COMP1521 Staff

Convenor,
Lecturer Sri Parameswaran
Admins Jashank Jeremy, Zac Kologlu

COMP1521 Student Background

Students in this course have (mostly) completed:

- COMPI511 or COMPI911

Everyone has learned fundamental C programming.

COMPI511 also studied linked lists, ADTs ...

since not everyone has seen these, we won’t use them

For this week’s tuts and labs:

- review/strengthen assumed C knowledge
Course Goals

COMP1511/1911 ...
- gets you thinking like a *programmer*
- solving problems by developing programs
- expressing your solution in the C language

COMP1521 ...
- gets you thinking like a *systems programmer*
- with a deep understanding of run-time behaviour
- and better able to reason about your C programs

These are *not* the same goals as COMP2121.

COMP1511/1911 vs COMP1521

COMP1511/1911 ...

COMP1521 ...

or maybe ...

Course Context

Themes

Major themes ...

- software components of modern computer systems
- how C programs execute (at the machine level)
- how to write (MIPS) assembly language
- Unix/Linux system-level programming
- how operating systems are structured
- introduction to concurrency, concurrent programming

Goal: you are able to understand execution of software in detail.
There is no prescribed textbook for COMP1521. Recommended reference ...

*Computer Systems: A Programmer’s Perspective*, Bryant and O’Hallaron
- covers most topics, and quite well
- but uses a different machine code

Available in UNSW Bookshop

---

**Textbook**

There is no prescribed textbook for COMP1521.

Material has been drawn from

- *Introduction to Computing Systems: from bits and gates to C and beyond*, Patt and Patel
- *The Elements of Computer Systems: Building a modern computer system from first principles*, Nisan and Schocken
- COMP2121 Course Web Site, Parameswaran and Guo

Always give credit to your sources!

---

**Systems and Tools**

Prac work based on Linux tools
- all tools available on the CSE lab machines
- can use VLAB to connect to CSE from home

Compilers: dcc on CSE machines (clang or gcc elsewhere)

Assembly language: MIPS on QtSpim (also Xspim on CSE)

Use your own favourite text editor

Other tools: make, gdb, man, bc -l

Learn to love the shell and command-line ... very useful!
Lectures

- Wednesday, 14:00—16:00; Thursday, 12:00—14:00;
delivered via YouTube (and via Microsoft Teams Live Events)

  - you will have email about how to access the event
  - feel free to ask questions via chat
  - lectures recorded and linked from course home page.

- present a brief overview of theory
- focus on practical demonstrations of coding
- demonstrate problem-solving (testing, debugging)
- lecture slides available on the web before lecture.

Tut-labs

- COMP1521 has 3 hour tut-labs starting week 1
- all timeslots have online classes
- delivered via Blackboard Collaborate
- many timeslots have face-to-face classes
- please follow UNSW policy: https://www.covid-19.unsw.edu.au/
- ensure you swipe your student card when directed

Tutorials

To get the best out of tutorials ...

- attempt the problems yourself beforehand
- ask if you don’t understand a question or how to solve it
- Do not keep quiet in tutorials ... talk, discuss, ...
- Your tutor may ask for your attempt to start a discussion.
Lab Classes

Each tutorial is followed by a two-hour lab class.

- Several exercises, mostly small coding tasks
- Build skills needed for assignments, exam
- Done individually
- Submitted via give, before Monday 21:00
- Automarked (with partial marks) — 15% of final mark
- Labs may include challenge exercises ... 
  - may be silly, confusing, or impossibly difficult
  - almost full marks (95+%) possible without completing any challenge exercises

Weekly Tests

From week 3, weekly tests:

- immediate reality-check on your progress.
- done in your own time under self-enforced exam conditions.
- time limit of 1 hour
  - can keep working after hour for 50% of mark
- automarked (with partial marks)
- best 6 of 8 tests contribute 10% of final mark
- any violation of test conditions ⇒ zero for whole component

Assignments

- Ass1: Assembly Language, weeks 4–7, 15%
- Ass2: C Programming, weeks 7–10, 15%
- Assignments give you experience with larger programming problems than lab exercises
- Assignments will be carried out individually.
- They always take longer than you expect.
- Don’t leave them to the last minute.
- There are late penalties applied to maximum marks, typically 2%/hour
CSE offers an inclusive learning environment for all students.

In anything connected to UNSW, including social media, these things are student misconduct and will not be tolerated:

- racist/sexist/offensive language or images
- sexually inappropriate behaviour
- bullying, harassing or aggressive behaviour
- invasion of privacy

Show respect to your fellow students and the course staff

Cheating of any kind constitutes academic misconduct and carries a range of penalties. Please read course intro for details.

Examples of inappropriate conduct:

- groupwork on individual assignments (discussion OK)
- allowing another student to copy your work
- getting your hacker cousin to code for you
- purchasing a solution to the assignment

- Labs, tests, assignments must be entirely your own work.
- You can not work on assignments as a pair or group.
- Plagiarism will be checked for and penalized.
- Plagiarism may result in suspension from UNSW.
- Scholarship students may lose scholarship.
- International students may lose visa.
- Supplying your work to any another person may result in loss of all your marks for the lab/assignment.
Final Exam

- online practical exam (you complete from home)
- limited on-line language documentation available
- some multiple-choice/short-answer questions, similar to tut questions.
- some questions will ask you to read C or assembler
- most marks for questions which ask you to write C or assembler
- also may ask you to answer written questions
- you must score 18+/45 on the final exam to pass course

Assessment

- 15% Labs
- 10% Weekly Programming Tests
- 15% Assignment 1 — due week 7
- 15% Assignment 2 — due week 10
- 45% Final Exam

Above marks may be scaled to ensure an appropriate distribution

To pass, you must:

- score 50/100 overall
- score 18/45 on final exam

For example:

55/100 overall, 17/45 on final exam ⇒ 55 UF not 55 PS

How to Pass this Course

- coding is a skill that improves with practice
- the more you practise, the easier you will find assignments/exams
- do the lab exercises
- do the assignments yourself
- practise programming outside classes
- treat extra tutorial questions like a mini prac exam
Assumed knowledge —

- design an algorithmic solution
- describe your solution in C code, using ...
  - variables, assignment, tests (==, !, <=, &&, etc)
  - if, while, for, break, scanf(), printf()
  - functions, return, prototypes, *.h, *.c
  - arrays, structs, pointers, malloc(), free()

**Not** assumed knowledge —

- linked structures, file operations, ADTs, sorting,
- recursion, bit operations