

Welcome to OS @ UNSW

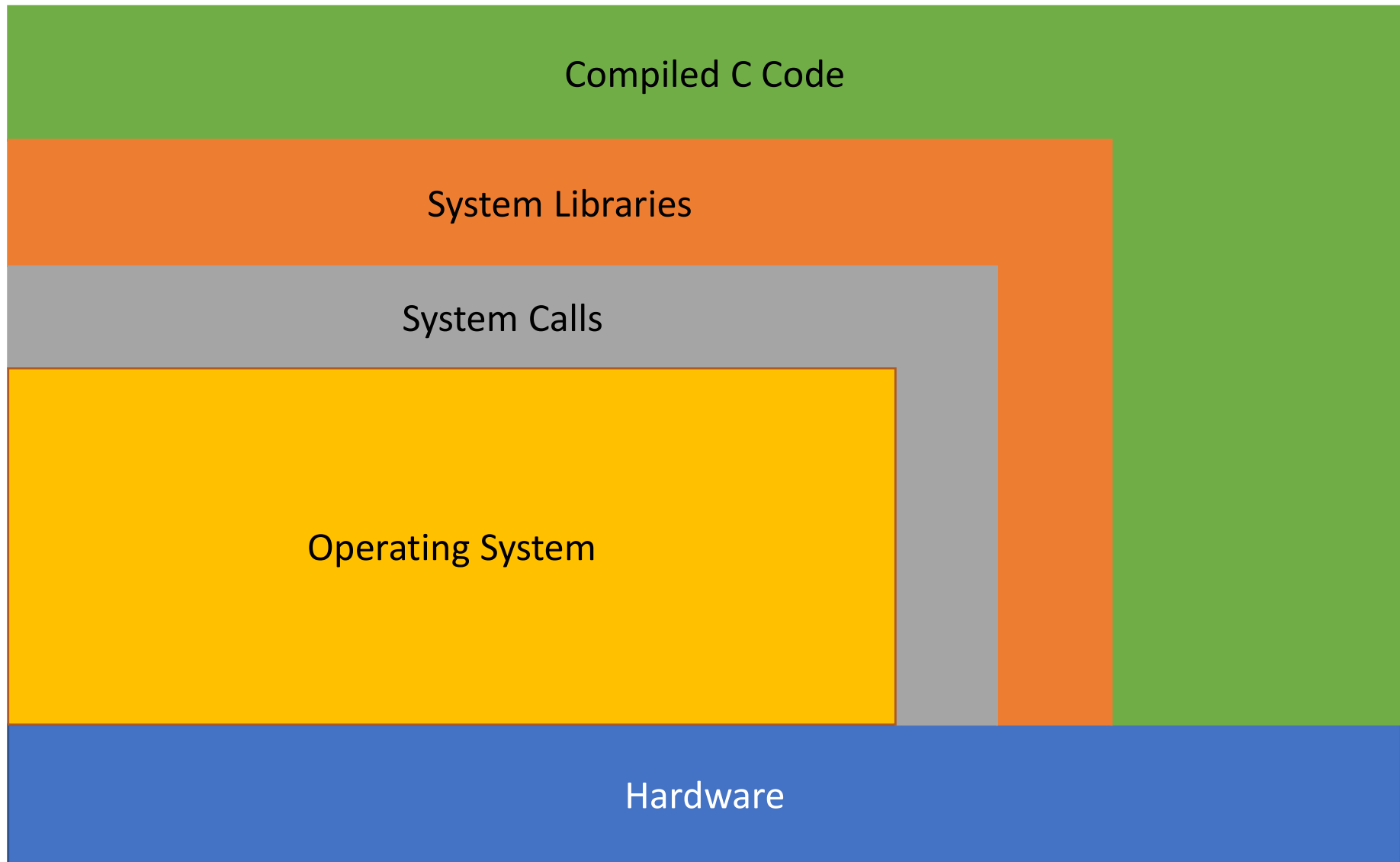
COMP3231/9201/3891/9283
(Extended) Operating Systems
Dr. Kevin Elphinstone

