COMP2511
Object Oriented Design & Programming

Course Introduction

Term 2, 2021

Our Team

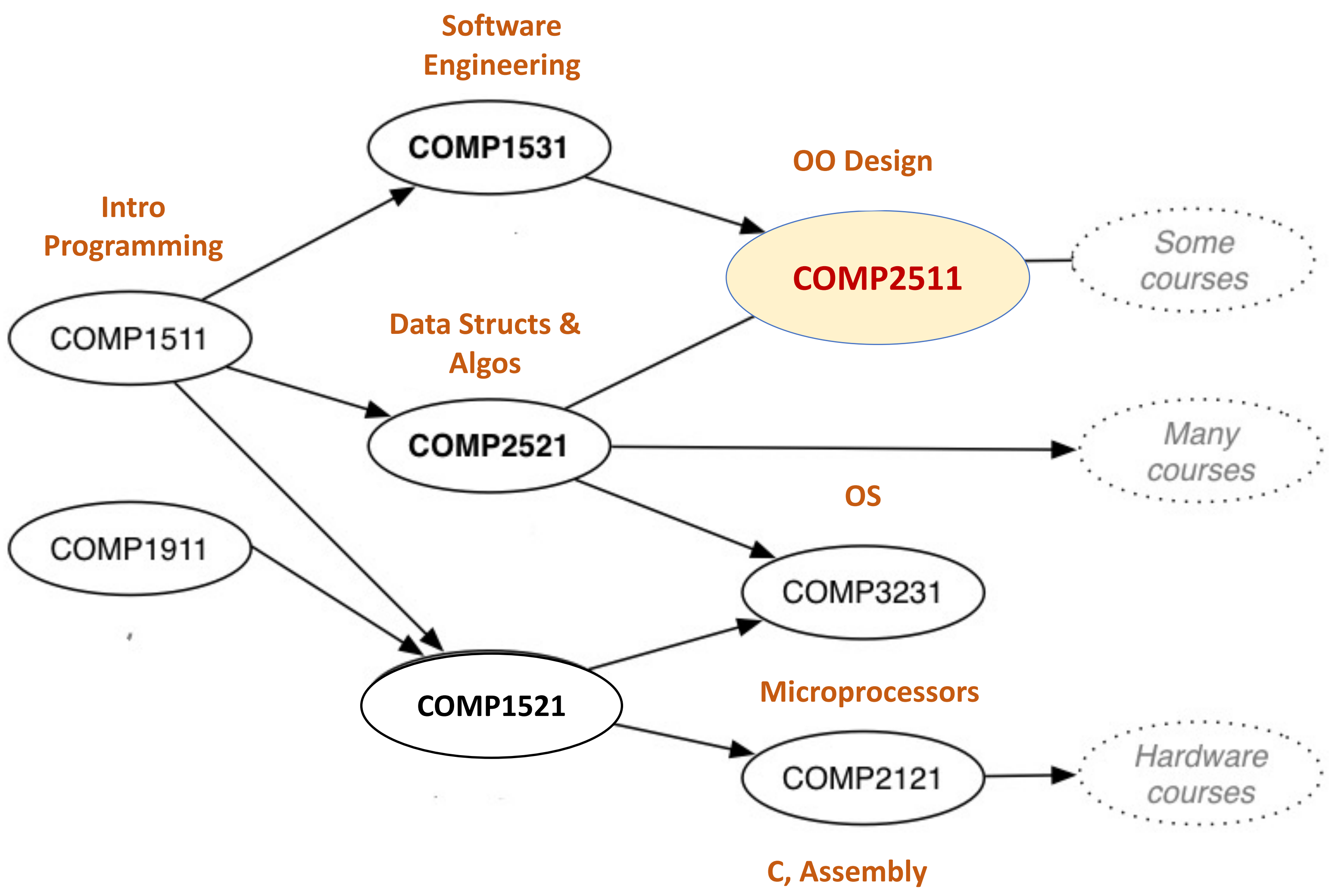
Lecturer-in-charge:
Dr Ashesh Mahidadia <ashesh@cse.unsw.edu.au>

Course Admin Team:
Braedon Wooding <b.wooding@unsw.edu.au>,
Nick Patrikeos <n.patrikeos@unsw.edu.au>,
Matthew Perry <matthew.perry@unsw.edu.au>

Course Account Email: **cs2511@cse.unsw.edu.au**
(Unless you specifically require to contact a member of the admin team, please use the ***above email*** *for any queries related to the course.)*

Class Web:
http://webcms3.cse.unsw.edu.au/COMP2511/21T2/

Course Context



Pre-requisites

- Completed **COMP 1511**
  - competent C programmers who can understand and use abstract data types
- Completed **COMP 1531**
  - understand and know how to use git and Github reasonably
  - be familiar with fundamental object-oriented design concepts

