# DAT320
# Operating Systems and Systems Programming

Course Introduction — Fall 2019

Lecturer: Hein Meling

# Today

- Class overview and administration
- What is an operating system?
- What is a process?

# Teaching Staff

- Instructor: Hein Meling <hein.meling@uis.no>
  - Office: E437 (by appointment only; send email)
- Teaching Assistants:
  - Eivind Mellemstrand Stavnes <em.stavnes@stud.uis.no>
  - Leon Nielsen <le.nielsen@stud.uis.no>

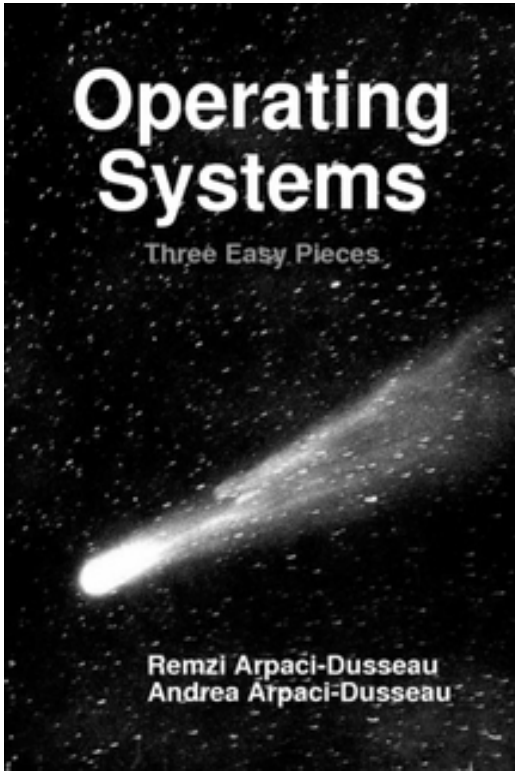

# Course Web Site

- https://github.com/dat320-2019/
  - Syllabus
  - Lecture Plan
  - Reading Material
  - Lab Projects
  - (but no lecture notes)

# Syllabus

- Chapters from the textbook
- Additional papers and blog posts
- Lab projects
- Slides published throughout the semester

# Textbook




- **Operating Systems: Three Easy Pieces**
  Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
  Arpaci-Dusseau Books
  August, 2018 (Version 1.00)

- http://pages.cs.wisc.edu/~remzi/OSTEP/

- Printed copies available in the campus book store and online (see link)

# Textbook Overview

| Intro | Virtualization | | Concurrency | Persistence | Appendices |
|---|---|---|---|---|---|
| Preface | 3 *Dialogue* | 12 *Dialogue* | 25 *Dialogue* | 35 *Dialogue* | *Dialogue* |
| TOC | 4 Processes | 13 Address Spaces code | 26 Concurrency and Threads code | 36 I/O Devices | Virtual Machines |
| 1 *Dialogue* | 5 Process API code | 14 Memory API | 27 Thread API code | 37 Hard Disk Drives | *Dialogue* |
| 2 Introduction code | 6 Direct Execution | 15 Address Translation | 28 Locks code | 38 Redundant Disk Arrays (RAID) | Monitors |
| | 7 CPU Scheduling | 16 Segmentation | 29 Locked Data Structures | 39 Files and Directories | *Dialogue* |
| | 8 Multi-level Feedback | 17 Free Space Management | 30 Condition Variables code | 40 File System Implementation | Lab Tutorial |
| | 9 Lottery Scheduling code | 18 Introduction to Paging | 31 Semaphores code | 41 Fast File System (FFS) | Systems Labs |
| | 10 Multi-CPU Scheduling | 19 Translation Lookaside Buffers | 32 Concurrency Bugs | 42 FSCK and Journaling | xv6 Labs |
| | 11 *Summary* | 20 Advanced Page Tables | 33 Event-based Concurrency | 43 Log-structured File System (LFS) | |
| | | 21 Swapping: Mechanisms | 34 *Summary* | 44 Flash-based SSDs | |
| | | 22 Swapping: Policies | | 45 Data Integrity and Protection | |
| | | 23 Complete VM Systems | | 46 *Summary* | |
| | | 24 *Summary* | | 47 *Dialogue* | |
| | | | | 48 Distributed Systems | |
| | | | | 49 Network File System (NFS) | |
| | | | | 50 Andrew File System (AFS) | |
| | | | | 51 *Summary* | |