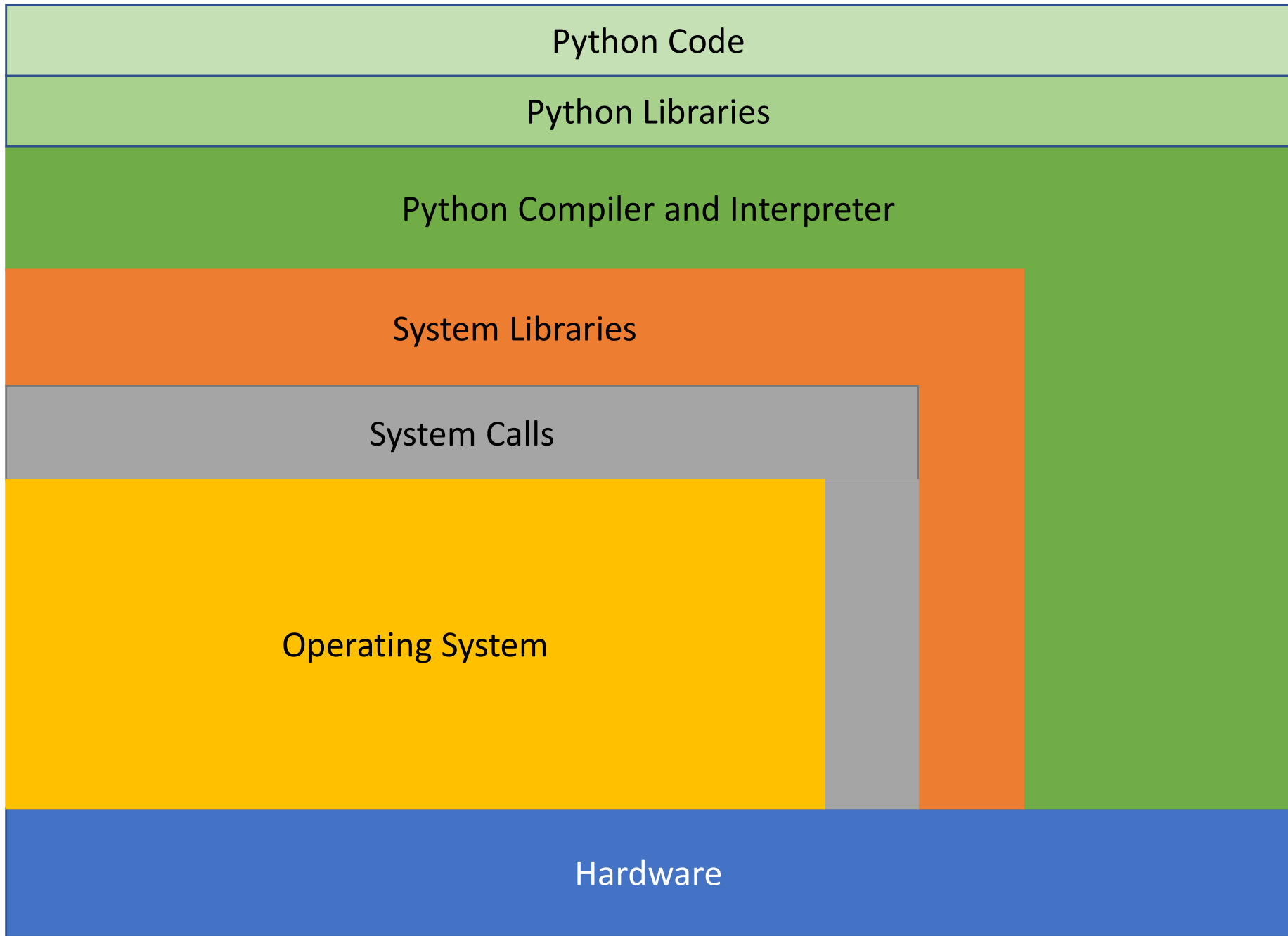


Questions

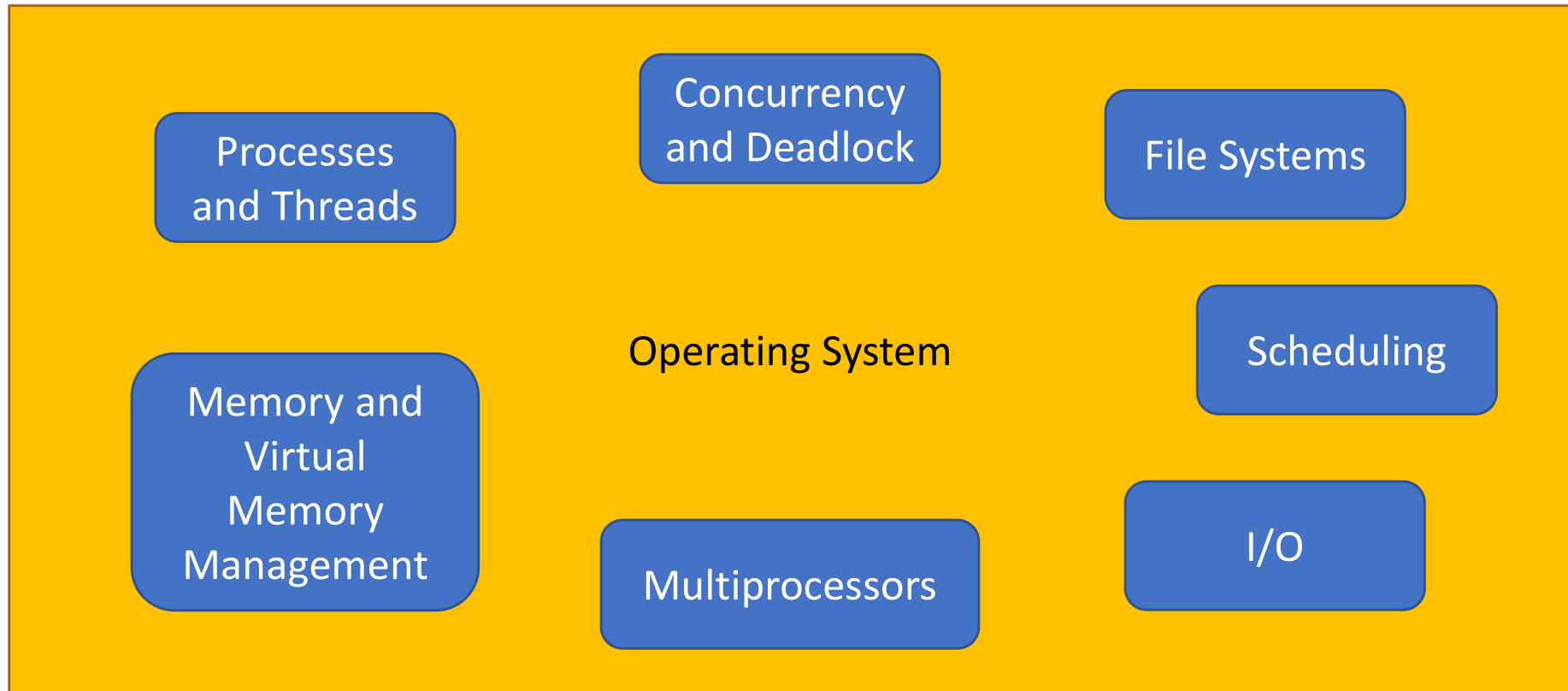
- Ask any questions you have in the course forum before the first lecture.
- I'll answer either on the forum or in the first lecture.

