CS4410/11: Operating Systems

Rachit Agarwal
Anne Bracy

Instructors — Rachit Agarwal and Anne Bracy

- **Assistant Professor**, **Cornell (54th day in Ithaca)**
- **Previously:** Postdoc, UC Berkeley
- PhD, UIUC
- **Research interests**: Systems, networking, theory
- **Non-research interests**:
    - Flying planes (Still training),
    - Photography (Mostly landscape, recently portraits),
    - Traveling (31 countries and counting …),
    - Mixing cocktails

Instructors — Rachit Agarwal and Anne Bracy

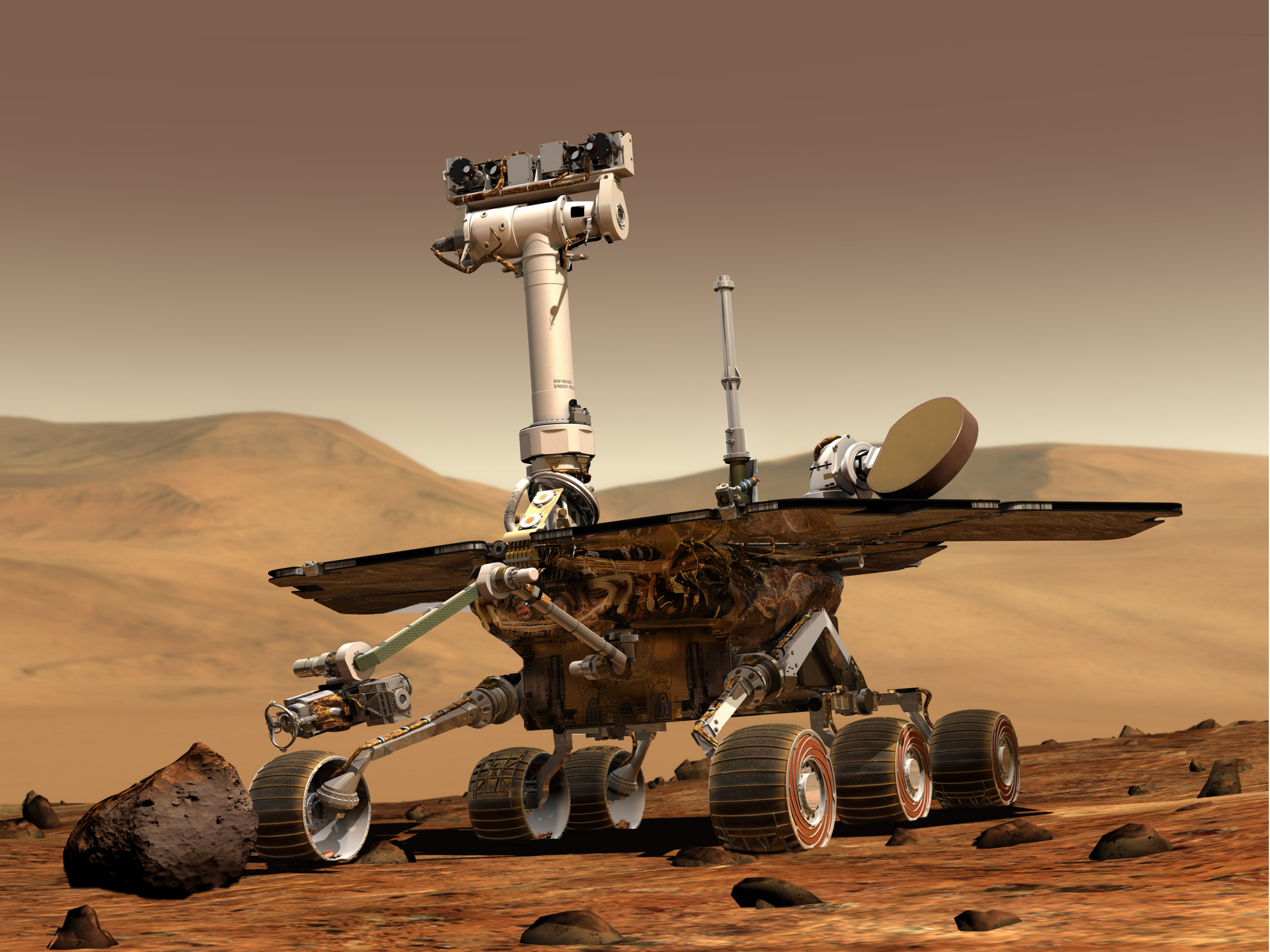
- **Senior Lecturer, Cornell (since Fall 2015)**
- **Previously:** Washington University in St. Louis, Intel Labs
- PhD, University of Pennsylvania
- **Professional interests:**
  - Teaching: Computer architecture, system software
  - Research: Microarchitecture, instruction fusion
- **Other interests**:
  - Travel
  - Speaking German
  - (legally) swimming in gorges

This course — Operating Systems

- **Learn about operating systems design and principles**

- **Today**:
  - What is an operating system?
  - Why study operating systems?
  - Course organization

- **Goal this semester**: Have fun (good grade will follow)!