COMP 2511 Major Themes

❖ **Object Oriented Design Process (SE)**

- Understand the principles of object-oriented design
- Be able to follow a systematic object-oriented design process
- Be able to interpret and use tools for object-oriented design
- Learn how to apply design principles and design patterns effectively to design flexible, maintainable and reusable systems

COMP 2511 Major Themes

- **Object Oriented Programming in Java**
  - Be able to write medium-scale object-oriented programs in Java

- **Software Engineering process**
  - Problem solving – apply SE principles to solve a real-world problem
    - Be able to work within a small team in the context of a software development project
    - Be able to plan and execute a software project according a systematic software process

- **User Interfaces**

  - Effective coding and testing techniques

Credit teaching material

- No text book, the lecture slides cover the required topics.

- However, you are strongly encouraged to read additional material and the reference books.

- In the lecture notes, some content and ideas are drawn from:
  - *Head First Design Patterns* , by Elisabeth Freeman and Kathy Sierra, The State University of New Jersey
  - *Refactoring: Improving the design of existing code* , by Martin Fowler

- Links to useful tutorials will be uploaded as necessary

How do we obtain our educational objectives?

- **Lectures:** 4 hour lectures (9 weeks)

- **Tutorials:**
  - A 1 hour tutorial session per week, which is scheduled before the lab.
  - Tutorials/Labs will be run via **MS Teams** .
  - Tutorials contribute to your class marks.
  - You will receive mark out of one for each tutorial. A number of factors contribute toward you receiving a high grade in this area, including but not limited to:
    - Attending the full tutorial
    - Asking good and logical questions throughout the tutorial
    - Taking initiative to answer questions and be engaged
    - Attending with your webcam on for online tutorials (or your phone if you don't have)

How do we obtain our educational objectives?

- **Labs:**
  - 2 hours per week, weeks 1-5 and 7 – 10. Your tutor or lab assistant will be available during your lab time-slot to answer any questions you may have regarding the lab questions.
  - Labs comprise small design and practical programming exercises, individual or in groups.
  - The lab classes will be used for lab **exercises**, project **mentoring** and project **assessments**.
  - You must submit the lab by the following Sunday 5:00 pm.
  - You should demonstrate your lab work to your tutor during the lab or the following week's lab for feedback on your problem solving, approach, the style of your solution and importantly marking Please note that you must attend all nine lab and tutorial sessions during the term.

How do we obtain our educational objectives?

❖ Labs:

- ❖ Labs contribute to your class marks.
- ❖ You will receive marks out of 2 for each lab. A rough guideline for how marks are awarded for your **lab** is below:
  - 2 / 2 marks (completion of all required tasks)
  - 1.5 / 2 marks (satisfactory completion most tasks)
  - 1.0 / 2 marks (did an OK job overall, satisfactory with some errors or items missing)
  - 0.5 / 2 marks (completed some things but did so quite poorly)
  - 0.25 / 2 marks (barely an attempt)

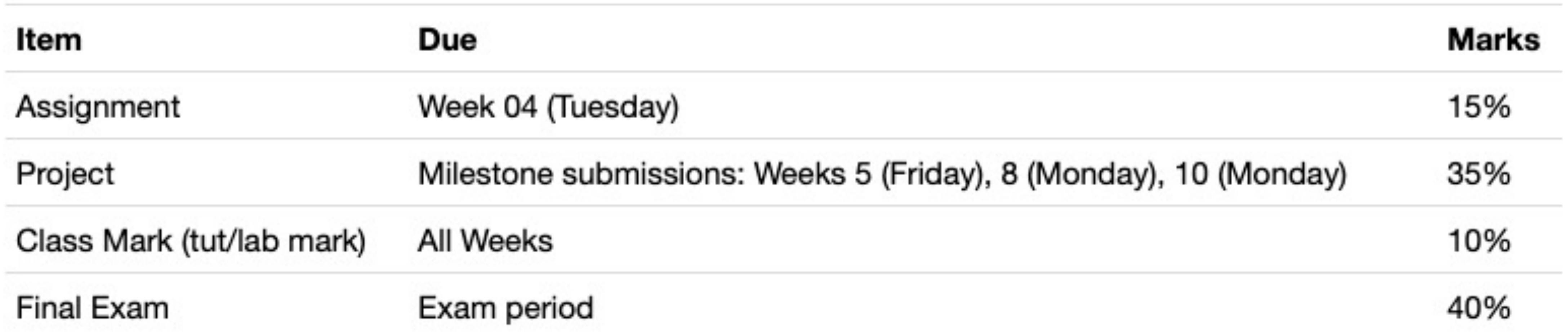
❖ Class Marks (tut/lab marks):

- ❖ Your class mark is made up of marks associated with tutorials, and marks associated with labs.
- ❖ There is a total of 24 marks that can be gained between 8 tutorials and labs throughout the course (i.e. 4 bonus marks), although it will be capped at 20 overall and comprise 10% of your overall course mark.