System Software Structure



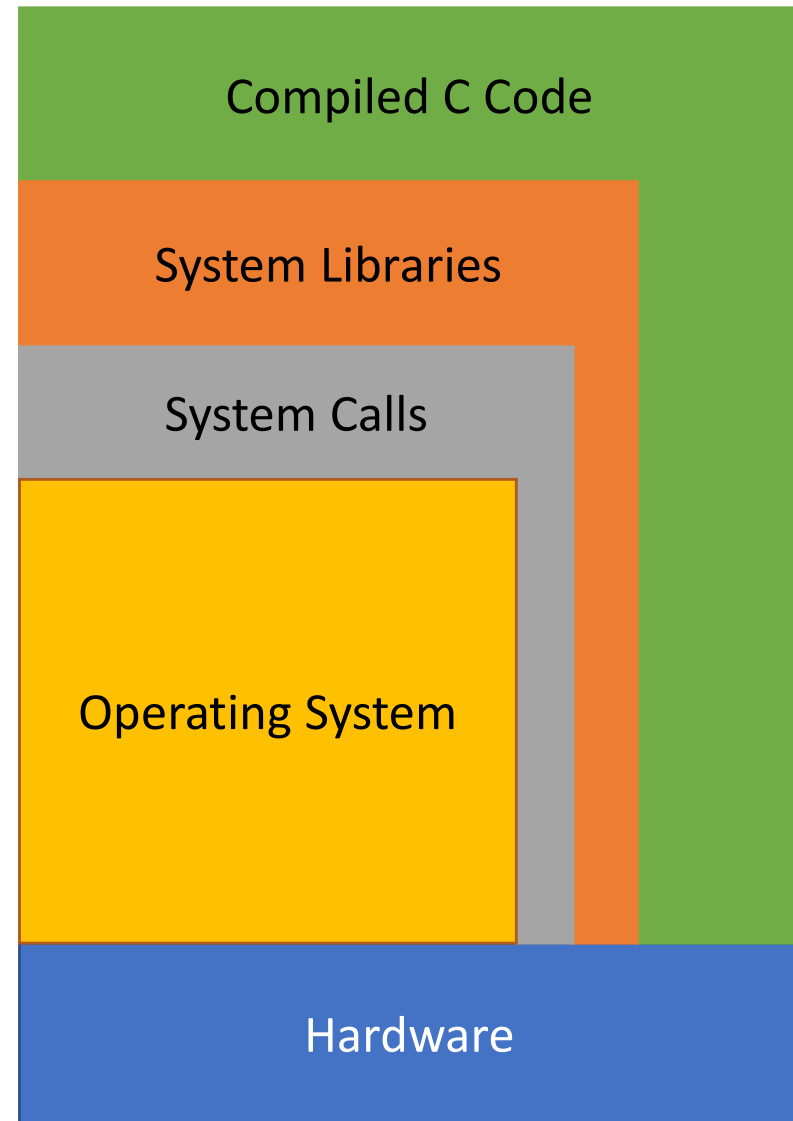


Major OS Topics



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.



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



Intended schedule*

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

* Subject to change



Overview of Course Outline

Prerequisites

- Data structures and algorithms
 - COMP2521, COMP9024 or COMP1927
 - Stacks, queues, hash tables, lists, trees, heaps,....
- Computer systems
 - COMP1521, DPST1092, COMP2121, COMP9032 or ELEC2142
 - Computer systems architecture
 - Assembly programming
 - Mapping of high-level procedural language to assembly language
 - Interrupts

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.

Why does this fail?

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

Operating System Coding



Why does this fail?

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

Lectures

- Common for all courses (3231/3891/9201/9283)
- 2 * 2 hrs each week
- 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.
- Lectures will be a mix of live streaming and pre-recorded
 - Will announce in advance
 - Video will be available afterwards in both cases

Administration

- Course Outline
- UNSW Timetable
- Consultations
- Group Nomination
- Survey Results!!

Work

- Lectures
- Tutorials
- Extended Lectures

Support

- Ed Forums
- Wiki

Assignments

- Submission Guide
- Assignment 0 Warm-up
- Assignment 1
- Assignment 2
- Assignment 3

Resources

OS/161

- General
- Man Pages
- Sys161 Pages

C coding































- Info Sheet

Debugging

- Learn Debugging

Lectures

The lecturer reserves the right to make changes to this schedule, so check it occasionally to see if there have been changes. The most likely changes are extra details on lecture content and references to the text. Click on the topic name to get one slide per page, and the print version to get 6 slides per page.

| Week | Topic | Book Ref | Print Format | Video |
|------|------------------------------------------------------------------|---------------|---------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| 1 | Course Introduction | |  |  |
| | Operating Systems Overview | 1 |  |  |
| | Processes And Threads | 2-2.2 |  |  |
| | Concurrency and Synchronisation | 2.3-2.3.7,2.5 |  |  |
| 2 | Deadlock | 6 - 6.7 |  |  |
| | Process and Thread Implementation | 2.2 - 2.2.5 |  |  |
| 3 | System Calls and R3000 Overview | 1.6 |  |  |
| | Computer Hardware, Memory Hierarchy, and Caching | 1.3 |  |  |
| 4 | File Management | 4 |  |  |
| | File Management Part 2 | 4 |  |  |
| | File Management (continued) | |  |  |
| 5 | Case Study: Ext2 | |  |  |
| | Case study: Ext3 | |  |  |
| | Memory Management | 3 |  |  |
| 6 | ASST2 Overview Video | |  |  |
| | Flexibility Week | | | |

Extended OS Comp3891/9283

Starts in week 1

- A combination of:
 - Examination of topics in more depth
 - Looking at research in areas (past/present)
 - OS/161 internals in more depth
- Separate Assessment
 - 80%-ish of final exam common with base course
 - 20%-ish targeted to extended students
 - ~~• Advanced assignment components part of the assessment~~
- Assumes the tutorials are not challenging enough
 - Effectively replaces the tutorial with extra interactive lecture.