# Lectures and Labs

- Combination of slides, live demo/coding, videos and whiteboard (mostly)
- Same room for the labs: bring your laptop
- There will be 6 mandatory lab assignments

| Activity | Day | Time | Room | |
|---|---|---|---|---|
| **Lecture** | Monday | 10:15 - 14:00 | E456 | |
| **Lab** | Tuesday | 08:15 - 10:00 | E456 | Assistance |
| **Lab** | Tuesday | 10:15 - 12:00 | E456 | Work on your own |
| **Lecture** | Wednesday | 10:15 - 12:00 | E456 | |

# (tentative) Lecture Plan Fall 2019

| W | M | D | Ch | Monday | Tuesday | Ch | Wednesday | Deadlines | Comment |
|---|---|---|---|---|---|---|---|---|---|
| **34** | **Aug** | **19** | 2 | Introduction to Operating Systems | C Introduction | 5 | Process API | | |
| | | | 4 | Abstraction: The Process | | 6 | Mechanism: Limited Direct Execution | | |
| **35** | **Aug** | **26** | 7 | Scheduling: Introduction | | 9 | Scheduling: Proportional Share | | |
| | | | 8 | Scheduling: Multi-Level Feedback Queue | | 10 | Multiprocessor Scheduling | | |
| **36** | **Sep** | **2** | 13 | Abstraction: Address Spaces | Introduction to Go | 15 | Mechanism: Address Translation | **Lab 1** | |
| | | | 14 | Memory API | | 16 | Segmentation | | |
| **37** | **Sep** | **9** | 17 | Free-Space Management | Organizing Go code | 19 | Paging: Faster Translation (TLBs) | **Lab 2** | |
| | | | 18 | Paging: Introduction | | 20 | Paging: Smaller Tables | | |
| **38** | **Sep** | **16** | 21 | Beyond Physical Memory: Mechanisms | | | *No Lecture* | | |
| | | | 22 | Beyond Physical Memory: Policies | | | | | |
| | | | 23 | Complete Virtual Memory Systems | | | | | |
| **39** | **Sep** | **23** | 26 | Concurrency: Introduction | Concurrency in Go | 28 | Locks | **Lab 3** | |
| | | | 27 | Thread API | | | | | |
| **40** | **Sep** | **30** | 29 | Lock-based Concurrent Data Structures | | 31 | Semaphores | | |
| | | | 30 | Condition Variables | | 32 | Common Concurrency Problems | | |
| **41** | **Oct** | **7** | 33 | Event-based Concurrency | Networking with Go | | | **Lab 4** | |
| | | | | | | | | | |
| **42** | **Oct** | **14** | | | | | | | |
| **43** | **Oct** | **21** | | *No Lecture* | | | *No Lecture* | **Lab 5** | AFT |
| **44** | **Oct** | **28** | | *No Lecture* | | | *No Lecture* | **Lab 6** | SOSP |
| **45** | **Nov** | **4** | | | | | | | |
| **46** | **Nov** | **11** | | | | | | | |
| **47** | **Nov** | **18** | | | | | | **Lab 7** | |
| **48** | **Nov** | **25** | | | | | | | |
| **49** | **Dec** | **2** | | **Lab exam: ?** | | | | | |
| **50** | **Dec** | **9** | | **Written exam: ?** | | | | | |

# Slip Days and Resubmissions

- Free slip days: 5
- For each extra slip day used, your grade is reduced by 5 points (0-100).
- **Resubmission (but only one pr. handin):**
  - 5 points for the first resubmission
  - 10 points for the second resubmission
  - 20 points for the third resubmission
  - A fourth resubmission will not be allowed, resulting in failing the lab.

# Unix Account Registration

- To complete the lab you'll need a Unix user account
- Go to http://user.ux.uis.no/ today!
- Register ASAP
- Physical access to the Linux lab (E353)
  - PIN code: 2244

# Autograder

- We will use Autograder to manage
  - Courses, Users and Groups on GitHub
  - Lab submissions, testing and scoring
- Not ready yet, but will be @ http://ag.itest.run/
  - When ready, sign up using your GitHub account
  - (you can create a new GitHub account for courses if you prefer)

# Group Registration

- The last lab will be carried out in groups
  - General rule: two students per group
  - Working alone or three member groups only accepted by application stating a reason
- Registration will be carried out on Autograder later

