COMP(2041|9044) 26T2 — Course Introduction

https://www.cse.unsw.edu.au/~cs2041/26T2/

COMP(2041|9044) Staff

Convenor Andrew Taylor

Lecturers Andrew Taylor

Admin Anna Brew, Dylan Brotherston, Rosemary Ai

Tutors Alex Dugina, Alex Lee, Alexander Blackmore, Anna Brew, Anthony Anastopoulos, Cherise Chan, Chris Liang, Christopher Casolin, Daniel He, David Shun, David Choulex, Duong Nguyen, Dylan Brotherston, Dylan Zhang, Haibo Zhang, Keshiga Gopalarajah, Khye Jac Low, Kyu-Sang Kim, Lachlan Huynh, Lachlan Crawford, Leon Qian, Mitchell Wang, Nicholas Sebastian, Raine Park, Rosemary Ai, Sandy Tran, Terence Lim, Tiger Liu, Tim Arney, Tomi Chong, Vedang Purohit, Wayne Sun, Winnie Chan, Wisesa Resosudarmo

Course Goals

- First programming courses deal with ...
  - one language (C or Python at CSE)
  - one program
  - small(ish) tightly-specified examples
  - narrow aspects of programming (e.g. basics, correctness)
- COMP(2041|9044) deals with ...
  - other languages (Shell & Python)
  - combining multiple programs to solve problems
  - larger (less-small) less-specified examples
  - tools for working with software (e.g. git, docker)
  - configuring systems (e.g package managers, mounting)
- get you to the point where:
  - you could build a package
  - put it on github
  - and have people download & use it

Course Goals - TLDR

COMP2041/COMP9044 will expand your coding skills

Assumed Knowledge

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

- write, debug, test programs in C or Python
  - OK for COMP2041/COMP9044 if you don't know C
  - basic Python knowledge will be assumed
  - COMP9021, COMP1531 (pre 2022), learning Python elsewhere, sufficient
- appreciate the use of abstraction in computing

Changes from recent years

- no web frontend/backend programming
  - moved to COMP6080
  - script to scrape/download web data covered
- no Perl
  - Python will be used to teach same material
  - Perl much less important that when COMP2041 started
  - basic Python assumed, more covered
- COMP1531 used to teach Python
  - Various levels of Python knowledge coming in to COMP2041/COMP9044
  - COMP2041/COMP9044 will assume basic Python knowledge

Lectures

- Tuesday, 14:00—16:00; Wednesday 14:00—16:00;
  - Some lectures delivered in person
  - Some lectures delivered live-streamed via YouTube
  - 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.

Tutorials

- Tutorials **start in week 1**.
  - online classes are via Blackboard Collaborate
- tutorials clarify lecture material
- work through problems related to lecture topics
- give practice with design *(think before coding)*
- answers available on the class webpage Friday afternoon
- tutorials run every week except flex-week (week 6)
  - including week 10

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 implementation/analysis tasks
- Aim to build skills needed for assignments, exam
- Aim to give experience applying tools/techniques
- Done **individually**
- Submitted via `give`, before Monday 12:00 (midday) the following week
- Automarked (with partial marks) — 15% of final mark
- Labs may include challenge exercises:
  - may be silly, confusing, or impossibly difficult
  - 95% possible for labs without completing any challenge exercises

Weekly Tests

From week 3, weekly tests:

- programming tests
- immediate reality-check on your progress.
- done in your own time under self-enforced **exam conditions**.
- Time limit of 1 hour
  - Half marks awarded for submissions after 1 hour
- Automarked (with partial marks) — 10% of final mark
- best 6 of 8 tests used to calculate the 10%
- any violation of test conditions $\Rightarrow$ zero for whole component

Assignments

- Assignments give you experience applying tools/techniques to 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.
- Help sessions will be available to assist with assignments.
  - will be **very** busy in the last days before an assignment is due.

Late Penalties

- Labs, Tests, and Assignments all have the same late penalty
- UNSW standard late penalty
- `0.2`% taken from your raw mark for each hour late
  - starts small, but adds up quickly
- after 5 days (125 hours because of maths) late, 100% penalty is applied, e.g.:
  - If your raw mark is 80/100 but you submit 1 minute late (rounded up to 1 hour), your mark will be 79.8/100
  - If your raw mark is 80/100 but you submit 1 hour and 1 minute late (rounded up to 2 hours), your mark will be 79.6/100
  - If your raw mark is 78/100 but you submit 3 days, 8 hours and 42 minutes late (rounded up to 81 hours), your mark will be 61.8/100
  - If your raw mark is 100/100 but you submit 5 days, 1 minute late (rounded up to 125 hours or more), your mark will be 0/100

Sample Solutions and Marking

Because of the late penalty allowing late submissions up to 5 days after the deadline
along with extensions for special consideration that may be granted

- Sample solutions for labs, and tests will be released two weeks after the due date.
- Marks for labs, and tests will be released between 12 days and 2 weeks after the due date.
- This means that solutions and marks for the last lab, and test will not be released until after the final exam.
- Sample solutions for assignments are not released.
- Marks for assignments are released in two parts.
  - Automarking will be released 2 weeks after the due date.
  - Hand marking (style, automarking adjustments, etc.) takes longer
    and will be released another 2 weeks or so after the automarking.

Code of Conduct

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

Plagiarism

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

Examples of inappropriate conduct:

- group work on individual assignments (discussion OK)
- reading someone else's solution before stating an assignment
- allowing another student to copy your work
- getting your hacker cousin to code for you
- purchasing a solution to the assignment

Plagiarism

- Labs, tests, assignments must be entirely your own work.
- You can not work on assignment 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.

Use of Generative AI Tools

- Generative AI tools, e.g. github copilot, chatGPT have great potential to assist coders
- Code they generate often has subtle errors & security vulnerabilities
  also often generate poor code
- expert coders (hopefully) can spot these problems
- need a deep understanding of language/system to make good use of these tools
- Use of tools like copilot, chatGPT may slow you getting this understanding
- Use of generative AI tools including github copilot, chatGPT **not permitted** in COMP2041/9044
- except assignments may permit use small amount of generated code with **attribution**
  - read the spec carefully
- other courses may allow use of these tools

Final Exam

- **in-person** practical exam, during exam period; you complete **in CSE labs**
- closed-book — limited language documentation available
- some multiple-choice/short-answer questions, similar to tut questions.
- some questions will ask you to read shell, Python, regex, ...
- 8-12 implementation questions, similar to lab exercises
- most marks for questions which ask you to write shell or Python
- also may ask you to answer written questions
- you *must* score 18+/45 (40%) on the final exam to pass course

Assessment

- 15% Labs
- 10% Weekly Programming Tests
- 15% Assignment 1 — due Monday week 7
- 15% Assignment 2 — due Monday week 11
- 45% Final Exam

Above marks may be scaled to ensure an appropriate distribution

To pass you must:

- score 50/100 overall
- score 18/45 (40%) on final exam

For example:

55/100 overall and 17/45 on final exam $\Rightarrow$ **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
- take weekly tests seriously
- start the assignments early
- practise programming outside classes
- treat extra tutorial questions like exam practice