569146-500
S/N0001 REV

Example 1 — Mars Rover

- **20Mhz processor, 128MB of DRAM, 256MB Flash**
- Cameras, Sensors, Batteries, Solar Panels, Antennas, ..
- Unpredictable environment (to say the least)
  - How to share resources while multi-tasking?
  - How to store files of types audio, images, logs, …?
  - How to send/receive data?
  - How to avoid/overcome failures?
- **An operating system designed in a principled manner**

Example 2 — Self-driving cars

- **150 MacBook Pros in one car**
- Cameras, Sensors, GPS, Image recognition, ..
- Unpredictable environment (to say the least)
    - How to share resources while multi-tasking?
    - How to store files of types audio, images, logs, …?
    - How to send/receive data?
    - How to avoid/overcome failures?
- **An operating system designed in a principled manner**

Example 3 — Smart phones (iPhone)

- **A8 chipset, 16GB DRAM, …**
- Camera, Sensors, Fingerprint device, Image recognition, ..
- Evolving ecosystem of **heterogeneous** applications
    - How to share resources while multi-tasking?
    - How to store files of types audio, images, videos, …?
    - How to send/receive data?
    - How to run new applications w/o reprogramming?
    - How to secure data (e.g., Apple pay)?
- **An operating system designed in a principled manner**

Example 4 — Web services (Google, Facebook, ..)

- **Hundreds of Thousands of servers, Billions of users**
- Search, Maps, Messaging, Images, Videos, …
- **Heterogeneous** applications and **heterogeneous** users
  - How to share resources across applications and users?
  - How to store files of types audio, images, videos, …?
  - How to send/receive data between servers?
  - How to run new applications w/o reprogramming?
  - How to secure data (e.g., privacy settings in Facebook)?
- **An operating system designed in a principled manner**

What is an operating system?

What is an operating system?

Software to manage hardware resources

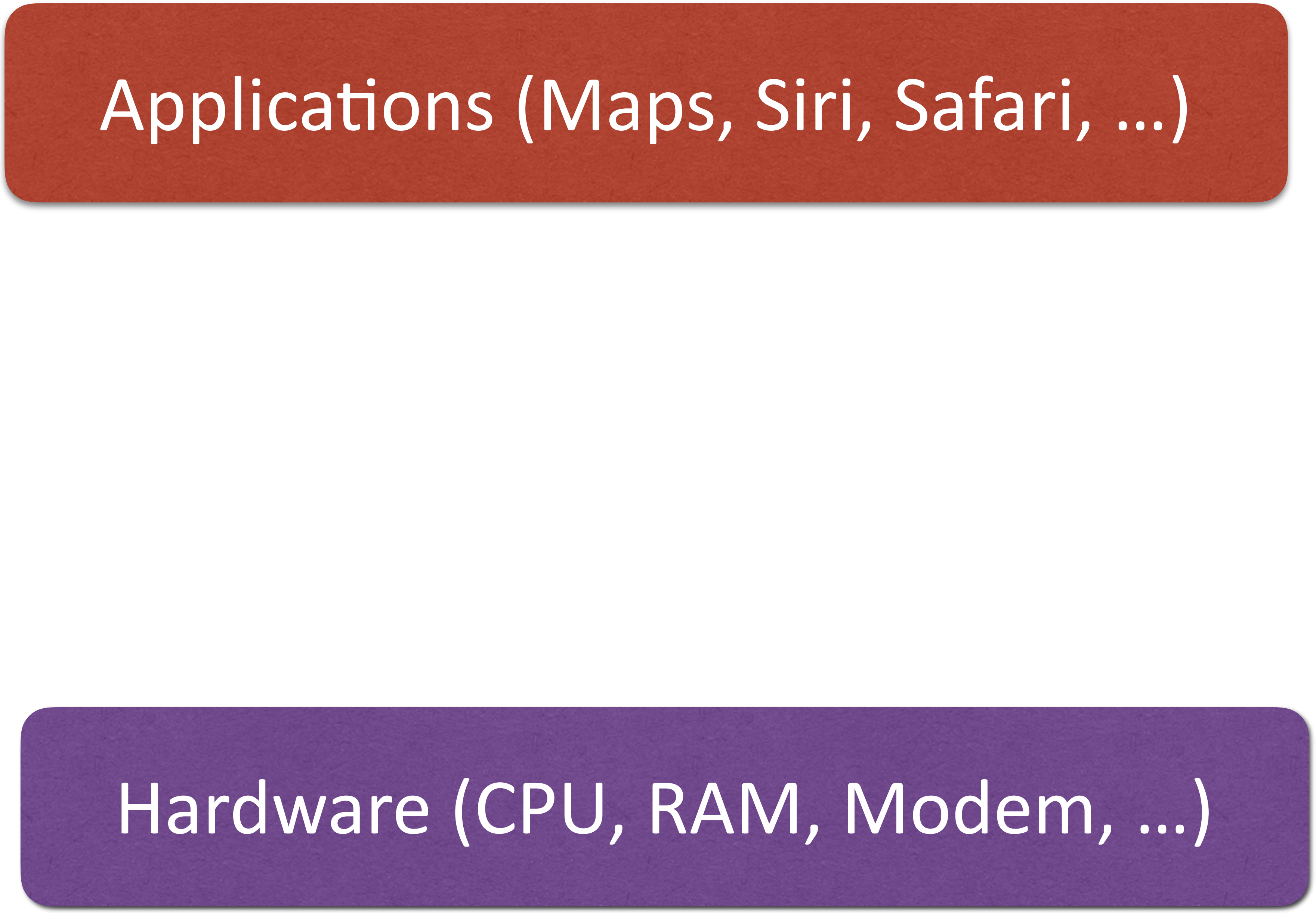
What is an operating system?

