**Computer Science 432: Operating Systems (Revised)**
*Duane A. Bailey*

| | |
|---|---|
| **Office:** | Physics 306. |
| **email:** | `bailey@williams.edu` |
| **Calendar:** | `https://tinyurl.com/cs432-calendar` |
| **Texts:** | *xv6: a simple, Unix-like teaching operating system*, an e-text by Cox, Kaashoek, and Morris.<br>*RISC-V: An Overview of the Instruction Set Architecture*, an e-text by Porter.<br>*The C Programming Language*, 2nd edition, by Kernighan and Ritchie. |
| **Website:** | `www.cs.williams.edu/~cs432` |
| **Lecture:** | Wachenheim 114, Monday and Wednesday, 8:30-9:45a.m. |
| **Lab:** | Ward Lab (TBL 301), Monday (1:10-2:25 or 2:35-3:50) |
| **Small Groups:** | Knuth Lab (TCL 312b), Wednesday or Thursday. |
| **Office Hours:** | Tuesdays, 1-3pm, Fridays 9-10:30am, or by appointment. |
| **Lab Code:** | 3-9-2-7-8-1 |
| **Zoom Links:** | See calendar. |

This course explores the design and implementation of computer operating systems. An operating system (O/S) is simply a (collection of) programs that manage the real or imagined resources available for use in a computer. Since our understanding of what a computer is changes constantly (your thermostat? your car? your house?) it is not surprising that our view of what constitutes an operating system evolves as well. Nonetheless, there are basic aspects of operating system design that have evolved over, say, the last six decades that are worth studying: kernel design, process scheduling, concurrency and synchronization, virtualization, memory management, I/O and file system integration, system security, and support for distributed computation. In this course, we will primarily investigate classic and modern approaches to design of unix-style operating systems.

This course will be based (roughly) on the text *xv6: a simple, Unix-like teach operating systems* by Cox, Kaashoek, and Morris. This text describes, in detail, the `xv6` operating system, developed by the authors and recently ported to the RISC-V open-source architecture from Berkeley. The approach of these efforts is not to expose students to a mature operating system, but to investigate the design and implementation choices that are typical. The labs in this course are motivated by the labs in their MIT course, 6.S081.

Due to the playful nature of our explorations, this course requires a good understanding of the C programming language. Over time, I expect this to develop into mastery. Kernighan and Ritchie's classic text, *The C Programming Language*, 2nd edition, is an ideal reference. Because the code we write in C translates into the instructions that are native to the hardware, we will also become intimately familiar with the RISC-V architecture. Toward that end, I recommend Harry Porter's *RISC-V: An Overview of the Instruction Set Architecture*.

Below is a schedule of topics we will investigate. It follows, to some extent, the outline of the MIT text, but we will occasionally consider other technical papers discussing issues relating to operating systems design.

**Tentative Schedule of Topics**

| **Week of** | **Monday** | **Wednesday** | **Friday** | **Lab** |
|---|---|---|---|---|
| February 4 | — | — | Intro. | C |
| February 7 | C | O/S Interface (Ch. 1) | | Pointers |
| February 14 | Xv6 | System Calls (Ch. 2) | | Xv6 |
| February 21 | Memory (Ch. 3) | Page Tables | | System Calls |
| February 28 | Traps (Ch. 4) | Page Faults | | Memory Management |
| March 7 | Interrupts (Ch. 5) | Timers | | Copy-on Write |
| March 14 | Locking (Ch. 6) | Locking | | (COW, continued) |
| Spring Break | | | | |
| April 4 | Locking (Ch. 6) | Locking | | — |
| April 11 | Scheduling (Ch. 7) | Scheduling | | Multithreading |
| April 18 | Scheduling | Files (Ch. 8) | | (Multithreading, continued) |
| April 29 | Files | Files | | File Systems |
| May 2 | TBA | TBA | OSCO I | (File Systems, continued) |
| May 9 | OSCO II | OSCO III | OSCO IV | |

Each week we will meet on Monday and Wednesday for lecture, and Monday afternoon for labs. There will also be a 1-hour "small group" meeting to discuss lab efforts. Assignments will be handed out, generally, on Mondays and due a week later. We will review submitted code in small group meetings.

**Small Group/Code Review Meetings (Knuth Lab)**

| **Small Group** | **Time (Knuth Lab)** | **Members** |
|---|---|---|
| A | Wed. 11-noon | Diego, Petros, Tai, Ye |
| B | Wed. 1:10-2:10 | De La, Owen, Roxanne, Whit |
| C | Wed. 2:35-3:35 | Alex, Derek, Emily, Paul |
| D | Thurs. 10:00-11:00 | Aidan, Jihong, Mel, Rachel |
| E | Thurs. 1:10-2:10 | Clara, Enoch, Nick, Sophie |
| F | Thurs. 2:35-3:35 | Atlas, Dylan, Emma, Garett |

The last four class meetings (including the last two Fridays) will be dedicated to the O/S Conference (OSCON), where each of you will give a 10 minute technical presentation on an OS-related topic.

**O/S Conference 2022**

| | | |
|---|---|---|
| OSCO I | Friday, May 6 | Whit, Diego, Dylan, Rachel, Clara, Garett |
| OSCO II | Monday, May 9 | Owen, Alex, Enoch, Aidan, Derek |
| OSCO III | Wednesday, May 11 | Petros, Atlas, Sophie, Mel, Jihong, Emily, Ye |
| OSCO IV | Friday, May 13 | Paul, De La, Nick, Tai, Roxanne, Emma |

We will be seeing a fair amount of technical material. A feature of a 400-level computer science course is that you will be expected to pick up the working details of the course on your own. Most of the work is to be completed by individuals. The Honor Code applies in a rather direct way: I expect that all your work be your own. Please feel free to discuss the technical material with your colleagues, but not matters of execution. I will be available, as always, to help with any concern.

⋆