greater than 12 (but is scaled to 12).

❖ Assignments

Two assignments ...

- Ass1: 15% towards final mark, on trees, individual
  (available in Week 02, due in Week 05)
- Ass2: 15% towards final mark, on graphs, group-based
  (available in Week 06, due in Week 10)

Assignments contribute 30% towards final mark.

Total mark for each assignment is greater than 15 (scaled to 15).

Late penalties apply if you miss assignment deadlines.

Good time management avoids late penalties!

❖ ... Assignments

Assignment 1 ...

- a C programming exercise, with ADTs
- still thinking about it ... but will involve trees
- done individually, with auto-marking

❖ ... Assignments

Assignment 2 ...

- implement parts of the game "Fury of Dracula"
- run via nightly tournaments
- carried out in groups of 4 or 5
- peer assessment of contribution (no passengers)
- copying old solutions won't work ... we're changing the rules

❖ Plagiarism

Just Don't Do it

❖ Final Exam

24-hour on-line exam during the exam period.

Exam should take ~3 hours (anytime during the 24-hour period)

On-line questions via email to class account.

On-line documentation available in exam:

- C quick reference; Unix programmers Manual

Format:

- some programming exercises (Prac)
- some descriptive/analytical questions (Theory)

❖ ... Final Exam

Final exam contributes 40% towards final mark.

Hurdle on final exam: must score at least 17/40

- failure to meet hurdle results in UF grade

Plagiarism checking on programming qeustions

How to pass? ...

- do the labs and assignments yourself
- practise, practise, practise, practise, practise, ...

❖ Special Consideration

UNSW has centralised special consideration processing

- all requests for extensions, supps, etc. must be documented
- apply via student.unsw.edu.au/special-consideration
- also send email to the class account cs2521@cse.unsw.edu.au

For more info, see Essential Advice for CSE Students

❖ Supplementary Exams

If you are unable to sit the Final Exam on the scheduled day ...

- apply for special consideration, with documentation
- must show how you were prevented from sitting the exam

The "fit-to-sit" rule applies ... if you take the exam, no Supp

Supp Exams are centrally timetabled, during O-week Term 3

It is your responsibility to check for details of Supp Exam.

❖ Course Assessment

```
quizzes     = mark for quizzes         (out of 12)
labs        = mark for lab exercises   (out of 18)
ass1        = mark for assignment 1    (out of 15)
ass2        = mark for assignment 2    (out of 15)

finalExam   = finalExam                (out of 40)
okExam      = finalExam >= 17/40

mark        = quizzes + labs + ass1 + ass2 + exam
grade       = HD|DN|CR|PS  if mark >= 50 && okExam
            = FL           if mark <  50
            = UF           if mark >= 50 && !okExam
```

❖ Summary

The goal is for you to become a better programmer

- more confident in your own ability
- with an expanded set of tools to draw on
- able to analyse/justify your choices
- producing a better end-product
- ultimately, enjoying the programming process

Produced: 30 May 2020