COMP1730/COMP6730
Programming for Scientists

Introduction to the course
and administrative matters

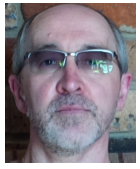
Convenor

Brian Parker

Bioinformatics &
Computational Biology

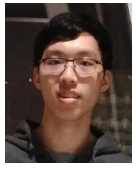
Tutors

Alexei Khorev



Malcolm Macdonald

Hancheng Shao



Jonathan Connor

Jonathan Ting

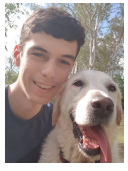
Kanav Thareja

Madhawa Perera

Mindika Premachandra

Richa Awasthy

Robert McArthur



Samuel O'Brein

Xiaodi Zhang

Chathura Gamage

Dilmi Jayasena

Jack Kennedy

Jamie Whittington

Minwei (Sandy) Zhao

Pok Wing (Cathy) Cheung

Vimukthini (Vimu) Pinto

Vinayak (Shashank) Gummuluru

Announcements

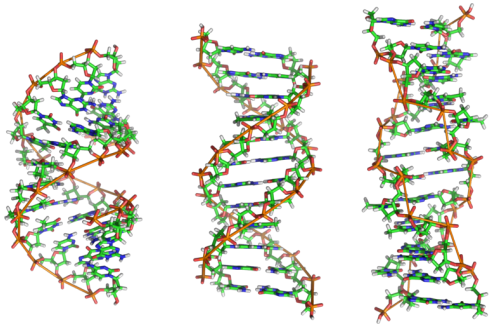
* Read announcements made in the news forum on wattle
* Particularly the weekly notice
* Wattle discussion forum for questions about the course and lab content
* `comp1730@anu.edu.au` for personal matters

Lecture outline

* Why learn programming?
* Course overview.
* Info, contacts & schedule.
* Assessment scheme.
* Important TODOs.

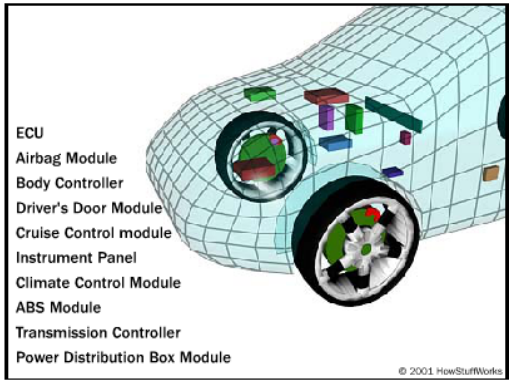
Why learn programming?

* Science rests on data... *more and more data*.
  - >8.4M novel coronavirus (COVID-19) genomes available; >10,000 new genomes sequenced per day (`www.gisaid.org`)
  - The Australian SKA Pathfinder radio telescope outputs 2.5GB/s (the SKA is expected to be around 100 times more).

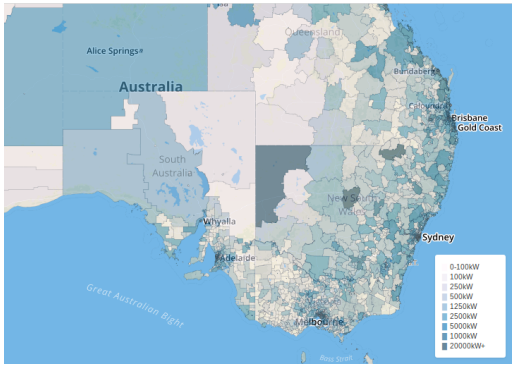


* Processing this data needs software.

* Technical systems increasingly run on software.
  - A modern car has over 30 computers, running $>$1,000,000 lines of code.



* Simulation and optimisation are needed to solve large-scale design challenges.
  - Intermittent renewables produced $\sim$8.25% of Australia's electricity in 2017. How do we design the grid to work with 100%?



* As scientist or engineer, you will need to understand how software works, and how to modify or 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 thinking, "Hey, I can just write a program to solve that..."