Software to manage hardware resources

Hardware (CPU, RAM, Modem, …)

What is an operating system?

Software to manage hardware resources

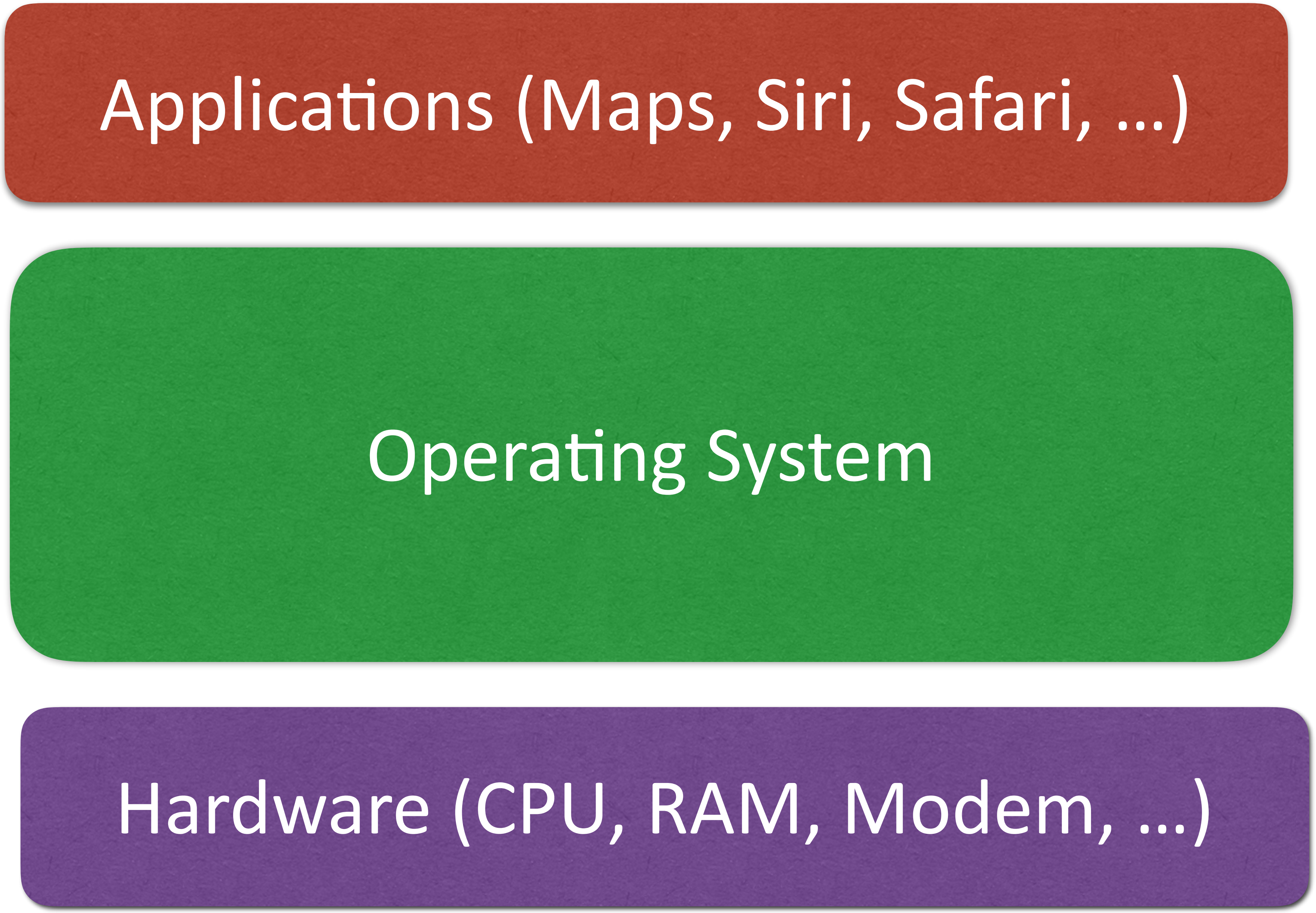


Applications (Maps, Siri, Safari, …)

Hardware (CPU, RAM, Modem, …)

What is an operating system?