Tutorials

- **Start in week 2**
- A mix of online and f2f
 - Depends on tutorial you enrolled in
- 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
 - There is intentionally more questions than can be covered
 - Review the questions beforehand

Assignments

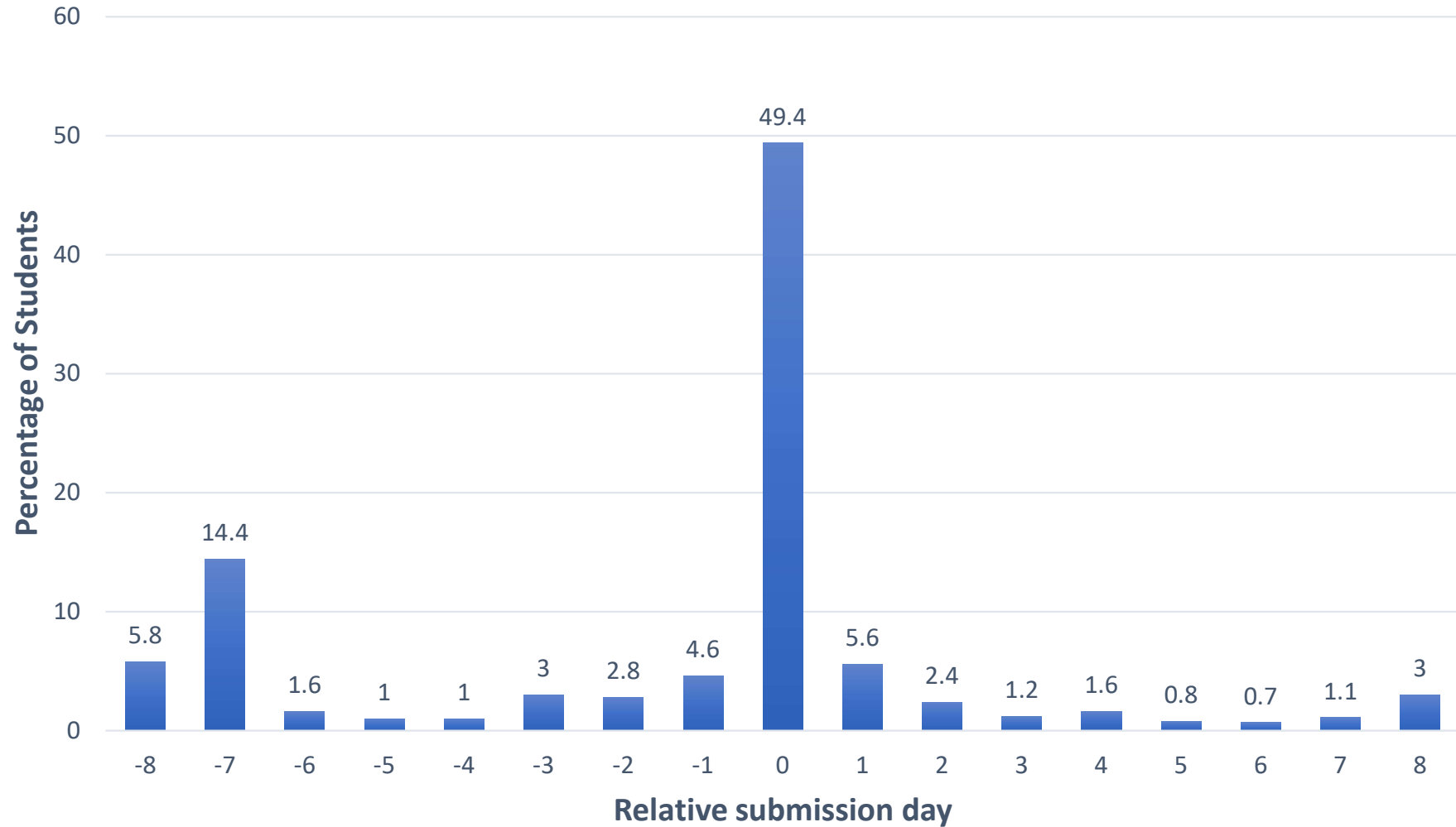
- Assignments form a substantial component of your assessment.
- They are challenging!!!!
 - Because operating systems are challenging
- We will be using OS/161,
 - an educational operating system
 - developed by the [Systems Group At Harvard](#)
 - With local changes.
 - It contains roughly 20,000 lines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

