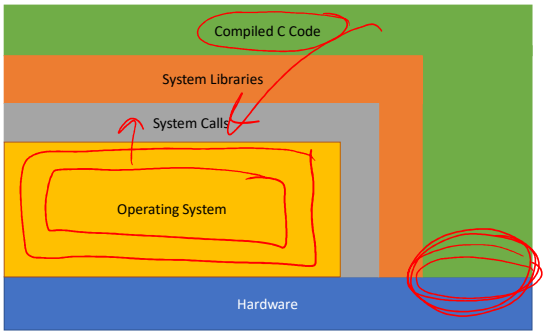
# Welcome to OS @ UNSW

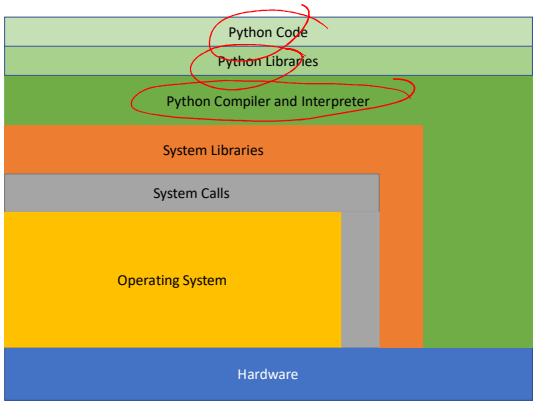
COMP3231/9201/3891/9283
(Extended) Operating Systems
Dr. Kevin Elphinstone

1

## System Software Structure

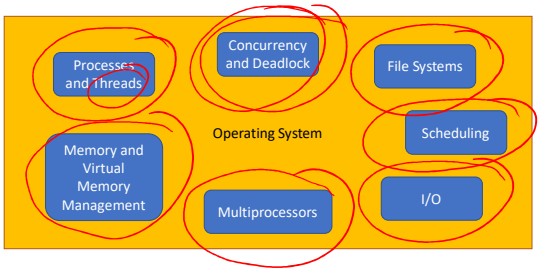



2




3

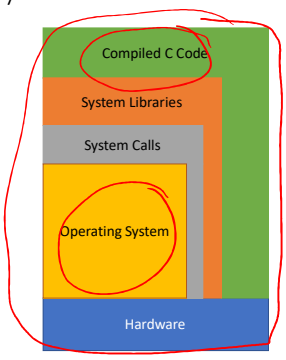
## Major OS Topics




4

## Why Learn Operating Systems?

- Understand the whole software stack
- Develop OS code
- Develop concurrent code
- Application performance
  - Understand operating system behaviour and how best to interface with it.
  - Diagnose system performance issues.




5

## How will we learn about Operating Systems?

**Lectures**
- Introduce OS theory and case studies

**Tutorials**
- Re-enforce theory
- Provide guidance on the assignments

**Assignments**
- Opportunity to write real OS code
  - OS/161 is a simplified UNIX-clone intended for teaching
- Consist of the following
  - Warm-up exercise
  - Concurrency and synchronisation
  - OS Structure involving system calls and file system
  - Memory management

6

## Intended schedule*

- Lectures
  - Weeks 1-5, 7-9
- Tutorials
  - Weeks 2-5, 7-10
- Assignments
  - ASST0 – Week 2
  - ASST1 – Week 4
  - ASST2 – Week 7
  - ASST3 – Week 10/11

* Subject to change

7

## Overview of Course Outline



8

## Pre-requisites

- COMPXXXX Data structures and algorithms
  - Stacks, queues, hash tables, lists, trees, heaps,….
- COMPXXXX Microprocessor and Interfacing or Computer Systems Fundamentals
  - Assembly programming
  - Mapping of high-level procedural language to assembly language
  - Interrupts



9

## Assumed Knowledge

- Computing Theory and Background
  - Basic computer architecture
    - CPUs, memory, buses, registers, machine instructions, interrupts/exceptions.
  - Common CS algorithms and data structures
    - Links lists, arrays, hashing, trees, sorting, searching…
  - Ability to read assembly language
  - Exposure to programming using low-level systems calls (e.g. reading and writing files)
- Practical computing background
  - Capable UNIX command line users
  - Familiar with the git revision control system
  - Competent C programmers
    - Understand pointers, pointer arithmetic, function pointers, memory allocation (malloc())
    - The dominant language for OS (and embedded systems) implementation.
  - Comfortable navigating around a large-ish existing code base.
  - Able to debug an implementation.