Programming example

* you want to calculate the monthly cost of a $600,000 home loan...
  - use one of the on-line calculators?
* ...for all loan terms in 10-25 years, and an interest rate of 2.5%, 3.5% or 4.5%.
* The formula is

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

(derive it, or look it up on wikipedia).
Let's write a program!

```
import math
import matplotlib.pyplot as mpl

def monthly_cost(principal, interest_rate, years):
    monthly_rate = interest_rate/12
    # interest rate is given in % so need to divide by 100
    r = monthly_rate/100
    n_payments = years * 12
    return principal * ((r * math.pow(1 + r, n_payments)) /
                        (math.pow(1 + r, n_payments) - 1))

years = range(10,26)
mc = [monthly_cost(600000, 2.5, y) for y in years]
mpl.plot(years, mc, 'g-')
mc = [monthly_cost(600000, 3.5, y) for y in years]
mpl.plot(years, mc, 'b-')
mc = [monthly_cost(600000, 4.5, y) for y in years]
mpl.plot(years, mc, 'r-')
mpl.show()
```

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.

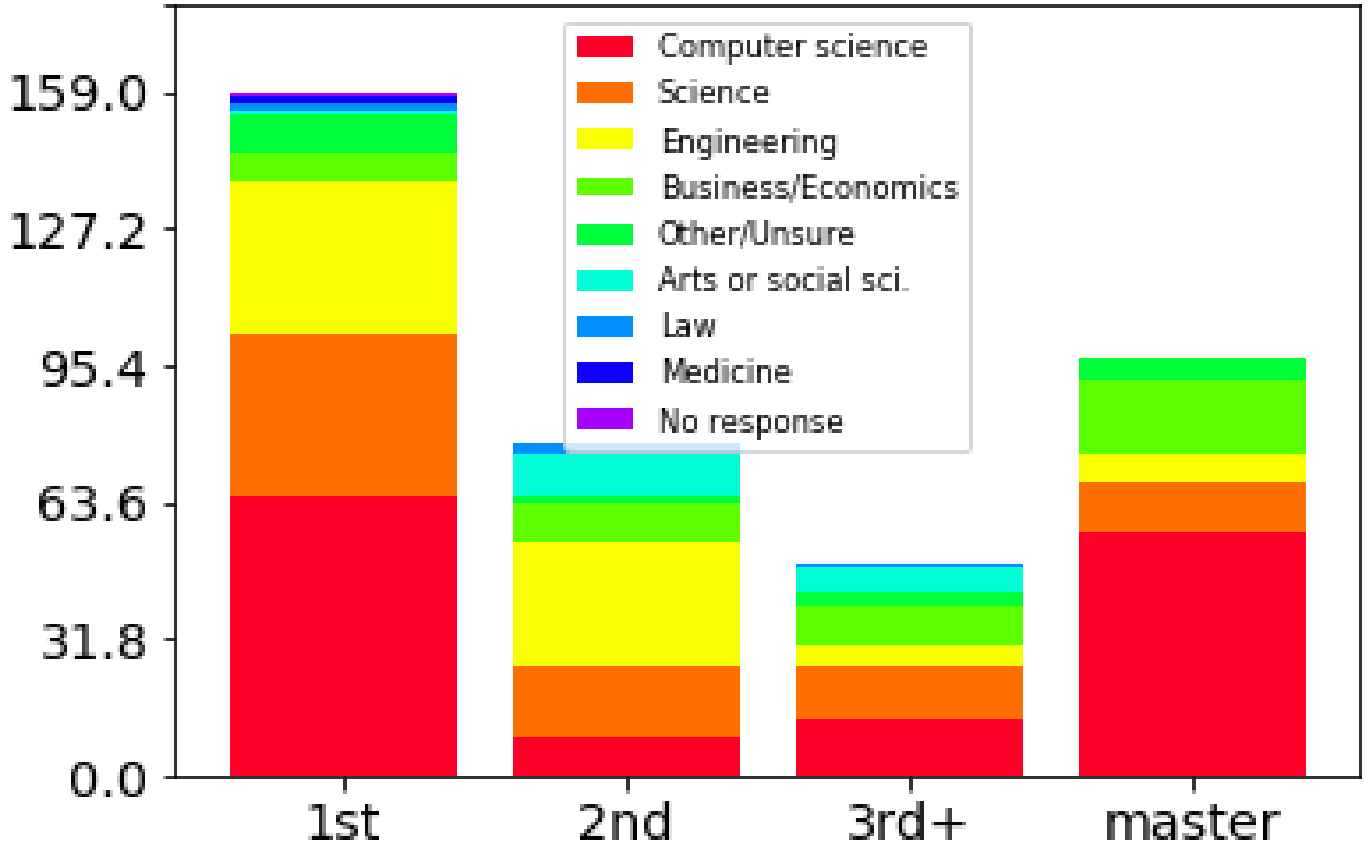
Learning outcomes

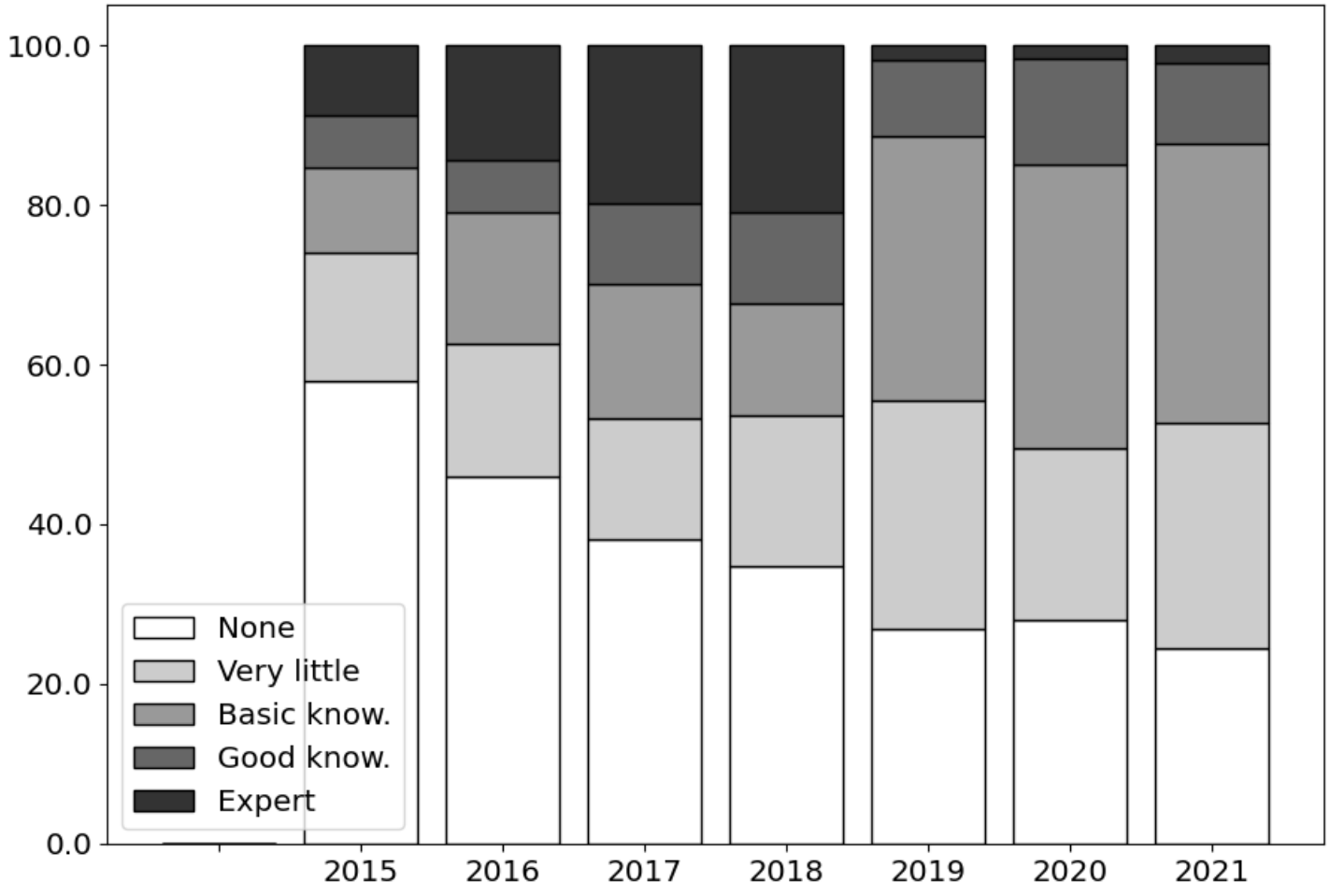
(revised from ANU Programs & Courses)

Students who succeed in all aspects of this course will:

* be able to design and write readable and correct small programs to solve practical data processing problems;
* be able to read, understand and debug small computer programs;
* understand some practical limitations on computer programs, including scaling (wrt time and memory) and numeric precision (rounding errors) issues.

