Programming as Problem Solving

COMP 1100

Semester 2, 2024

Ranald Clouston

The Australian National University

comp1100@anu.edu.au

Acknowledgement of Country

We acknowledge and celebrate the First Australians on whose traditional lands we meet, and pay our respect to the elders past and present. I would also like to acknowledge and welcome any other Aboriginal and Torres Strait Islander people who are enrolled in our courses.

You can learn more about Acknowledgements of Country here.

COMP1100/1130 Team

- Course convener
  - Ranald Clouston
- Lecturer
  - Roger Su
- Tutors

  Ishaan Kapoor

  Jess Allen

  Malcolm Macdonald

  Peter Oslington

  Madeleine Stewart

  Liz Yevdokimov

What is a computer?

Photo c/o Early Office Museum Archives

Dorothy Vaughan 1910-2008

- Trained as a mathematician, worked as a teacher
- Joined Langley Memorial Aeronautical Laboratory (later part of NASA) in 1943 as a **computer**
- Rose to head of West Area Computing by 1949, making important contributions to data analysis for the development of space travel
- Reinvented herself as a programmer of electronic computers, using FORTRAN, in early 1960s

- Background reading: Margot Lee Shetterly, `Hidden Figures' (2016) – also a film!; Short NASA biography

Human computers vs electronic computers

- Human computers provided with an **algorithm** written in natural language (e.g. English) and/or mathematics, along with data
- Human computers able to use common sense, domain knowledge, mathematical knowledge

- Electronic computers far faster at accurately processing large amounts of data but…
  - Only understand very limited, inflexible, notion of `language'
  - No common sense or domain knowledge
  - So more burden on (and fun for!) the programmer – writing and testing

