COMP1730/COMP6730
Programming for Scientists

Introduction and Administrative Matters

Communication

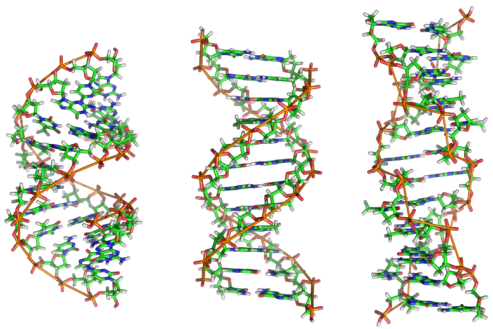
* Read the news forum on Wattle
* Particularly the weekly notice
* Wattle discussion forum for questions about the course content
* comp1730@anu.edu.au for personal matters

Lecture outline

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

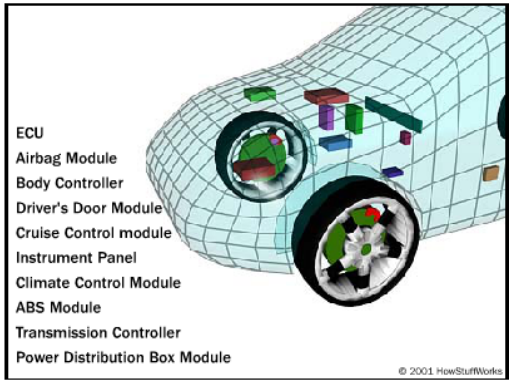
Why learn programming?

* Science rests on data. . . *more and more data*.
  - The Australian SKA Pathfinder radio telescope outputs 2.5GB/s (the SKA is expected to be around 100 times more).
  - A human genome (around 3 billion base pairs) can be sequenced in 3 days.

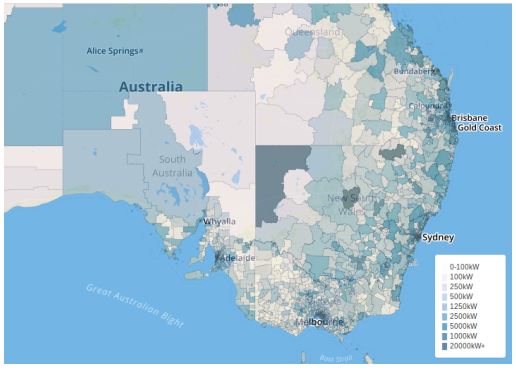


* Processing this data needs software.

* Technical systems increasingly run on software.
  - A modern car has over 30 computers running more than 10,000,000 lines of code.



* Simulation and optimisation are needed for large-scale design questions.
  - Intermittent renewables accounted for around 8.25% of Australian energy generation in 2017. How do we design the power grid to work with 100%?



"*Whatever branch of engineering you're in, make sure you know how to program.*"

(Chris Culbert, NASA Chief Technologist)

* As a 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 $300,000 home loan...
  - use one of the on-line calculators?
* ...for all loan terms in 10-25 years, and an interest rate of 5.5%, 6.5% or 7.5%.
* 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_interest_rate = interest_rate/12
    # interest rate is given in % so need to divide by 100
    r = monthly_interest_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(300000, 5.5, y) for y in years]
mpl.plot(years, mc, 'g-')
mc = [monthly_cost(300000, 6.5, y) for y in years]
mpl.plot(years, mc, 'b-')
mc = [monthly_cost(300000, 7.5, 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 has been consistently ranked in top 5 most popular programming languages,
* particularly for science and engineering uses.
* Open source, available on most platforms.
* Many packages:
  - over 200 in the python standard library;
  - over 100,000 on pypi (`pypi.python.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.

Expectations from this Course

* This course is not an in depth coverage of everything you can do in Python.
* You will not be an expert programmer at the end of this course (unless you are one already).
* The focus in this course is on getting you up to speed on how do to something. Please ask me (or your tutor) if you want more information about a particular topic.
* You may not appreciate the importance of some of the things we talk about in this course until your project (or team) gets large.

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.
  - For *personal* questions, use the course email: comp1730@anu.edu.au.
  - Catch-up labs in person (HN 1.23) and online on Fridays 11am-1pm and 1pm-3pm. Starting in Week 2.