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!!!!

Assignment Submission Times 16% late

Historical Assignment Submission Statistics



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

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 |

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.

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

Assignments

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

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 lesser contributor down.
- Submissions are required to have significant contributions attributable to individual group members.
 - E.g. verifiable using the git revision control system

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

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

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.

Assessment

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

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

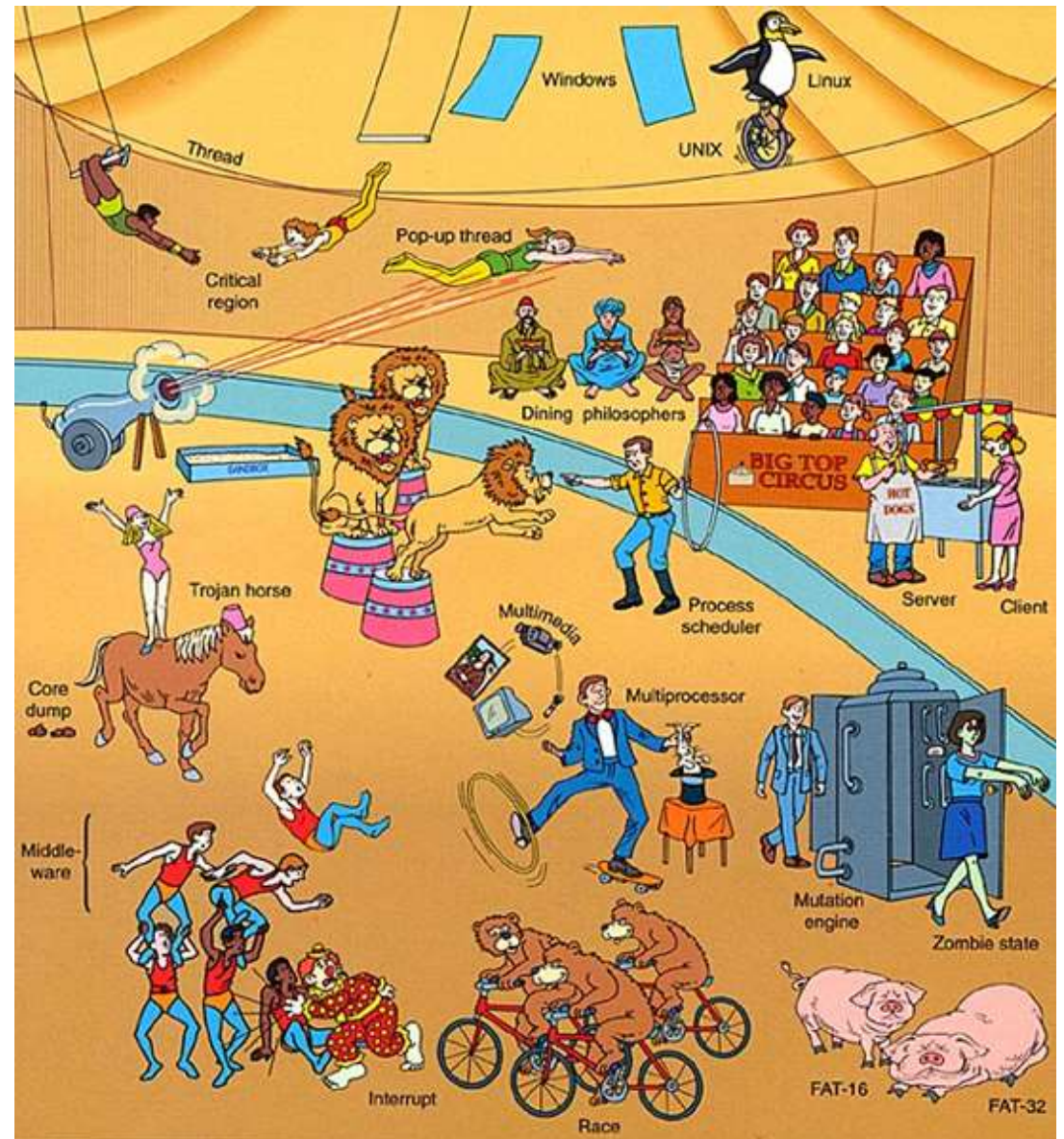
- Additionally, minimum of 40 required in exam (E) and class (C) components to pass.

