PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

CSE341: Programming Languages

Lecture 0
Course Mechanics
ML Variable Bindings

Brett Wortzman

Spring 2020

Welcome!

We have 10 weeks to learn *the fundamental concepts* of programming languages

With hard work, patience, and an open mind, this course makes you a much better programmer

- Even in languages we won't use
- Learn the core ideas around which *every* language is built, despite countless surface-level differences and variations
- *Poor* course summary: "Learn ML, Racket, and Ruby"

Today's class:

- Course mechanics
- Remote instruction logistics
- *[A rain-check on motivation]*
- If time: Background and setup for ML

Concise to-do list

In the next 24-48 hours:

1. Read course web page:
   http://courses.cs.washington.edu/courses/cse341/20sp/
2. Read all course policies (syllabus and links within)
3. Verify Zoom setup
4. Complete Homework 0 (worth 0 points)
5. Get set up using Emacs [optional; recommended] and ML
   - Installation/configuration/use instructions on web page
   - Essential; non-intellectual
     - No reason to delay!

Who: Instructor

Brett Wortzman:

- Teaching faculty
- Usually teach 142; excited to work with majors!
- Wrote undergraduate thesis on similar topics

Who: TAs

Blake Berger
(he/him)

Max Packer
(he/him)

Omar Ibrahim
(he/him)

Daniel Snitkovskiy
(he/him)

Josie Lee
(she/her)

Kenny Wu
(she/her)

Who: Course Creator

Dan Grossman, CSE Faculty

Occasionally will be referred to as “Prof. DJG”

Has poured blood, sweat, and tears into this material.

We are incredibly lucky to be able to build off of his course!

Staying in touch

- Message Board (Ed)
  - For most class discussions
  - Staff will monitor regularly
  - Feel free to answer each other's questions!
- Course email list: **`cse341a_wi20@u.washington.edu`**
  - Students and staff already subscribed
  - Very low traffic; usually just unexpected / urgent situations
- Anonymous feedback link on webpage
  - For good and bad: If you don't tell us, we don't know

Lecture: Brett

- Slides, code, and reading notes / videos posted
  - May be revised after class
  - *Take notes*: materials may not describe everything
    - Slides are just *visual aids* for me to use

- Ask questions, focus on key ideas

- Engage actively
  - Arrive *punctually* (especially this quarter!) and well-rested
  - Reduce *distractions*
  - *Write* down ideas and code as we go
  - Communicate *proactively* if attendance difficult

Section: TAs

- Required: will usually cover new material
- *Not* recorded: be sure to attend
- Sometimes more language or environment details
- Sometimes main ideas needed for homework
- *Will* meet this week: using Emacs and ML
- Material often (but not always!) also covered in reading notes / videos

Reading Notes and Videos

- Posted for each lesson and/or unit
  - Go over most (all?) of the material (and some extra stuff)
- So why come to class?
  - Materials let us make class-time much more useful and interactive
    - Answer questions without being rushed because *occasionally* "didn't get to X; read/watch about it"
    - Can point to optional topics/videos
    - Can try different things in class, not just recite things
- Don't need other textbooks – Dan has basically written one

Office hours

- Regular hours TBD
  - Times will be posted on course website and announced
  - Changes as necessary announced on message board
  - Will be run via Zoom
- Use them
  - Ideally not *just* for homework questions (but that’s OK too)
- Will be logistically challenging this quarter
  - Use message board as much as possible

Our Remote World

- Instruction fully online this quarter
- We will primarily use Zoom
  - Please read and follow the best practices on the website
- Lots of interesting and new challenges
  - We will do our best to support
  - Communicate with us!
- Put your health, safety, and wellbeing (and that of your loved ones) first

Homework

- Seven or eight total
- To be done individually
  - Unless otherwise instructed
- Doing the homework involves:
  1. Understanding the concepts being addressed
  2. Writing code demonstrating understanding of the concepts
  3. Testing your code to ensure you understand and have correct programs
  4. "Playing around" with variations, incorrect answers, etc.

  (2) and (3) are graded, but focusing only on (2) and (3) makes homework assignments more difficult
- Challenge problems: Low points/difficulty ratio

Note CSE 341 writing style

- Homeworks tend to be worded very precisely and concisely
  - Written "as a computer scientist" (a good thing!)
  - Technical issues deserve precise technical writing
  - Conciseness values your time as a reader
  - You should try to be precise too

- *Skimming or not understanding why a word or phrase was chosen can make the homework assignment more difficult*

- By all means ask if a problem is confusing
  - Being confused is normal and understandable
  - And I may have made a mistake

Quizzes

- Three or four total
- Conducted online during given window
  - E.g. available for 24 hours, complete within 1 hour
- To be done *individually*
  - Unless otherwise instructed
- Same concepts, but different format from homework
  - More conceptual (but write code too)
  - Open book/notes

Academic Integrity

- Read the course policy carefully
  - Clearly explains how you can and cannot get/provide help on homework and projects

- Always explain any unconventional action

- I am a great believer in and enforcer of academic integrity
  - Great trust with little sympathy for violations
  - Honest work is the most important feature of a university

- This course especially: Do not web-search for homework solutions! We will check!

Coursera (more info in document)

- Dan has taught this material to thousands of people around the world
  - A lot of work and extremely rewarding
- You are not allowed to participate in that class!
  - Do not web-search related to homework problems!
- This should have little impact on you
  - Two courses are separate
  - 341 is a great class and staff is committed to adding value above what you would get from Coursera

Has Coursera help/hurt 341?

- Biggest risks
  - Becomes easier to cheat – don’t! (And we’ve changed things)
  - Instructors become too resistant to change – hope not!

- There are benefits too
  - The videos
  - More robust grading scripts
  - Way fewer typos
  - Easier software installation (new SML Mode)
  - Taking the “VIP version” of a more well-known course
  - Change the world to be more 341-friendly

Questions?

Anything I forgot about course mechanics before we discuss, you know, programming languages?

What this course is about

- Many essential concepts relevant in any programming language
  - And how these pieces fit together
- Use ML, Racket, and Ruby languages:
  - They let many of the concepts “shine”
  - Using multiple languages shows how the same concept can “look different” or actually be slightly different
  - In many ways simpler than Java
- Big focus on *functional programming*
  - Not using *mutation* (assignment statements) (!)
  - Using *first-class functions* (can’t explain that yet)
  - But many other topics too

Why learn this?

This is the “normal” place for course motivation

- Why learn this material?

But in our experience, we don’t yet have enough shared vocabulary

- So 3-4 week delay on motivation for functional programming
- I promise full motivation: delay is worth it
- (Will motivate immutable data at end of “Unit 1”)

341 claim

Learning to think about software in this “PL” way will make you a better programmer even if/when you go back to old ways

It will also give you the mental tools and experience you need for a lifetime of confidently picking up new languages and ideas

A strange environment

- Next 4-5 weeks will use
  - ML language
  - Emacs editor
  - Read-eval-print-loop (REPL) for evaluating programs
- Need to get things installed and configured
  - Either in the department labs or your own machine
  - We’ve written thorough instructions (questions welcome)
- Only then can you focus on the content of Homework 1
- Working in strange environments is a CSE life skill

Mindset

- "Let go" of all programming languages you already know

- For now, treat ML as a "totally new thing"
  - Time later to compare/contrast to what you know
  - Lots of subtle, non-obvious differences that pop up at unexpected times
  - You *might* be able to get away with "oh that seems kind of like this thing in [Java]" for a while
  - But this will eventually confuse you, slow you down, and cause you to learn less