Australian
National
University

COMP1730/COMP6730
Programming for Scientists

Introduction and administrative matters
Acknowledgment of the country

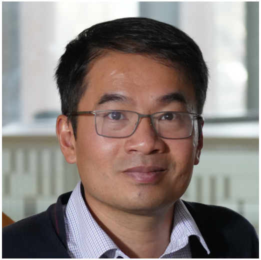
We acknowledge and celebrate the First Australians on whose traditional lands we meet, and pay our respect to the elders past and present.

Conveners

Dr. Alberto F. Martin

Physical Science,
High Performance Computing

A/Prof. Minh Bui



Bioinformatics,
Computational Biology

https://comp.anu.edu.au/courses/comp1730/people/

Senior tutors

Hancheng Shao

Robert McArthur

https://comp.anu.edu.au/courses/comp1730/people/

Other Tutors

- Dr. Alexei Khorev
- Dr. Aline Maalouf
- Bodan Liu
- Chathura Nagoda Gamage
- Dilmi Jayasena
- Han Zhang
- Muhammad Salman
- Nutthadech Banditakkarakul
- Rohan Khatwani
- Sandy Zhao
- Shashank Gummuluru
- Zora Zhuang

Announcements

- Complete “week 1 checklist” on course website
- Read News & Announcements on Ed Discussion, **importantly the weekly information**
- Lectures: Mondays 1-2pm and Tuesday 4-5pm.
  - Ignore message in timetabling system that lecture is full. Just come to the lecture hall!
- Labs: from week 2 (no lab in week 1)
  - Lab in week 2 is to practice concepts from lectures in week 1, and so on.
- Install Fest in week 1:
  - Friday, 26/07, 6-9pm, Hanna Neumann (HN 1.23 and 1.24)
  - Robert McArthur will be there to help you out
  - Food provided!

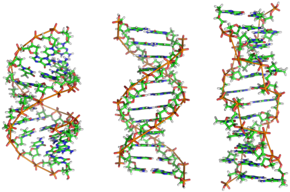
Lecture outline

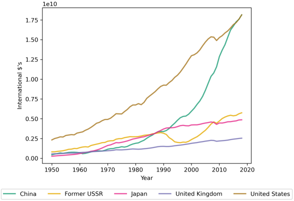
- Why programming for scientists?
- Course overview.
- Info, contacts & schedule.
- Assessment scheme.
- Academic integrity.
- Student representatives.

Why programming for scientists?

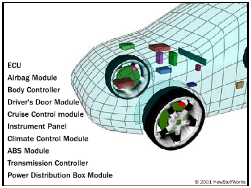
"Science rests on data, processing data needs software."

- **Biology:** use DNA data to understand evolution of Life on Earth and track COVID-19 virus variants. http://www.iqtree.org
- **Economics:** Modelling GDP growth over time and across countries. https://quantecon.org (Prof. John Stachurski, ANU)

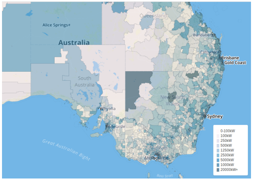




- Technical systems increasingly run on software.
  - **Engineering:** Software on a modern car has > 100M lines of code.



- Simulation and optimisation are needed to solve large-scale design challenges.
  - Intermittent renewables produced ∼35.9% of Australia's electricity in 2022. How do we design the grid to work with 100%?



- *As scientist or engineer, you need to understand how software works, and perhaps extend it*:
  - understand algorithms and implementation to interpret and explain their results;
  - debug programs (find and correct errors);
  - modify existing programs to solve your (unique) problem.
- By the end of the course, we hope you’ll tackle a novel problem by saying, “Hey, I can just write a program to solve that...”

Programming example

- you want to calculate the monthly repayment of a $500,000 home loan...
  - use one of the on-line mortgage calculators?
- ...for all loan terms in 20-30 years, and an interest rate of 2% to 6%.
- The formula is

  $$A = P\frac{r(1+r)^n}{(1+r)^n - 1}$$

  where P is principal (loan amount), r is interest rate, n is loan duration. (derive it, or look it up on wikipedia).
- Let's write a program!

```
# first way to calculate monthly repayment
P = 500000
r = 0.06
n = 30
year_repayment = P * r * (1+r)**n / ((1+r)**n - 1)
print(year_repayment/12)

# second way to calculate monthly repayment
P = 500000
r = 0.06/12
n = 30*12
month_repayment = P * r * (1+r)**n / ((1+r)**n - 1)
print(month_repayment)
```

Why python?

- This is *not* a course on programming in python; it's a course on programming, that uses python.
- Python is nowadays the *most popular* programming language,
- particularly for science and engineering uses.
- Open source, available on most platforms.
- Many modules:
  - over 200 in the python standard library;
  - over 100,000 on `pypi.org`.
- We will use **python 3**.

Course description & aims

- Introduction to programming (using python).
  - No prior programming or computer science knowledge is required.
  - This does not mean it is easy!
- Two aims:
  - Programming as a practical skill.
  - Understand some basic CS concepts; build foundation for later courses.