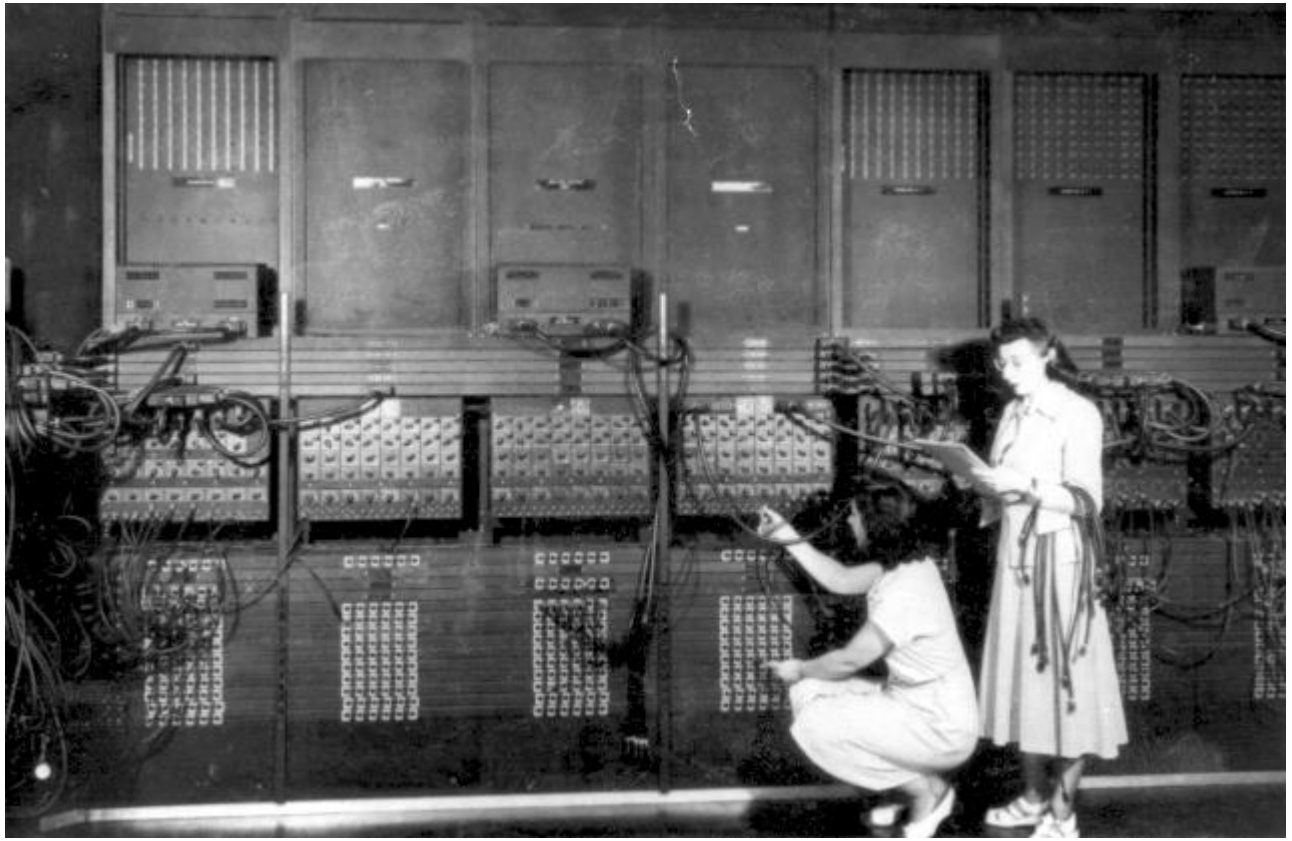
The First Programmers

- **Ada Lovelace** (1815-52) is usually considered the first programmer
  - But Charles Babbage's steam-powered `Analytical Engine', for which she designed programs, has yet to actually be built!

- More directly historically significant is the **ENIAC**, the first 'computer' as we now understand the term (1945)…

The First Programmers

- First working programmers: Kathleen McNulty, Frances Bilas, Betty Jean Jennings, Ruth Lichtermann, Elizabeth Snyder, Marlyn Wescoff



- Read more: Jennifer S. Light "When computers were women." *Women, science, and technology: A reader in feminist science studies* (2013)

Photo c/o the United States Army Military Laboratory

Machine level programming

Not portable: depends on specific CPU model ❌

Error prone ❌

Hard to handle complex problems ❌

Machine dictates the language in which the programmer thinks ❌

Full control ✅

Need more problem-oriented languages.

Need more abstract and safer languages.

Abstraction versus control is a tradeoff!

Memory Address	Machine Code (hex)	Assembler Instructions	(Poorly coded) C program
0x100000e9b	c7 45 f0 00 00 00 00	movl $0x0, -0x10(%rbp)	int sum = 0;
0x100000ea2	c7 45 ec 00 00 00 00	movl $0x0, -0x14(%rbp)	int i = 0;
			loop:
0x100000eb9	8b 45 ec	movl -0x14(%rbp), %eax	if (i >= length(values))
0x100000ebc	3b 45 f4	cmpl -0xc(%rbp), %eax	
0x100000ebf	0f 8d 21 00 00 00	jge 0x100000ee6	goto done;
0x100000ec5	48 63 45 ec	movslq -0x14(%rbp), %rax	sum += values[i];
0x100000ec9	48 8b 4d f8	movq -0x8(%rbp), %rcx	
0x100000ecd	8b 14 81	movl (%rcx,%rax,4), %edx	
0x100000ed0	03 55 f0	addl -0x10(%rbp), %edx	
0x100000ed3	89 55 f0	movl %edx, -0x10(%rbp)	
0x100000ed6	8b 45 ec	movl -0x14(%rbp), %eax	i = i + 1;
0x100000ed9	05 01 00 00 00	addl $0x1, %eax	
0x100000ede	89 45 ec	movl %eax, -0x14(%rbp)	
0x100000ee1	e9 d3 ff ff ff	jmp 0x100000eb9	goto loop;
			done:
0x100000ee6	8b 45 f0	movl -0x10(%rbp), %eax	mean = sum / i;
0x100000ee9	99	cltd	
0x100000eea	f7 7d f4	idivl -0xc(%rbp)	