# (tentative) Lab Assignments

| Lab | Topic | Grading | Submission | Deadline |
|---|---|---|---|---|
| 1 | Unix Tools and C Programming | Pass / Fail ✓ | Individual | 3. Sep |
| 2 | Introduction to Go Programming | Pass / Fail ✓ | Individual | 10. Sep |
| 3 | Scheduling & Data Race Detection | Pass / Fail ✓ | Individual | 24. Sep |
| 4 | Base & Bounds Memory Management | Pass / Fail ✓ | Individual | 8. Oct |
| 5 | Paged Memory Management | Pass / Fail ✓ | Individual | 22. Oct |
| 6 | Network Programming in Go | Pass / Fail ✓ | Individual | 29. Oct |
| 7 | ChanStat: TV Channel Statistics | Graded | Group | **19. Nov** |

# Grading Policy

- Final Exam: 60 %
- Lab project: 40 %
  - **Lab exam** - explain your code
  - Graded based on final handin and examination
  - Must **pass all labs to attend exam**
  - More details will follow later

# Why Go?

- General-purpose programming language
  - Low-level, but garbage collected
- Fast, scalable and easy to learn
  - Designed for ease of reading code
- Builtin support for concurrency
- Easy to build command line tools, clients and servers (microservices)
  - Key enablers for cloud computing services

https://golang.org/

# Learning a new Programming Language

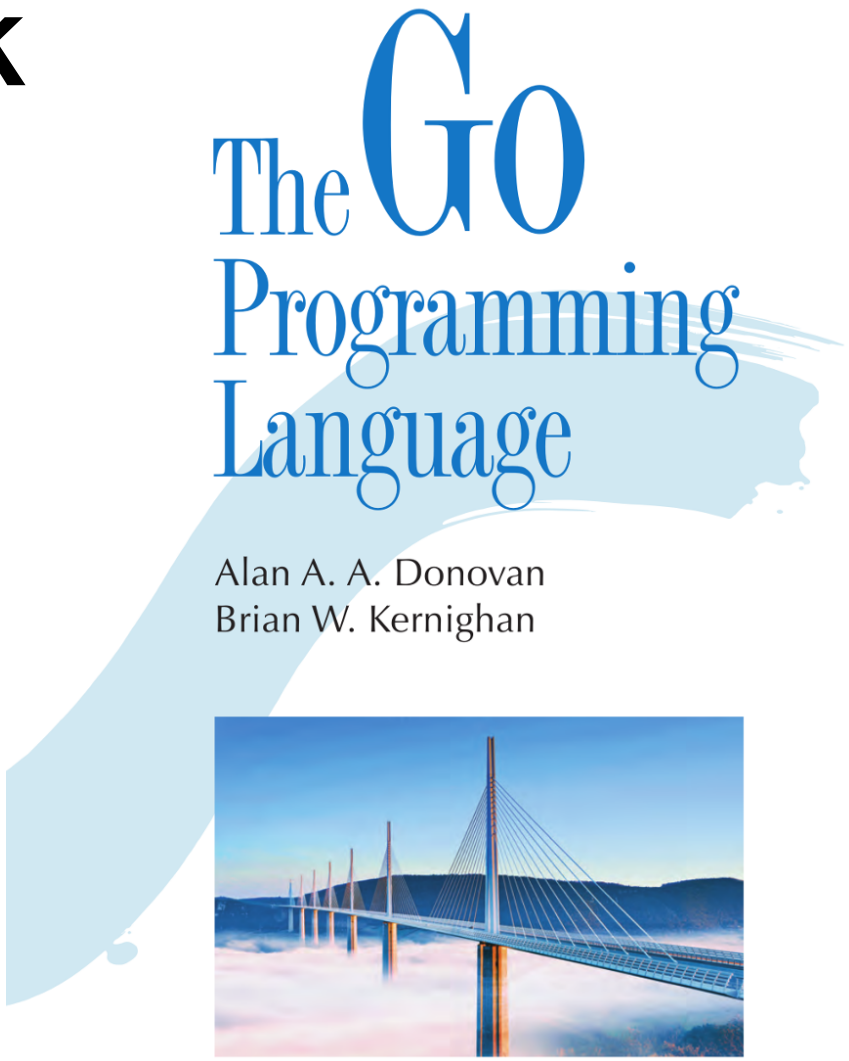
- Impossible to learn systems programming in theory
- You need to *get your hands dirty*!
- Start today!
- Best way to learn is to do real programming!
  - Compiler returns great error messages — great way to learn!

**https://golang.org/**

# Supplement Book

The Go Programming Language
Alan A. A. Donovan and Brian W. Kernighan
Addison-Wesley; ISBN: 978-0134190440
Published Oct 26, 2015

**http://gopl.io/**



# Why Study Operating Systems?