CSE 341
Section 3

HW1 Debrief, Tail Recursion, More Pattern-Matching
Winter 2019

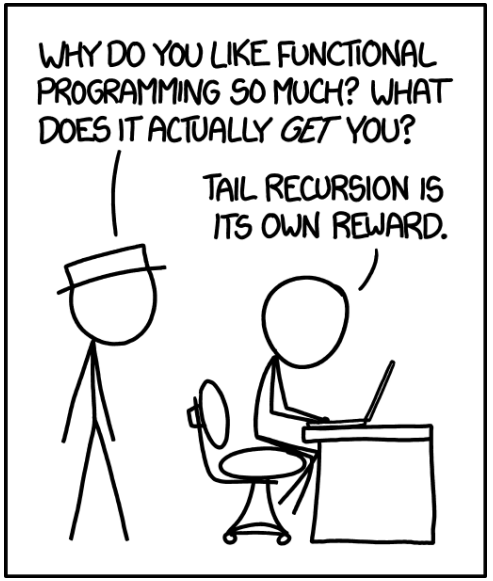
Learning Objectives

- More tail recursion (QC, ~15 min)
  - What is and isn't tail recursive?
  - How can we make functions tail recursive?
  - When *can't* we be tail recursive?
- Pattern-matching over expression trees
- Function patterns(?)

Key Concepts Review

- Custom datatypes
  - all of (records), one of (variants)
- Pattern matching
  - Powerful way to break apart data
- Tail recursion
  - Space efficiency of loops with recursive functions

Tail Recursion



What is it?

Briefly: if a function will immediately return after making a call, we can reuse the stack space of the current function.

Tail Recursion

Quickcheck! (6 minutes, ungraded)

Speak with a neighbor if you'd like.

Tail Recursion

Was length tail recursive?

Was all_positive tail recursive?

Why tail recursion?

Let’s look at more examples of tail recursion

Pattern Matching

- We can pattern match over `datatype`s
- Beware “non-exhaustive matching”
  - Pattern matching can avoid “empty list” exceptions!
- Most functions pattern match over a single argument
  - SML has special syntax for this common case!
  - Use is a matter of taste
- Live demo