Software to manage hardware resources



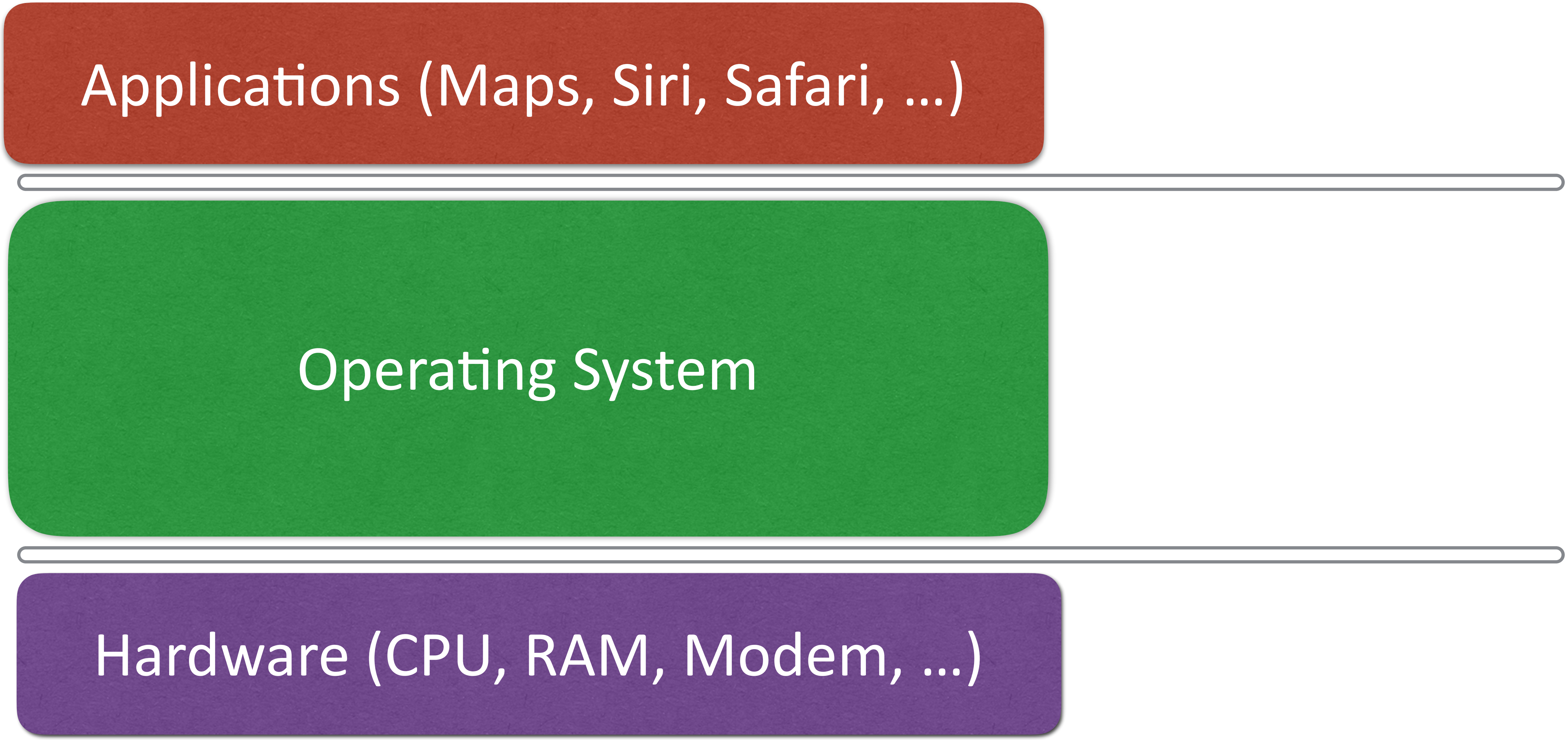
Applications (Maps, Siri, Safari, ...)

Operating System

Hardware (CPU, RAM, Modem, ...)

What is an operating system?

Software to manage hardware resources



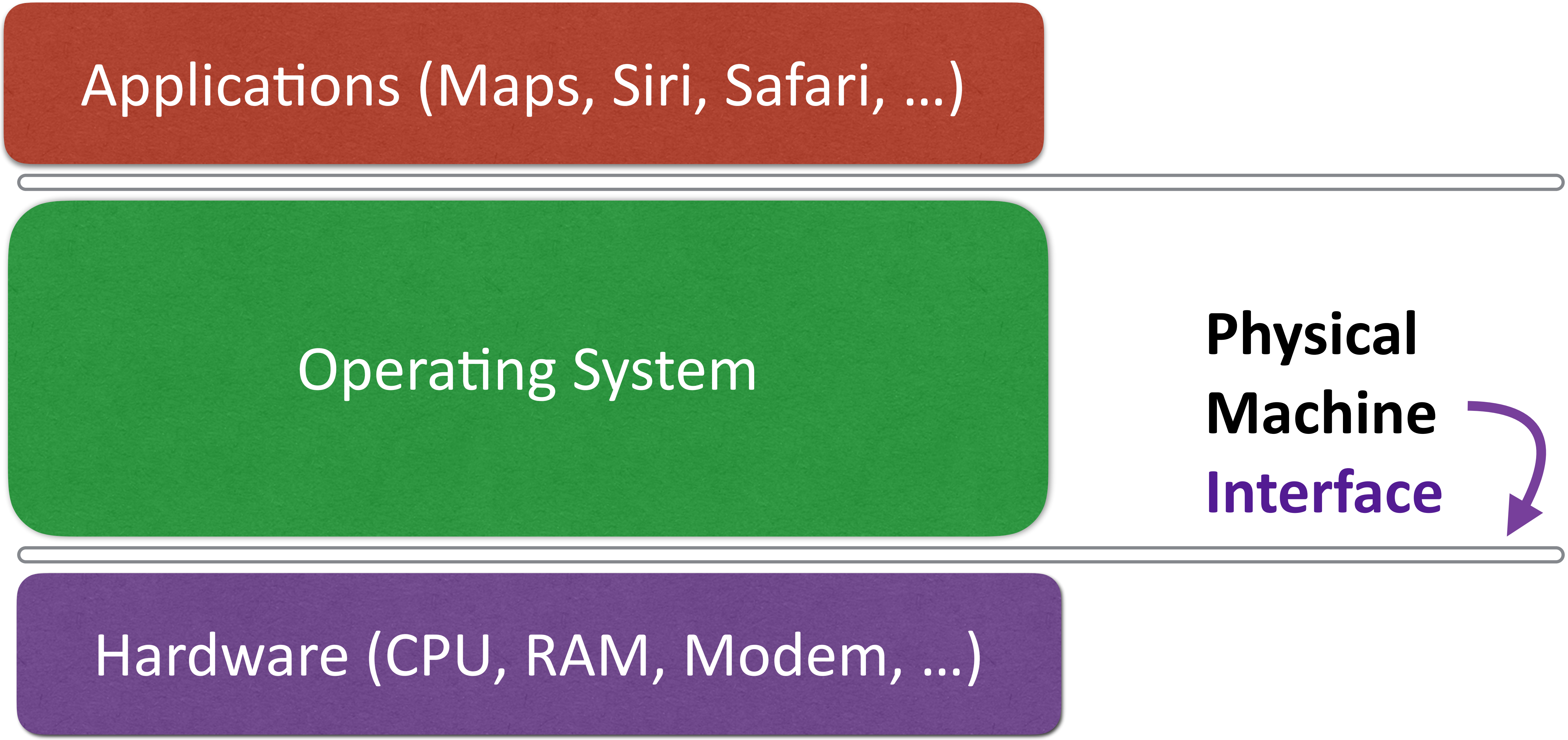
Applications (Maps, Siri, Safari, …)

