# My first thoughts on going completely virtual

1. My planned course is not going to work.
2. Maybe there is an opportunity here.
3. Let's model a realistic, distributed SW development process.

---

## Today

- Logistics
- Brief introduction (my background)
- Your background and expectations
- What is Software Engineering
- Course overview

---

## Logistics: the CSE 403 team

**Instructor**
- René Just
  - rjust@cs.washington.edu

**Teaching assistants**
- Kaushal Mangipudi and Brendan Wallace
  - kaushalm@uw.edu, bwbw@uw.edu

Please send inquiries to the entire team.
Logistics: meetings

- **Lectures**: M/W/F 12:30pm – 1:20pm (Zoom)
- **Project meetings**: Thu 1:30pm – 2:20pm (Zoom by default)
- **Team meetings**: Tue 1:30pm – 2:20pm (Zoom by default)

No project and team meetings this week.

Logistics: resources

- **Course website**: [https://homes.cs.washington.edu/~rjust/courses/2020Spring/CSE403](https://homes.cs.washington.edu/~rjust/courses/2020Spring/CSE403)
- Submission of assignments via Canvas: [https://canvas.uw.edu](https://canvas.uw.edu)
- Discussions on Slack: [https://cse403group.slack.com](https://cse403group.slack.com)

My background

My research areas

- Software testing and verification
- Software debugging
- Software security
My background

My research areas
- Software testing and verification
- Software debugging
- Software security
- Empirical software engineering
- Data science / Applied ML

Your background and expectations

10 min introduction (breakout rooms)
- Background: What is your SE background?
- Goals: What are your career goals (grad school, SW engineer, etc)?
- Top-2 expectations: What do you expect from this course?

1. Introduce yourself to your group
2. Post a summary of your group discussion on Slack
(Bulleted lists for: backgrounds, goals, and expectations)

What is Software Engineering?
What is Software Engineering?

- Developing in an IDE and software ecosystem?
- Coding and debugging?
- Deploying and running a software system?
- Empirical evaluations?
What is Software Engineering?

- Developing in an IDE and software ecosystem?
- Coding and debugging?
- Deploying and running a software system?
- Empirical evaluations?

All of the above -- much more than just writing code!

What is Software Engineering?

More than just writing code
The complete process of specifying, designing, developing, analyzing, deploying, and maintaining a software system.

- Common Software Engineering tasks include:
  - Requirements engineering
  - Specification writing and documentation
  - Software architecture and design
  - Programming
  - Software testing and debugging
  - Maintenance and refactoring

The Role of Software Engineering in Practice

Focus of intro to programming courses. Our goal: go beyond the inner cycle.
The Role of Software Engineering in Research

Experimental infrastructure is software, too!

Example (automated debugging)
- 150 configurations, 1000+ benchmarks
- 1-85 hours per execution
- 200,000+ CPU hours (~23 CPU years)

Software bugs can lead to wrong scientific conclusions.

Why is Software Engineering important?

Software is eating the world!

- ~15 million lines of code
  - Let's say 50 lines per page (0.05 mm)
- 300,000 pages
- 15 m (49 ft)
## Summary: Software Engineering

**What is Software Engineering?**
- The complete process of specifying, designing, developing, analyzing, and maintaining a software system.

**Why is it important?**
- Decomposes a complex engineering problem.
- Organizes processes and effort.
- Improves software reliability.
- Improves developer productivity.

## Course overview: the big picture

- **Software processes, requirements, and specification**
  - Different software development processes.
  - Precise writing (requirements and specifications).
- **Software development**
  - Decompose a complex problem and build abstractions.
  - Improve your coding skills.
  - Effectively use version control (Git).
- **Software testing and debugging**
  - Write effective (unit) tests.
  - Hands-on experience, using testing and debugging techniques.
  - Continuous integration.
- **Class project**
  - Apply all of the above in a group project.

## Course overview: grading

**Overall grading**
- **60%** Semester-long class project
- **30%** In-class exercises and individual assignments
- **10%** Participation

**No final exam this quarter.**

## Expectations

- Programming experience and familiarity with one programming language (Java, C++, ...).
- Active participation in discussions.
- Teamwork and communication.
- Reflecting on and improving submitted assignments.

You must already know how to program.