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 moderated marks only once in the past

Textbook

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



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

Ed Forums

- Where announcements are posted!!
- 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://edstem.org/>
 - Longer link on class web page
 - You will have received an invite from them to your UNSW email address.
 - z8888888@unsw.edu.au
 - You need to join to follow the course.

Piazza Etiquette

Search first!

- You are probably not the first to experience the problem, so see if the question is answered before asking again.

Add to an existing post if directly related

- If you are experiencing a variant of the same issue, add to an existing post.

Start a new post for a separate issue

- Try to have an accurate title
- Avoid adding an unrelated question to a hot topic because you just happen to be there when you had the thought. It makes it hard to find for others.

Avoid bitmaps (screenshots)

- Bitmaps are not searchable so you limit the chances of fellow students finding your post, and indirectly make us less enthusiastic about providing a detailed answer to your non-searchable post.

Provide some context

- Cut-n-paste the error if appropriate, and include the preceding output to provide a chance for others to understand what is going on. Mention the OS/machine/environment you're using if it's not clear from the cut-n-paste.

Mark questions resolved if they are!

- Don't leave follow-ups unresolved if you have fixed your issue.

Leave questions unresolved if they are!

- I filter using 'unresolved' to find outstanding issues, I won't find them unless they are marked unresolved.

You're very welcome to post if you know the answer to an issue.

- The course staff do not have a monopoly on answers, nor do we monitor the forum 24hrs a day. A quick answer can make somebody's day (or at least avoid wasting it). A responsive forum can be an awesome resource for the entire course.

Enforcing standards

- Don't be offended if we reject your post
 - Simply post again following the guidelines

A good example

Hi, been trying to diagnose this for a while. Basically our program fails with:

```
panic: Fatal exception 2 (TLB miss on load) in kernel mode
panic: EPC 0x80020984, exception vaddr 0x0
```

Using GDB I backtraced to this call for copyout

```
#3  0x80020984 in copyout (src=0x0,
    userdest=0x0, len=0)
```

A bad example

Unable to access the full range of the page table, when initialising all values of the page table to NULL, I am unable to access the whole page table.

Here's how i accessed it:

```
void init_pt(paddr_t ***pt) {
    for (paddr_t i = 0; i <= 255; i++) {
        for (paddr_t j = 0; j <= 63; j++) {
            kprintf("Started init_pt[%d][%d]\n", i, j);
            pt[i][j] = NULL;
            kprintf("Finished init_pt[%d][%d]\n", i, j);
        }
        pt[i] = NULL;
    }
}
```

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 post private threads in Ed
- 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.

What next?

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



KevinElphinstone | [Settings](#) | [Logout](#)

Checklist

Checklist

FrontPage RecentChanges FindPage HelpContents **Checklist**

[Edit \(Text\)](#) [Edit \(GUI\)](#) [Info](#) [Unsubscribe](#) [Add Link](#) [Attachments](#) [More Actions:](#) ▼

Startup Checklist

- Watch the online intro lecture.
 - Bring any questions to the first lecture.
- Join Piazza (you should have received an invite sent to zID@unsw.edu.au)
- Review assignment 0
- Choose where you plan to do your assignment work (desktop, laptop, and at CSE).
 - Make sure the toolchain works on where you plan to work (see [Setup Overview](#))
- Set up git (see [Setup Overview](#))
- Choose an editor capable of code browsing (see [Setup Overview](#)).
- Complete ASST0