Operating System

Hardware (CPU, RAM, Modem, …)

What is an operating system?

Software to manage hardware resources



Applications (Maps, Siri, Safari, …)

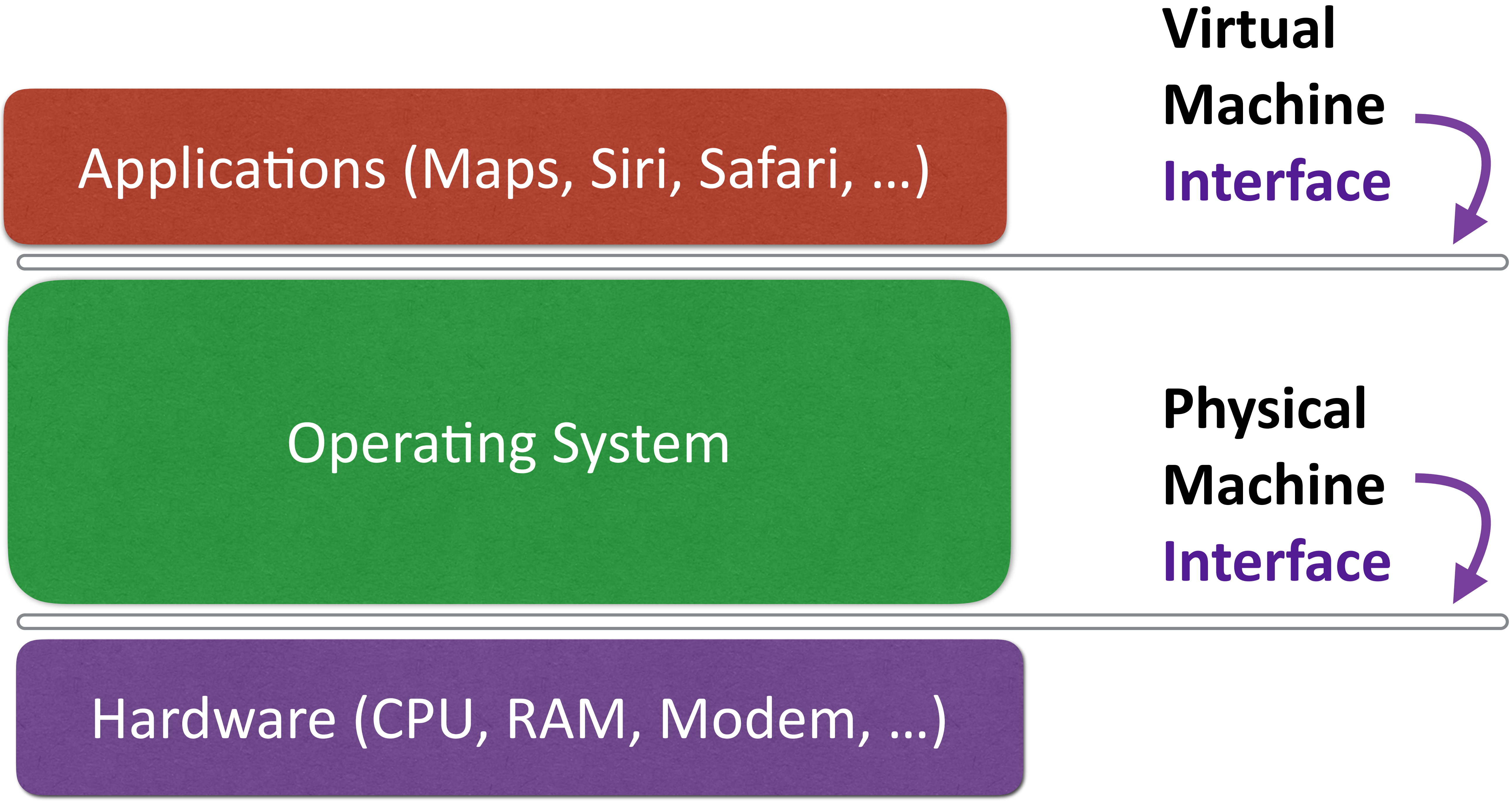
Operating System

Hardware (CPU, RAM, Modem, …)

Physical
Machine
Interface

What is an operating system?