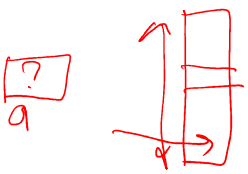


10

## Why does this fail?

```
void set(int *x)
{
    *x = 1;
}
void thingy()
{
    int *a;
    set(a);
    printf("%d %d\n",*a);
}
```

Operating System Coding

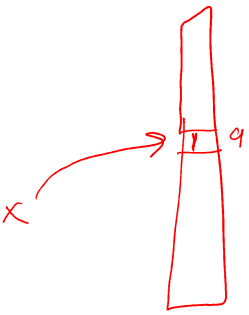






11

## Why does this fail?

```
void set(int *x)
{
    *x = 1;
}
void thingy()
{
    int a;
    set(&a);
    printf("%d %d\n",a);
}
```



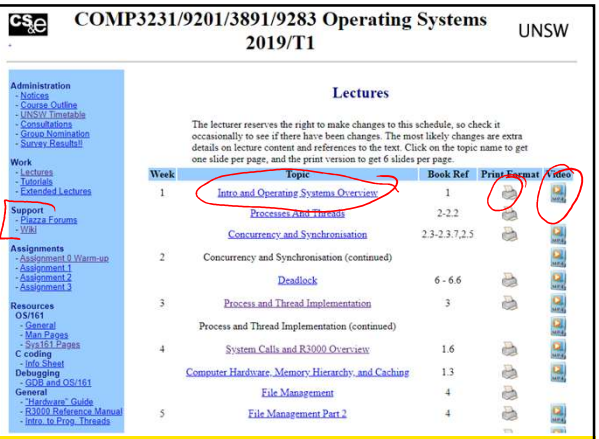



12

## Lectures

- Common for all courses (3231/3891/9201/9283)
- The lecture notes will be available on the course web site
  - http://www.cse.unsw.edu.au/~cs3231
    - Available prior to lectures, when possible.
    - Slide numbers for note taking, when not.
  - The lecture notes and textbook are NOT a substitute for attending lectures.
    - Video might be…..
  - Will attempt to have "video" available, baring technical hitches.
    - Echo360 is a backup

13




14

## Tutorials

- Start in week 2
- Attendance is strongly recommended
  - but not marked.
- Tutorial questions cover a broad range of examples
  - Answers available online the week after.
  - Use the tutorial to focus where needed
    - Review the questions beforehand
    - We'll experiment with prioritising with online polls or similar

15

## Assignments

- Assignments form a substantial component of your assessment.
- They are challenging!!!!
  - Because operating systems are challenging
- We will be using OS/161,
  - an educational operating system
  - developed by the Systems Group At Harvard
    - With local changes.
  - It contains roughly 20,000 lines of code and comments
    - Comments are part of the documentation

16

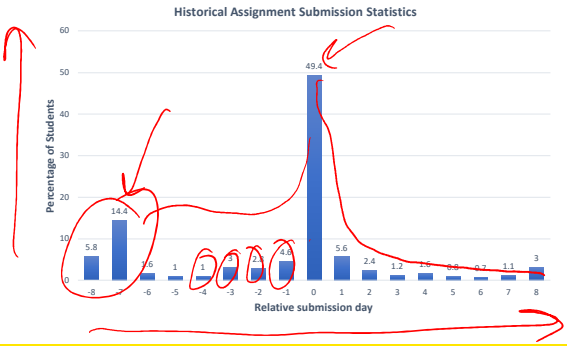
## Assignments

- Don't underestimate the time needed to do the assignments.
  - 80% is understanding
  - 20% programming
- Avoid
  - 1% understanding
  - 9% programming
  - 90% debugging
- If you start a couple days before they are due, you will be late.
- To encourage you to start early,
  - Bonus 2% of awarded mark per day early, capped at 10%
  - See course outline for exact details
    - Read the fine print!!!!

17

## Assignment Submission Times

16% late




18

## Assignments

- Late penalty
  - 4% of total assignment value per day
    - Assignment is worth 20%
    - You get 18, and are 2 days late
    - Final mark = 18 – (20*0.04*2) = 16 (16.4)
- Assignments are only accepted up to one week late. >5 days = 0

19

## Assignments

- Warmup assignment (ASST0)
  - Done individually
  - Available NOW!!!!
- ASST2 and ASST3 are in pairs
  - Info on how to pair up available soon
- Additionally, advanced versions of the assignment 2 & 3
  - Available bonus marks are small compared to amount of effort required.
  - Student should do it for the challenge, not the marks.
  - Attempting the advanced component is not a valid excuse for failure to complete the normal component of the assignment

| Assignment | Due |
|---|---|
| ASST0 | Week 2 |
| ASST1 | Week 4 |
| ASST2 | Week 7 |
| ASST3 | Week 10 |

20

## Assignment 0

- Warm-up exercise due in week 2
  - It's a warm-up to have you familiarize yourself with the environment and easy marks.
    - Practice with git revision control
    - Practice submitting a solution
    - Practice using code browser/editor
  - Do not use it as a gauge for judging the difficulty of the following assignments.

21

## Assignments

```
Submission test failed. Continue with submission (y/n)? y
```

- Lazy/careless submitter penalty: 15%
- Submitted the wrong assignment version penalty: 15%
  - Assuming we can validly date the intended version

22

## Assignments

- To help you with the assignments
  - We dedicate a tutorial per-assignment to discuss issues related to the assignment
  - Prepare for them!!!!!