High level programming languages

Written in a style more appealing to humans

Text needs to be transformed into machine code
(or into something that can in turn be transformed)

Interpreter: executes code line-by-line

Compiler: transforms whole program

You will see both in this course for a language called **Haskell**

Photo of Grace Hopper c/o biography.com

Part One

Course(s) overview

COMP1100

- The **entry point for study in Computer Science**.
- A broad course aimed at many different backgrounds and destinations.
- Teaching the fundamentals of algorithms and programming as means of solving problems.
- Introduces many concepts that will be revisited in more depth later in the CS curriculum.
- Uses the programming language Haskell.

COMP1100 is not…

- Designed for students who wish to do one or two computer science courses during their degree to gain some useable programming skills
  - If that describes you, COMP1730 is **strongly recommended**
  - If you want to do two such courses, 1730 -> 1110 is an option
  - You may use 1100 as a taster of ‘real’ computer science…
  - But understand that the course is designed for students who will do multiple such courses

- Designed to teach general computer / IT skills
  - We are focused on **programming**

Programming Experience

- Many of you will have programmed before, e.g. in high school, or as a hobby.
- Many of you will not have.
- Prior programming is **not** expected in this course.
- Nor, in our experience, is it a major advantage: we focus on the principles and foundations of programming in a way that is very different from most students' experience.

Mathematical Maturity

- We **do** expect students to be comfortable with mathematical thinking
  - abstraction, algorithms, attention to detail
- Building these mathematical skills at the same time as picking up programming skills can be a challenge for a minority of students.
- Students whose mathematics is weaker could consider delaying COMP1100 by one semester, to take a course like MATH1005 (Discrete Mathematical Models)

Exiting the Course

- Easy to do in the first week, picking another course up.
- 31 August ‘Census date’ deadline for withdrawal without fees.
- 4 October deadline for withdrawal without failure.
- **Visas** – it is **not always true** that withdrawal from a course without replacement will violate your visa; talk to CECC Student Services about this if you need information.

Part Two

How do I interact with this course?

2A. How do I get basic information about the course?

2B. How do I get the content of the course?

2C. How do I practise my skills?

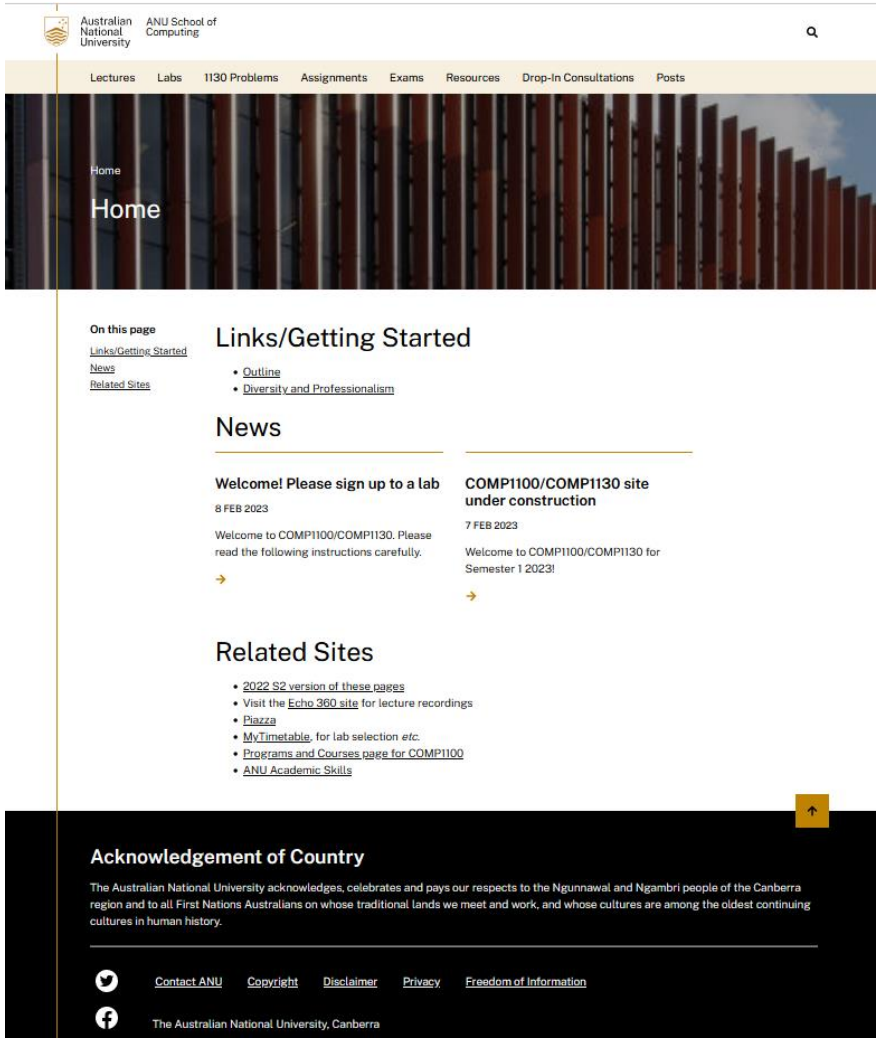
2D. How do I get answers to my questions?

