



Case Study

Principles of Functional Programming

The Water Pouring Problem

- ▶ You are given some glasses of different sizes.
- ▶ Your task is to produce a glass with a given amount of water in it.
- ▶ You don't have a measure or balance.
- ▶ All you can do is:
 - ▶ fill a glass (completely)
 - ▶ empty a glass
 - ▶ pour from one glass to another until the first glass is empty or the second glass is full.

Example task:

You have two glasses. One holds 7 units of water, the other 4. Produce a glass filled with 6 units of water.

Strategy

States and Moves

Representations:

Glass: Int (glasses are numbered 0, 1, 2)

State: Vector[Int] (one entry per glass)

I.e. Vector(2, 3) would be a state where we have two glasses that have 2 and 3 units of water in it.

Moves:

Empty(glass)

Fill(glass)

Pour(from, to)

Variants

In a program of the complexity of the pouring program, there are many choices to be made.

Choice of representations.

- ▶ Specific classes for moves and paths, or some encoding?
- ▶ Object-oriented methods, or naked data structures with functions?

The present elaboration is just one solution, and not necessarily the shortest one.

Guiding Principles for Good Design

- ▶ Name everything you can.
- ▶ Put operations into natural scopes.
- ▶ Keep degrees of freedom for future refinements.