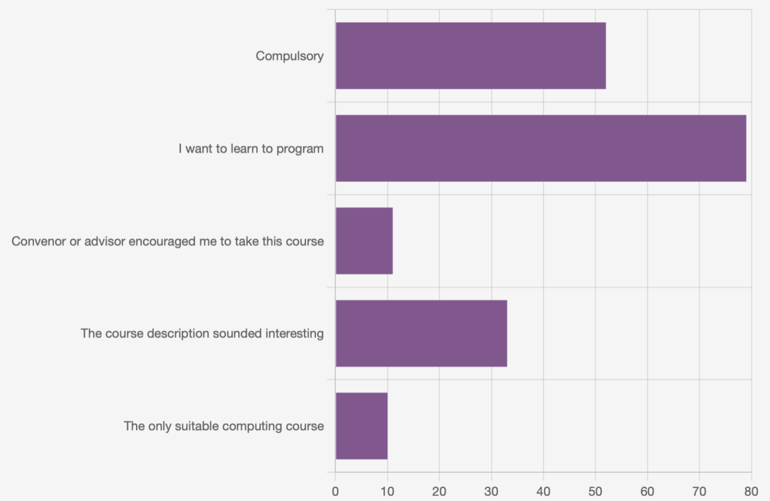
Learning outcomes

(adapted from ANU Programs & Courses)

Students who succeed in all aspects of this course will be able to:

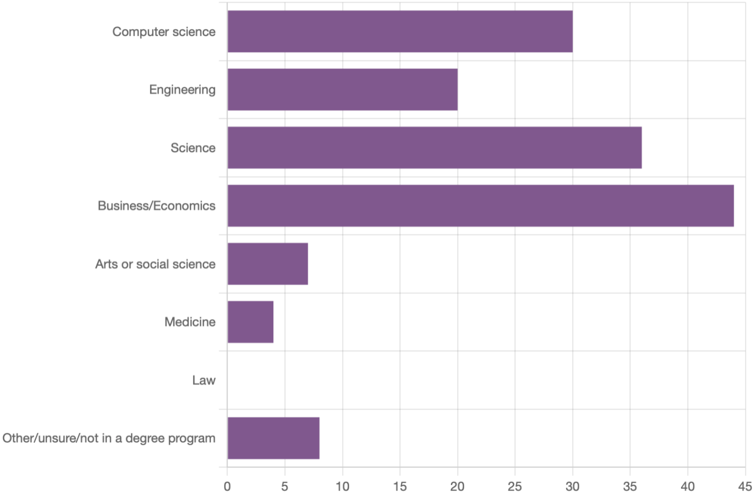
- Design and write programming code to solve practical problems of a scientific or engineering nature.
- Read, test and debug small computer programs.
- Use key python libraries for data processing and visualisation.
- Understand widely-used algorithms and data structures, and their computational complexity.
- Apply design approaches used in scientific pipelines, including data abstraction and array-based and object-oriented programming.
- Understand and apply principles of high code quality.
- Communicate effectively to both specialist and non-specialist audiences about data processing problems in writing and verbally.

About you: why do you take this course?

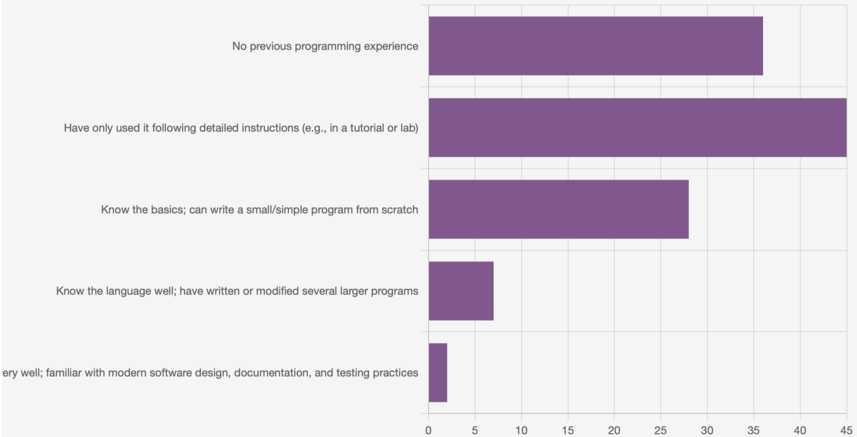


(*preliminary data – only 129 responses until 21/07/2024*)