Getting Information: Course Website

https://comp.anu.edu.au/courses/comp1100/

- We do *not* make much use of Wattle

We *strongly* recommend that you click on the ‘Course Outline’ tab



Getting Content: Lectures

- The start of the learning journey; introduces the core concepts of the course, with 'live coding' to show you programming in action.

Lecture recordings

- https://echo360.org.au
- Please inform convenors urgently if a recording is inadequate
- Note that most students find that watching a recording is an inferior experience to a live lecture, so make your attendance choices wisely

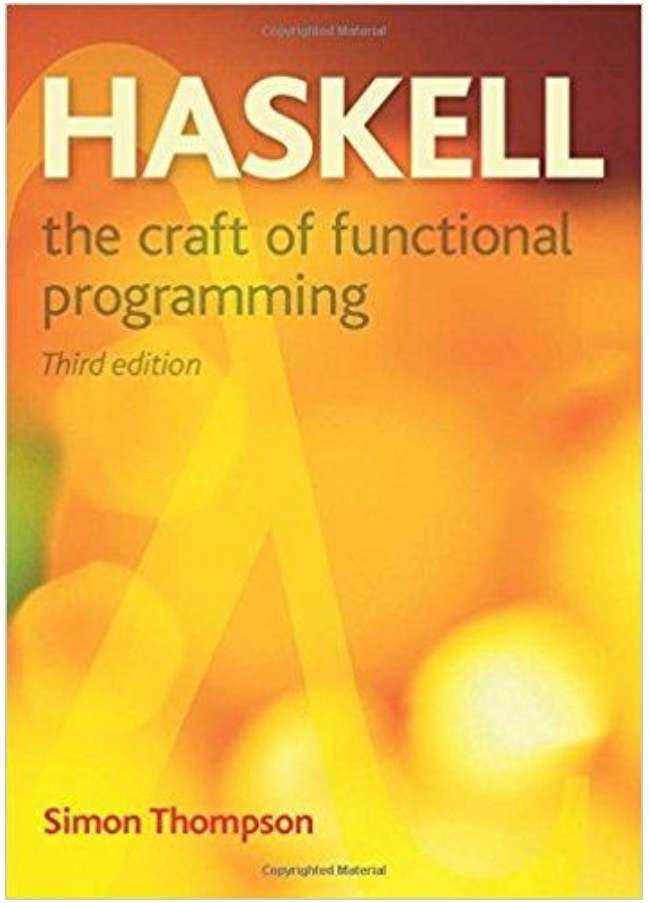
Getting Content: The Textbook

Thompson, Simon

HASKELL: the craft of functional programming

Third edition

- Available free and legally!
- **Not** required
- Can be purchased, and copies in the library



