COMP1531

🐶 Software Engineering
1.1 - Overview

Why does COMP1531 matter?

Why does COMP1531 matter?

COMP1531 is the course for you to transition from a **solo programmer** to a **collaborate software engineer**. By the end of this course you will:

- Understood how software operates in large projects by building a larger scale software system
- Have the skills in order to write and manage code with a group of people
- (Bonus) Be familiar with the basics of Python and web applications

Core Content

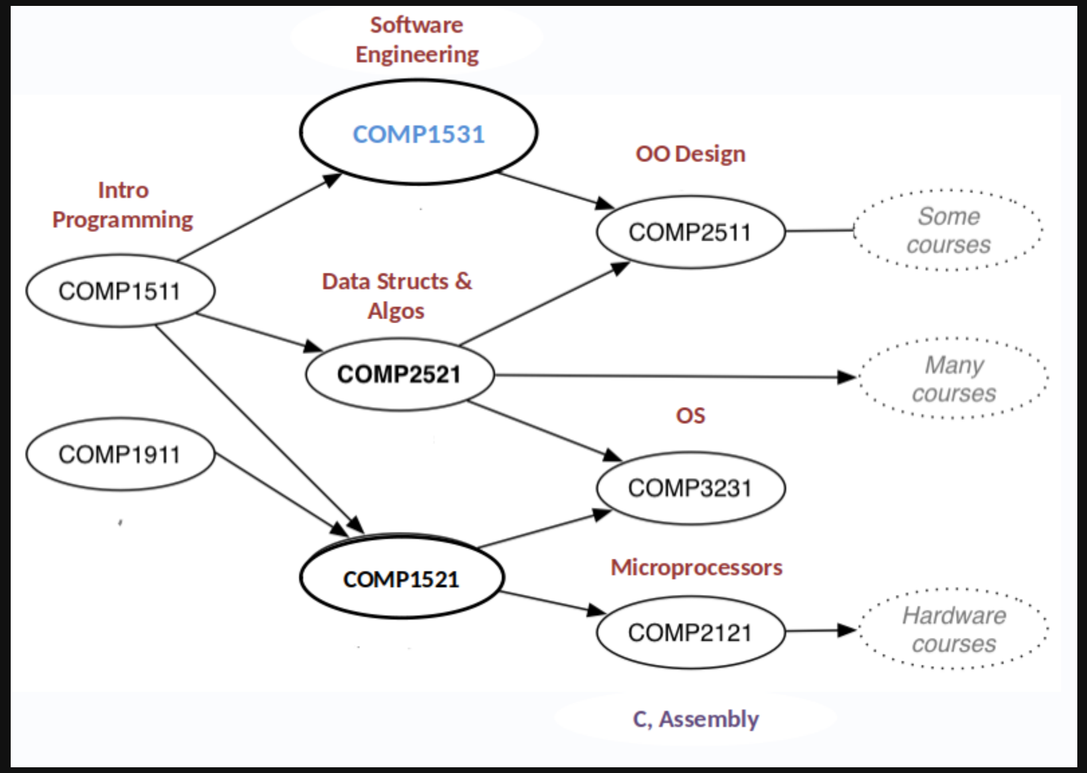
Software Engineering & the SDLC	Git & Project Management & Teamwork
Python	Web & HTTP

This course covers a broader breadth of knowledge than COMP1511, so learning about multiple topics concurrently throughout term is normal.

Software Engineering

- COMP1511: Learning programming by writing code to solve problems
- COMP1531: Learning software engineering by using programming in teams to build user-facing applications by applying the software development lifecycle

Relevance to your program



Can we also say hello to our 130 Mechatronic Friends!

Assumed Knowledge

That you are at least a mediocre C programmer

I.E. You have done COMP1511

- Control structures
- Data types
- Abstraction
- Testing

What is a software engineer

What's the difference between **Computer Science** and **Software Engineering?**

What is a software engineer

- What's the difference between **Computer Science** and **Software Engineering?**
  - At UNSW, Software Engineering is an extension of Computer Science, where we give extra focus to how software systems are built, how to manage projects, and how to test software to provide quality assurance.
- Do you need to be a **Software Engineering student** to be employed as a **Software Engineer?**

What is a software engineer

Applying engineering methodologies to our current programming capabilities.

