Intro to Racket
Don’t worry — this is still CS 42!

Compute the *inner product*:

\[(1 2 3) \cdot (6 5 4)\]

Definition of inner product:

\[a \cdot b = a^T b\]
CS 42: Principles & Practice of Computer Science
The Principles in CS 42

1. **Theory of computation • Machines** (~4 weeks)
   What do we mean by *computer*?
   Programs $\iff$ Data

2. **Functional programming** (~4 weeks)
   Recursion
   Functions $\iff$ Data

3. **Problem-solving techniques** (~4 weeks)
   How do we match a problem with its solution?
   How good is a solution?
   Algorithms + Data Structures

4. **Object-oriented programming** (~3 weeks)
   How do we structure a program so that it can grow and change?

5. **Logical programming** (~1 week)
   How does language influence thought?
What is “Functional Programming”?
Read and discuss: Backus paper

What’s something you agree with?
What’s something you disagree with?
What’s something you don’t (yet) understand?
Why do you think the highlighted text is highlighted?
Can you sum up the reading in at most six words?
Functional programming
A working definition

No mutable state (no “assignments”)

Functions, functions, functions

Say more “what”; say less “how”.
Prior experience: programming languages
Racket—a functional programming language

A working definition

Variables

Functions

Say more “what”; say less “how”.
How do we learn a new programming language?
Racket Demo

(see notes on handout)
Take-home exam #1

**Available:** this Friday night, outside Olin 1279

**Must return by:** next Friday at 5pm

**Time-limit:** 75 minutes

**Covers:** everything up to and including this week but more focus on automata, circuits, and assembly

**Resources:** Open-note (but no computer)

**Honor code:** don’t discuss exam questions

There will be an **assignment on Thursday**, too.