Practising Skills: Labs

- You will learn more, and have more fun, in labs than any other part of the course!
- Experience suggests you are *very unlikely* to prosper in this course without regular lab participation
- Tutors there to answer questions and guide your learning.
- Sign up **now** at https://mytimetable.anu.edu.au/even/student.
- 3% of the course linked to attendance and participation in the labs, and to submission of attempts at lab exercises.

Practising Skills: Beyond Labs

- Assignments
- Hundreds of exercises in the textbook.
- Past exam questions on the website.
  - Warning: the course changes over time. Past exams may contain material you are not expected to know, and may miss material you are expected to know.

Getting Questions Answered

- Live questions in lectures extremely welcome!
- Tutors also there to answer questions in labs.
    - But these questions should be related to the current content.
    - How to get **your** questions answered?

- Ed Discussions Forum
- Drop-In Consultations
- Email (in **very rare** circumstances)

Ed Discussions Forum

https://edstem.org/au/courses/18356/discussion/

Your participation is very welcome

- Asking questions is great, but you could also try answering!
- Unless urgent or relating to course logistics, we usually give time for students to have their say

Please read the Ed Posting Etiquette pinned post before making any posts

Drop-In Consultations with Tutors

Possible times in MyTimetable

- But number of sessions per week will depend on final enrolment numbers

Unstructured time for you to get your questions responded to by experts

Starts week 3

Getting Questions Answered: Email

- Email your tutor *only* if you have questions specific to your interactions with that tutor (about lab attendance, assignment marking etc.)
- Email the lecturers, instead of using Ed Discussions, *only* if you have an issue that tutors could not help with or should not see.
    - Use comp1100@anu.edu.au (Ranald and Roger) *only*
    - Use your ANU address
- For all enquiries not directly related to this course, contact CECC Student Services (e.g. email studentadmin.cecc@anu.edu.au).

Part Three

Course Representatives

•CECC

•CLASS REPRESENTATIVES

Class Student Representation is an important component of the teaching and learning quality assurance and quality improvement processes within the ANU College of Engineering and Computer Science (CECC).

Each semester, we put out a call for Class Representatives for all ANU College of Engineering, Computing and Cybernetics (CECC) courses. Students can nominate themselves for one or more of the courses they are enrolled in.

Australian National University

• Roles and responsibilities:

- The role of Student Representatives is to provide ongoing constructive feedback on behalf of the student cohort to Course Conveners and to Associate Directors (Education) for continuous improvements to the course.

- 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.

- Note: Class representatives will need to be comfortable with their contact details being made available via Wattle to all students in the class.

- For more information regarding roles and responsibilities, contact:
  ANU College of Engineering, Computing and Cybernetics
- ANUSA CECC representatives (sa.cecc@anu.edu.au).

• Why become a class representative?

- **Ensure students have a voice** to their course convener, lecturer, tutors, and College.
- **Develop skills sought by employers**, including interpersonal, dispute resolution, leadership and communication skills.
- **Become empowered**. Play an active role in determining the direction of your education.
- **Become more aware of issues influencing your University** and current issues in higher education.
- **Course design and delivery.** Help shape the delivery of your current courses, as well as future improvements for following years.

Want to be a class representative?
Nominate today!

Please nominate yourself to your course convener by end of Week 2

Class Representatives in COMP1100/1130

- We are looking for 2 course representatives!
- Looking for *diverse* representation – male and female, Australian and international, different degrees etc.

- If you are interested, please submit your application to comp1100@anu.edu.au before Monday week 2.
- Briefly write why you might be a good course representative. Describe your background or anything that might be relevant.
- We will announce the course representatives in week 2 or 3.