IEEE definition: "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software."

We're going to build thought-out, testable, scalable software that is meets set out requirements and is easily maintained

That was "What" - but "Why"?

What happens in a world without
good software engineering
principles being used?

That was "What" - but "Why"?

Software engineering fundamentally exists to allow **businesses** and **organisations** to de-risk their business goals compared to just hacking away.

- More predictability about time and budget
- Minimise errors and increase reliability

Software engineering adds small overheads through the software development process to provide higher assurances overall.

Bad things can happen

Assessment

Item	Weighting	Notes
Class Mark	15%	Tutorials + Labs
Project	50%	3 iterations
Exam	35%	Programming + Theory

Teaching Strategies

- Lectures
- Tutorials
- Labs
- Major Project
- Help Sessions
- Exam

Teaching Strategies | Lectures

- 2 x 2 hours per week
- Schedule listed here, showing live stream links to Youtube (to watch them live)
- Slides for the lectures, and the recordings uploaded later, found on the course work page.

Teaching Strategies | Tutorials & Labs

- Tutorial and lab schedule and meeting links/locations can be found on the timetable.
- Tutorial and lab content can be found on the course work page.
- Tutorials and labs contribute to your class mark (see course outline).
- Labs need to be submitted on the Monday the following week (e.g. week 1 lab due Monday week 2)
- Labs need to be demonstrated in the following week (e.g. week 1 lab must be demonstrated in week 2)

Teaching Strategies | Major Project

- You will work from weeks 1-10 with a group of 4-5 on a major software project
- Groups are formed throughout your tutorial in week 1
- This project will be released on Friday evening of week 1
- Major project information will be posted here.

Teaching Strategies | Help Sessions

- Help sessions are online "drop-in" sessions where you or your group can get further assistance outside of class time.
- Begin in week 2
- Schedule listed here

Teaching Strategies | Exam

- Final exam will tentatively be open book and online
- There is no hurdle requirement
- There are no mercy supps
- Details about the final exam will be shared here closer toward the end of teaching term

Getting Help

If you need help throughout the course, seek it in this order:

- Step 0: Your team
- Step 1: EdStem forum (Webcms3 sidebar)
  - Look for answers before posting
  - You were invited "z5555555@unsw.edu.au"
- Step 3: Help Sessions
- Step 4: Emailing Tutor / Assistant Tutor
- Step 5: Lecturers cs1531@cse.unsw.edu.au

Getting Setup

- Any operating system is fine for this course.
  - **CSE Machines (VLab / SSH)**: Ready to go!
  - **Linux**: Simply install Git
  - **Mac OSX**: Simply install Python & Git
  - **Windows**: We recommend you install WSL and then install git/python like it's linux

Getting Setup

- In terms of technologies you have to use:
  - Webcms3 (your home!)
  - EdStem (forum)
  - Zoom (for tutorials/labs)
  - Collaborate/Moodle (for help sessions)
  - Gitlab (for all of your work!)
  - Microsoft Teams (for project communication)
- In terms of software languages we use:
  - Git (install info here)
  - Python (install info here)

Software Development Life Cycle (SDLC)

SDLC

1. Requirements Analysis

- **Understand the problem you are trying to solve**
- Analyse and understand problem domain
- Determine functional and non-functional components
- Generate user stories / use cases
- Arguably the most important part of software engineering

SDLC

2. Design

- Producing software architecture/blue-prints
- System diagrams and schematics
- Modelling of data flows
- Happens ideally before you write any code
- Means that when you write code

SDLC

3.Development

- Choose a programming language and write the code

4. Testing

- Use unit or behaviour tests to test your software
- Automating testing for every code change

SDLC

5. Deployment

- Make the software available for use by the users

6. Maintenance

- Monitor the system, track issues, interview users to reassess requirements (start again!)

SDLC in COMP1531

In this course we touch on every part of the SDLC at different points, with a general focus in order of:

- Testing
- Design
- Requirements

6. Maintenance

- Monitor the system, track issues, interview users to reassess requirements (start again!)

Feedback