About you: students in the course Sem. 1




100.0
80.0
60.0
40.0
20.0
0.0
2015
2016
2017
2018
2019
2020
2021
None
Very little
Basic know.
Good know.
Expert

Course info & contacts

* cs.anu.edu.au/courses/comp1730/
* Wattle for forums, quizzes, surveys, assignment submission.
* *Read the news & announcements*!
* To ask a question:
  - Use the discussion forum on wattle.
  - Teams channel for labs and after lectures.
  - For *personal* questions, use the course email: comp1730@anu.edu.au.
  - Always *use your ANU email*.

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 any on-line forum.

Schedule overview

* Two lectures / week.
  - Some live & recorded, some pre-recorded.
  - Follow content & schedule on the course web site, and read the news & announcements.
* One 2-hour lab / week (starting from week 2).
  - Answer the lab time preference survey on wattle – you have until midnight on Wednesday!
  - Changes to the initial allocation will be possible to the end of week 2.
* You are expected to spend another 6 hours to study the course (e.g., solving all lab exercises).

Assessment scheme (preliminary)

* 5 small homework assignments (15%)
* 1 larger project assignment (35%)
* Final exam (50%)

S. Week	
3	Homework 1 due (Monday) In lab: Questions on Hw 1
4	Homework 2 due (Monday) In lab: Questions on Hw 2
6	Homework 3 due (Monday) In lab: Questions on Hw 3
	Break
7 or 8	Homework 4 due (Monday) In lab: Questions on Hw 4
9	Homework 5 due (Monday) In lab: Questions on Hw 5
12	Project due
Exam period	Final exam(s)

* The complete assessment scheme is on the course web site at `cs.anu.edu.au/courses/comp1730/assessment`.
* Note: "*any submitted work may be subject to an additional oral examination*", which can change the assessment mark in any way.
* The assessment scheme (items and weights) will be final at the end of week 2. (Dates can change after that.) Any changes will be announced through the course web page and news forum.

* All assignment deadlines are hard – no late submissions will be accepted.
* Read

  `www.anu.edu.au/students/program-administration/assessments-exams/`

  regarding deferred assessments and special consideration.

Academic honesty

* Discussing programming problems and ways to solve them with other students is a great way to learn
  – just don't discuss assessment problems.
* Homeworks are *individual*. You must write your own code, and be able to show that you understand every aspect of what you have written.

* The project assignment may be done in small groups.
  - Collaboration (including copying solutions) between groups is *not* permitted.
  - The assignment will also have an indvidual component, which you must do by yourself.
* The final exam is *individual*. You may not discuss the exam questions or your answers with anyone (this includes any on-line forum).
* Any academic misconduct will leave a record on student file or even appear on your transcript.
* If you are unsure, please ask your tutor or convenors.

Studying remotely

* All course material is available online.
* Lab groups will run online in Microsoft Teams or in-person at CSIT labs.

Important TODOs

* Complete the **demographic information questionarie**.
* Complete the **lab time preference** survey.
  - You have until midnight on Wednesday!
  - A preliminary allocation to lab groups will be posted later this week.
  - You will have a limited time to change your lab.
  - Labs only start in semester week 2.
  - In-lab assessment starts in semester week 3.

Important TODOs

* Prepare for the labs! Watch recorded lectures, read lab instructions, and attempt some of the exercises before attending your lab.
* Make sure you have a working python programming environment:
  - install Anaconda (spyder) on your own computer; or
  - install another python3 implementation; or
  - verify that you can reliably use the VDI.

  Read `cs.anu.edu.au/courses/comp1730/labs` for more information.

Student course representatives

* Course representatives:
  - point of contact for fellow students who have issues/comments that they are not comfortable to raise with convenor directly;
  - participate in the SRC meetings a few times per semester.
  - Reps are encouraged to provide collective feedback directly to the convenor/lecturer.
* Interested? Write to `comp1730@anu.edu.au` or talk to us after the lecture.