Class Logistics
Welcome!

- We have 14 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: “Uses ML, Python and D”
- Today’s class
  - Course mechanics
  - Introduction to language paradigms
Consise To-do List

- Read course website. Lots of details.
- Get set up using Emacs [optional; recommended]
Homework

- Five total
- To be done individually
- Doing 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. Only (2) is graded, but focusing on (2) makes homework harder
Note my Writing Style

- Homeworks tend to be worded very precisely and concisely
  - I am a computer scientist and I write like one (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 harder

- By all means ask if a problem is confusing
  - Being confused is normal and understandable
  - And I may have made a mistake
  - Once you’re unconfused, you might agree the problem wording didn’t cause the confusion
Read the course policy carefully. Clearly explains how you can and cannot get/provide help on homework and projects.

Always explain any unconventional action.

This course especially: Do not web-search for homework solutions!
What this course is about

- Many essential concepts relevant in any programming language—how these pieces fit together
- Use ML, Python, and D languages:
  - They let many of the concepts “shine”
  - Using multiple languages shows how the same concept can “look different” or actually be slightly different
  - The first two are in many ways simpler than Java
- Start with functional programming
  - Not using mutation (assignment statements) (!)
  - Using first-class functions (can’t explain that yet)
  - But many other topics too
- Big focus on multi-paradigm D language
Language Paradigms
Models of Computation

- Imperative Model
- Functional Model
- Logic Model
Language Implementation

- Interpreted
- Compiled
- Virtual Machines
Interpreted

Source Program

Parser

Interpreter (Machine Language)

AST

Code Generator

Bytecode Instructions

Virtual Machine

Operating System

Raw Hardware I/O Devices

CPU