How do we obtain our educational objectives?

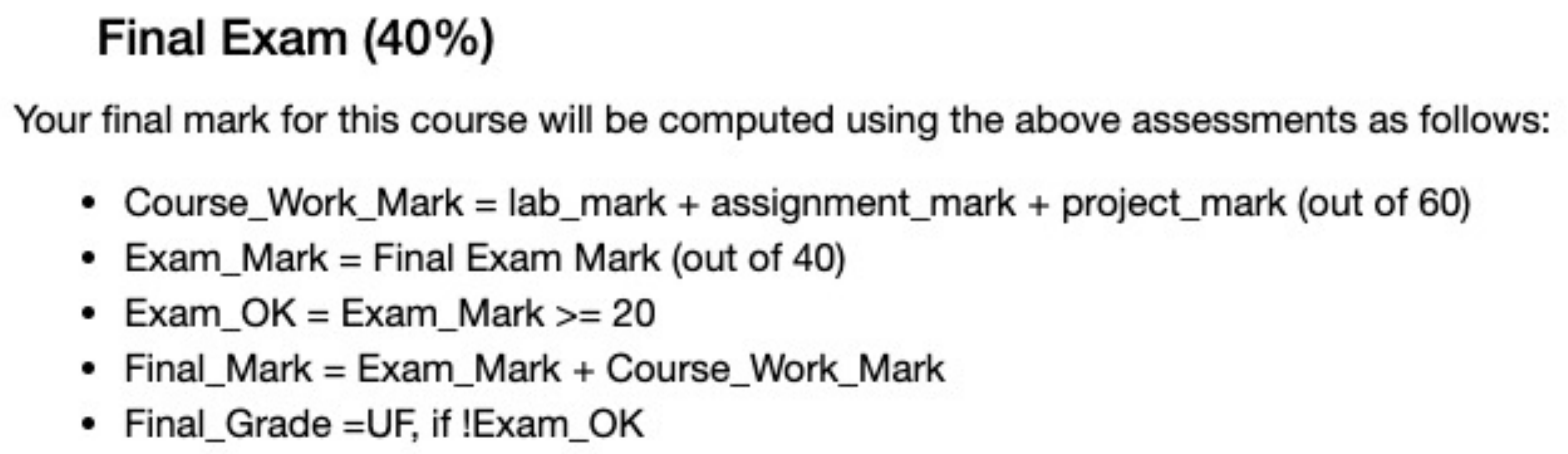
- **A Group Project…**
  - Contributes to 35% of the final course mark.
  - Carried out in teams of **four** .
  - Project specification will be released in Week 04.
  - Implemented using an *Agile Software Development Model :* Working software to be delivered in iterations, marks will be awarded for each iteration demo, which will count towards your overall group project mark. Responsibilities to be assigned to each group member during each iteration and all team members **MUST** contribute equally (*Tutors will check GitLab*)
  - There are **three milestones** in the project.
    - You need to submit milestone 1 in week 5 and provide demo in week 7 for assessment.
    - You need to submit milestone 2 on Monday of week 8 and provide demo in week 8 for assessment.
    - You need to submit milestone 3 on Monday of week 10 and provide demo of the entire project in week 10 (final demo) for assessment.

Assessments



Item	Due	Marks
Assignment	Week 04 (Tuesday)	15%
Project	Milestone submissions: Weeks 5 (Friday), 8 (Monday), 10 (Monday)	35%
Class Mark (tut/lab mark)	All Weeks	10%
Final Exam	Exam period	40%

Course Mark



Final Exam (40%)

Your final mark for this course will be computed using the above assessments as follows:

- Course_Work_Mark = lab_mark + assignment_mark + project_mark (out of 60)
- Exam_Mark = Final Exam Mark (out of 40)
- Exam_OK = Exam_Mark >= 20
- Final_Mark = Exam_Mark + Course_Work_Mark
- Final_Grade =UF, if !Exam_OK

Supplementary Exam

- Students are eligible for a Supplementary Exam if :
  - they cannot attend the final exam due to illness or misadventure
  - successfully apply for a special consideration
    (must apply for a special consideration, and get **approved**)
  - For more information, read *Essential Advice for CSE Students*, the web link is available in the course outline.

https://www.engineering.unsw.edu.au/computer-science-engineering/about-us/organisational-structure/student-services/policies/essential-advice-for-cse-students

System

- Most work done on Linux or Mac
  - Lab work and group project can be done on the CSE machines (using vlab) or your own device
  - Technology stack
    - Java 11 SE,
    - Visual Studio Code
- Collaboration and Versioning Tool - GitLab

Plagiarism

Just don’t do it!