Software to manage hardware resources



Virtual Machine Interface

Applications (Maps, Siri, Safari, …)

Operating System

Physical Machine Interface

Hardware (CPU, RAM, Modem, …)

Virtual machine

Software emulation of an “abstract machine”

- Illusion of hardware having features one wants
  - E.g., networking (files vs. packets)
  - E.g., storage (files vs. registers)
- Simplicity of programming
  - Each application: “Yay! I have all the resources!”
  - Each application: “I don’t care if you have SSD or disk”
- More powerful than hardware interface
  - E.g., network failures masked

More discussion throughout the course

What is an operating system?

Software to manage hardware resources:

- Multi-tasking and concurrency (5 weeks)
    - Processes, Threads, Synchronization, Deadlocks
- Sharing resources among users and systems (2 weeks)
    - Scheduling and memory management
- Storage and fault-tolerance (2 weeks)
    - File Systems, RAID
- Networking (2.5 weeks)
    - Unreliable and reliable communication
- Security (1 week)

What makes an operating system good?

Two criteria:

- **Principles**
  - Does the design conform to a set of principles?
- **Performance**
  - Does the design meet certain objectives?

Operating Systems Design Principles

Discussed throughout the course. Center around:

- **Reliability**
  - Does the system operate as per its specification?
  - E.g., NASA does not want Mars Rover to convert into Wall-E
- **Availability**
  - What portion of the time is the system working?
  - E.g., A flash memory error led to 13 day problems in Rover
- **Security**
  - Can the system be compromised by an attacker?
  - E.g., Imagine if Martians take control of Rover (or have they?)

…

Operating Systems Design Principles [Cont.]

- **Privacy**
  - Is the data accessible only to authorized users?
  - E.g., NSA tracking people using phones
- **Portability**
  - Across hardware, applications, …
  - E.g., Re-write the entire iOS to use iPhone 7?
- **Fairness**
  - Do applications receive their fair share of resources?
  - E.g., Google Map users and Google search users