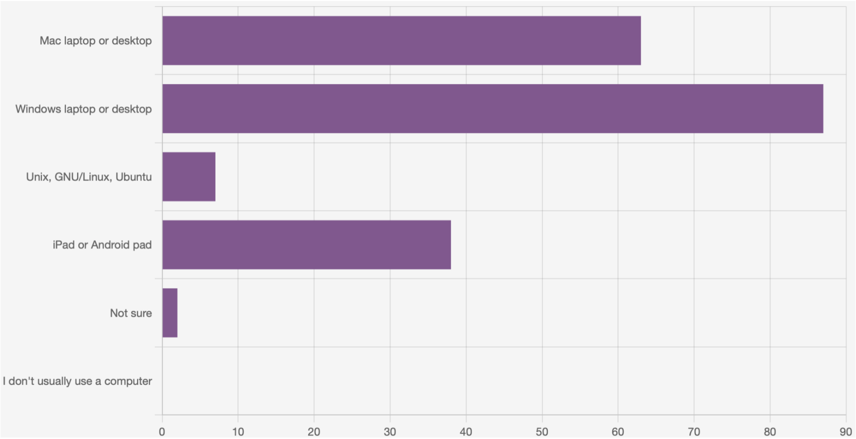
About you: students in the course



Prior programming knowledge



Platform



You need to do final exam on lab Linux machines!

Course info & contacts

- https://comp.anu.edu.au/courses/comp1730/
- Ed Discussion forums, quizzes, surveys, assignment submission. *Read news & announcements*!
- To ask a question:
  - Use the discussion forum on Ed Discussion.
  - For *personal* questions, email `comp1730@anu.edu.au` (Always *use your ANU email*), or
  - Minh Bui's 2- hours: Mon 2-3pm
  - Alberto Martin's office hours: Tue 5-6pm.

Discussion forum – 3 simple rules

1. **Read before you post.**
   Before posting a question, check if your question has already been answered.

2. Give your post a **good, descriptive topic**.
   Don’t write “A question”. Write something like “Variable assignment: why does the value not change?”.

3. You **may not post** solutions to assignment problems.

- This applies to <u>any</u> on-line forum.

Schedule overview

- https://timetabling.anu.edu.au/sws2024/
- Two lectures / week.
  - Recording on echo360, if unable to attend or for later use.
  - Follow content & schedule on the course web site, and read the news & announcements.
- One 2-hour lab / week (starting from week 2).
  - Select your lab on MyTimeTable until end of week 1!
- You are expected to spend another 6 hours to study the course (e.g., solving all lab exercises - lab is not meant to solve within 2 hours).

Assessment scheme

https://comp.anu.edu.au/courses/comp1730/assessments/

- 5 small homework assignments (15%)
- 1 larger project assignment (35%)
  - Mandatory in-lab assessment!
- Final exam (50%), it's a hurdle - at least 15/50 to pass

S. Week	
2	Homework 1 due (Sunday)
3	Homework 2 due (Sunday)
5	Homework 3 due (Sunday)
	Break
Break	Homework 4 due (Sunday)
8	Homework 5 due (Sunday)
10	Project due (Sunday)
11	**In-lab project assessment!**
Exam period	Final exam(s)

- Final mark is the sum of assessment marks, with 50% required to pass (mind the final exam hurdle!)
- If you are close to the boundary of the higher grade, e.g., 48-49% or 78-79%, we will consider pushing it up based on your **lab participation**, such as interaction with tutor, submitting code to Ed Lessons, answering quizzes.
- Note: "*any submitted work may be subject to an additional oral examination*", which can change the assessment mark in any way.
- All assignment deadlines are hard – late submissions without an approved extension will not be accepted.
  - No partial marks for late submissions!

Academic integrity

- Academic integrity is taken seriously at ANU! – Academic Integrity Rule 2021 is a legal document at the University.
  - Being uninformed of or misunderstanding the Rule is never an excuse for a breach of academic integrity.
  - Any student is expected to undertake the online **Academic Integrity modules on Wattle**.
- Discussing programming problems (e.g. from labs) and ways to solve them with other students is a great way to learn – just don't discuss assessment problems.
- All assignments are *individual*. You must write your own code, and be able to show that you understand every aspect of what you have written.

- The final exam will be in-person and *individual*. You may not discuss the exam questions or your answers with anyone.
- Any academic misconduct will leave a record on internal student file or even appear on your transcript in severe cases.
- If you are unsure, please ask your tutor or conveners.
- Examples of NOT OK:
  - *"The code I used in my assignment was pulled from StackOverflow but I didn't realise I had to reference an online post."* **(Plagiarism)**
  - *"I used several sources to solve this assignment. There's a mix of my ideas and parts of others. I thought it was considered mine."* **(Plagiarism)**
  - *"I discussed the* ***individual assignment*** *with a friend and acknowledged their contribution to my assignment."* **(Collusion)**

Student representatives

- Class Student Representation is an important component of the teaching and learning, quality assurance and quality improvement processes.
- Students can nominate themselves for one or more of the courses they are enrolled in.
- The role is to provide ongoing constructive feedback on behalf of students to course conveners and to Associate Director (Education) for improvements to the course.

Responsibilities of Class Reps

- Act as the official liaison between your peers and convener
- Be available and proactive in gathering feedback from your classmates
- Attend regular meetings, and provide reports on course feedback to your course convener
- Close the feedback loop by reporting back to the class the outcomes of your meetings
- Interested? contact us `comp1730@anu.edu.au` by end of week 1.