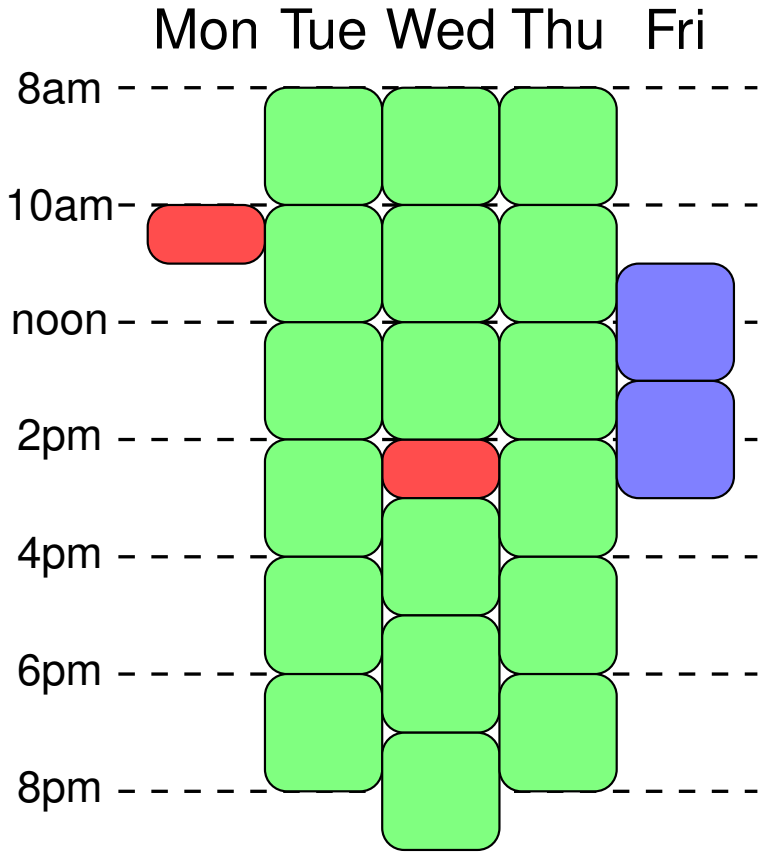
Discussion forum – 3 simple rules

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.

Schedule overview



* 2 lectures / week.
* 1 2-hour lab / week (from week 2).
* Two catch-up/consultation labs each week.

* Except as detailed in the assessment scheme, attendance is never mandatory.

Assessment scheme (preliminary)

* 5 small homework assignments (15%)
* 1 larger project assignment (25%)
* Final exam in 1 or 2 parts (60%)

S. Week	
4	Homework 1 due (Monday) In lab: Questions on HW 1
5	Homework 2 due (Monday) In lab: Questions on HW 2
6	Homework 3 due (Monday) In lab: Questions on HW 3
	Break
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 (Monday)
Exam period	Final Examination(s)

* The complete assessment scheme is on the course web site at `cs.anu.edu.au/courses/comp1730/assessment`.
* The assessment scheme will be final at the end of week 2. Any changes will be announced through the course web page and news forum.
* All assignment deadlines are hard – no late submissions will be accepted.
* See `www.anu.edu.au/students/program-administration/assessments-exams/` regarding deferred assessments and special consideration.

Examinations

* The course has a final examination.
* The examination may be split into two parts.
* The final examination is a hurdle - you must score at least 40% in it to pass the course.
* More details about the final examination will be made available closer to the time.

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 individual 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, of course, in any on-line forum).
* Any academic misconduct will leave a record on your student file that will never go away.
* If you are unsure, please ask your tutor (or me).

Studying Remotely

* All course material is available online.
* All lab groups will run online in Microsoft Teams.
* Catch-up labs will run both in person and online.
* All students have access to CodeBench (starting from lab 2).
* Instruction videos on using Teams and CodeBench will be made available in Wattle.

Important TODOs

* Complete the **demographic information questionnaire**.
* **Sign up to a lab group**.
  - If there is no place free in any lab at any time that you can attend:
    > don't sign up to a group you cannot attend;
    > email `comp1730@anu.edu.au` with your ANU ID, a complete list of all groups that you can attend, and any preference.
  - Labs only start in semester week 2.
  - In-lab assessment starts in semester week 4.