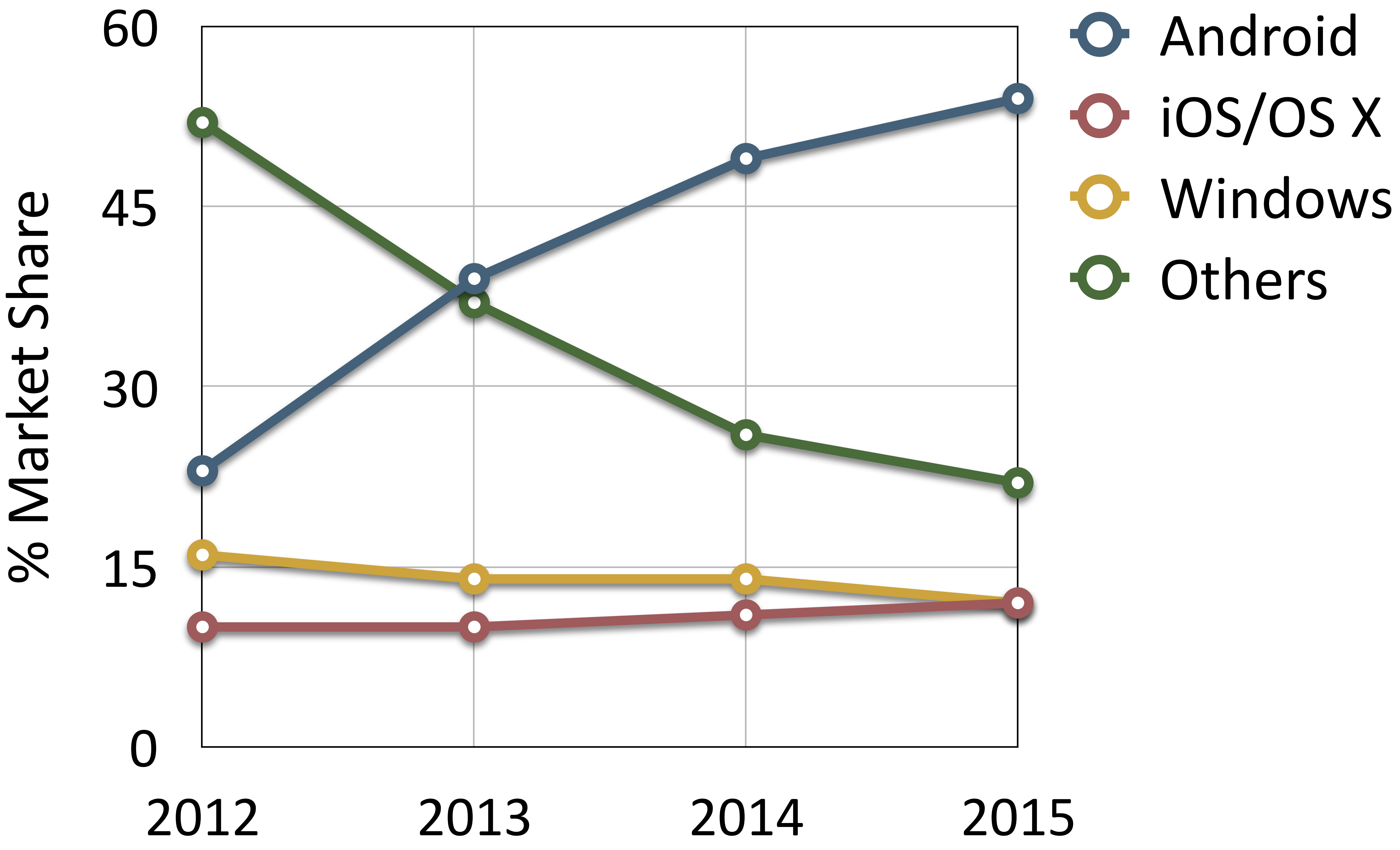
Operating Systems Performance

- **Latency**
  - How long does an operation take to complete?
- **Throughput**
  - #Operations per unit time
- **Utilization**
  - Fraction of resources used over time
- **Scalability**
  - How does the performance change with size?
- **Predictability**
  - Consistency (over time) for an objective

Why study operating systems?

- Laptop
- Cell phone
- Microwave
- Washer
- Dryer
- Dishwasher
- Coffee maker
- Refrigerator
- Television
- Game console

Why study operating systems? [Cont.]



Why study operating systems? [Cont.]

	Android	iOS/OS X	Windows	Others
Release	2008	2007/2001	1993	-

- OS have **existed since 1954**
- **Most widely used OS today designed in last decade**

This is the most exciting era for OS designers!

Why study operating systems? [Cont.]

- **Ever-evolving applications**
  - Self-driving cars
  - Internet of Things
  - Smart homes
- **Ever-evolving technologies**
  - new hardware (e.g., rack-scale computers)
  - new memory technologies (e.g., Intel 3D-X Point)
  - new networking technologies (e.g., RDMA)

Require new innovations in Operating Systems design (but principles remain mostly unchanged)

Why study operating systems? [Cont.]

Jonathan James

NASA, 1.7M$ software

crappy code

Adrian Limo

“Homeless Hacker”

MS, NYT, Yahoo!, BoA

please correct flaws

Kevin Mitnick

“Hacker Poster Boy”

16 years!

Woz

Free long-distance calls

Tsutomu

“Poster boy hacked me! Huh!”

This course — principles and performance

- Design principles (more or less) same across various OS
- Performance objectives same across various OS
- Many ideas applicable to other areas:
  - Big data analytics (scheduling, storage, ….)
  - Datacenters (concurrency, scheduling, storage, …)
  - Genomics (storage, security, …)

Focus on fundamentals
(implementation varies across OS)

This course — organization (lectures)

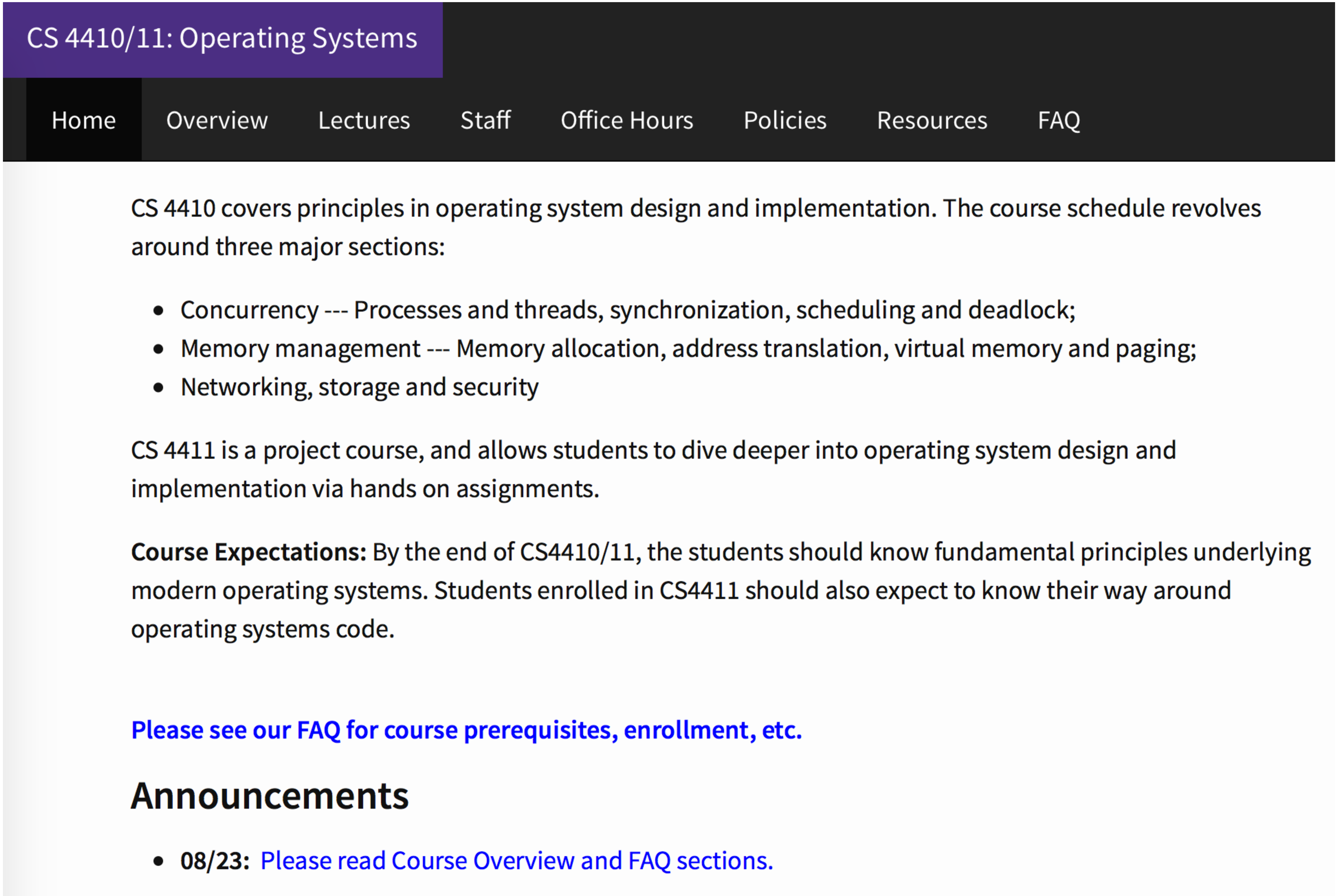
(carefully read the webpage)