23

## Group Work Policy

- Groups of two
- Group members do not have to be in the same tutorial
- Group assignments will be marked as a group
  - Including 'groups' of one.
- Group members are expected to contribute equally to each assignment.
  - No "I'll do the 2nd if you do the 3rd assignment"
  - We accept statements of unequal contributions and do adjust marks of the lessor contributor down.
- Submissions are required to have significant contributions attributable to individual group members.
  - E.g. verifiable using the git revision control system

24

## Plagiarism

- We take cheating seriously!!!
- We systematically check for plagiarised code
  - Penalties are generally enough to make it difficult to pass
- We can google as easy as you can
  - Some solutions are wrong
  - Some are greater scope than required at UNSW
    - You do more than required
    - Makes your assignment stick out as a potential plagiarism case
  - We do vary UNSW requirements

25

## Exams

- There is NO mid-session
- The final written exam is 2 hours
- Supplementary exam are available according to UNSW & school policy, not as a second chance.
  - Medical or other special consideration only

26

## Assessment*

- Exam Mark Component
  - Max mark of 100
- Based solely on the final exam
- Class Mark Component
  - Max mark of 100
- 100% Assignments

* Course outline is authoritative.

27

## Assessment

- The final assessment is a weighted geometric mean of 60% exam (*E*) and 40% class (*C*) component.

$$M = e^{\frac{60 \ln E + 40 \ln C}{100}}$$

- Additionally, minimum of 40 required in exam (*E*) and class (*C*) components to pass.

28

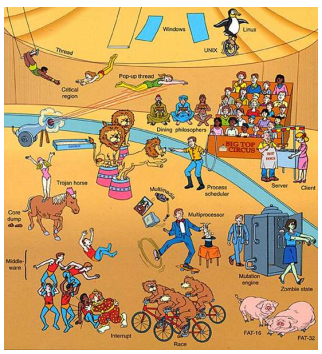
## Assessment

- You need to perform reasonably consistently in both exam and class components.
- Geometric mean only has significant effect with significant variation.
- Reserve the right to moderate marks, and moderate courses individually if required.
  - Warning: We have not moderated marks in the past.

29

## Textbook

- Andrew Tanenbaum, *Modern Operating Systems,* 3rd/4th Edition, Prentice Hall




30

## References

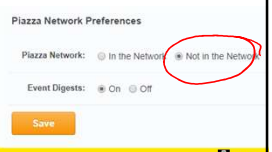
- A. Silberschatz and P.B. Galvin, *Operating System Concepts*, 5th, 6th, or 7th edition, Addison Wesley
- William Stallings, *Operating Systems: Internals and Design Principles,* 4th or 5th edition, Prentice Hall.
- A. Tannenbaum, A. Woodhull, *Operating Systems--Design and Implementation*, 2nd edition Prentice Hall
- John O'Gorman, *Operating Systems*, MacMillan, 2000
- Uresh Vahalla, *UNIX Internals: The New Frontiers*, Prentice Hall, 1996
- McKusick et al., *The Design and Implementation of the 4.4 BSD Operating System*, Addison Wesley, 1996



31

## Piazza Forums

- Forum for Q/A about assignments and course
  - Ask questions there for the benefit of everybody
  - Share your knowledge for the benefit of your peers
  - Look there before asking
- https://piazza.com/
  - Longer link on class web page
    - You will have received an invite from them to your UNSW email address.
      - z8888888@unsw.edu.au
  - Please join and contribute.
    - You don't have to join the "Piazza Network"
    - You opt-in or opt-out in Account Settings

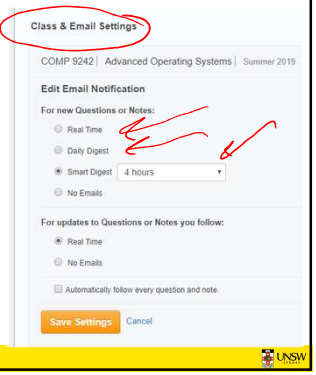





32

## You can control volume of Piazza mail

- Select your preferred latency of notification






33

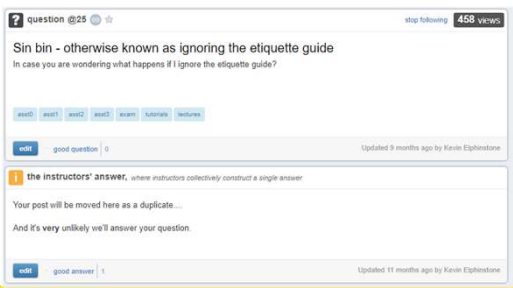
## Piazza Etiquette




34

## Enforcing standards

- Don't be offended if we move your post to the "sin bin"
  - Simply post again following the guide






35

## Consultations/Questions

- Questions should be directed to the forum.
- Admin and Personal queries can be directed to the class account cs3231@cse.unsw.edu.au
  - Don't PM me in Piazza
- We reserve the right to ignore email sent directly to us (including tutors) if it should have been directed to the forum.
- Consultation Times
  - See course web site.
  - Must email (cs3231@cse) at least an hour in advance and show up on time.
    - If we get at least one email, we'll run the consult.



36

## What next?

https://wiki.cse.unsw.edu.au/cs3231cgi/Checklist


37