- Two CS4410 lectures per week
    - If you are here, you know when and where

- One CS4411 lecture per week
    - Friday 2-3PM, B14 Hollister Hall

This course — organization (website)

http://www.cs.cornell.edu/Courses/cs4410



CS 4410 covers principles in operating system design and implementation. The course schedule revolves around three major sections:

- Concurrency --- Processes and threads, synchronization, scheduling and deadlock;
- Memory management --- Memory allocation, address translation, virtual memory and paging;
- Networking, storage and security

CS 4411 is a project course, and allows students to dive deeper into operating system design and implementation via hands on assignments.

Course Expectations: By the end of CS4410/11, the students should know fundamental principles underlying modern operating systems. Students enrolled in CS4411 should also expect to know their way around operating systems code.

Please see our FAQ for course prerequisites, enrollment, etc.

Announcements

- **08/23:** Please read Course Overview and FAQ sections.

This course — organization (office hours)

(carefully read the webpage)

- ~20 office hours per week (may increase/decrease)
  - Schedule on webpage, all in G13 Gates Hall

- **Instructor office hours (this week):**
  - **Rachit:** 411C Gates Hall @ 10AM, Thursday
  - **Anne:** 452 Gates Hall @10AM, Friday
  - Others, by appointment *only*
  - Only if TA cannot answer your questions
  - No “technical” questions over emails

This course — organization (Grades :-))

(for students registered in CS4410 *only*)

- One Final Exam: 30%
- 5 Projects: 40%
- ~10 homeworks: 30%

Yes, no prelims!
(Also, we will take best 6 out of 10 homework marks)

*Projects to be done individually
*Homeworks to be done in pairs

This course — organization (Grades :-))

(for students registered in CS4410 *and* CS4411)

- One Final Exam: 30%
- 2 Projects + 6 Projects: 40%
- ~10 homeworks: 30%

Yes, no prelims!
(Also, we will take best 6 out of 10 homework marks)

* First two project to be done individually
* Next six projects to be done in pairs
* Homeworks to be done in pairs

This course — organization (Grades :-))

(for students registered in CS4411 *only*)

- 6 Projects: 100%

* Projects to be done in pairs

This course — organization (problem solving sessions)

(Yay, or Nay)

- Problem solving sessions?

- Useful for you
  - TAs bring 1 question
  - You try for ~15 minutes
  - TA solve the problem on blackboard

- 1 extra hour per week?

Yay, or Nay?

This course — organization (zero tolerance)

Zero tolerance policy (read webpage very carefully)

- Cheating
  - All submitted work must be yours
  - Okay to collaborate, but not to share/copy solutions
  - Properly attribute any resource used
  - Piece of cake to detect cheating
  - If you think you may be cheating, you probably ARE!

(carefully read all course policies on the webpage)!

*If you have a concern, talk to us about policies by 1st Sep.

This course — organization (zero tolerance)

Zero tolerance policy (read webpage very carefully)

- Delays (homeworks)
  - No late submissions allowed on homeworks
  - “k out of n” policy to accommodate all kinds of issues
- Delays (projects)
  - You have 3 no-penalty late submission days
  - Across all projects! Use them carefully.
- Zero credit for delayed submissions

*If you have a concern, talk to us about policies by 1st Sep.

This course — organization (questions?)

Ask any question you may have

This course — CS4410/11

What is the best way to learn OS?

Get involved in research!

- I may be willing to advise a couple of students
- Come see me during my office hours!