Lecture 24

CPSC 110

Peyton Seigo

2018-10-31

Lecture 24

Abstract Fold Functions

Creating a `fold` function from an existing mutually referential template

1. Encapsulate and name `fold-<Type>`
   - `@template <Type1> <Type2> ... <TypeN> encapsulated add-param`
2. Write `check-expect`s
3. Purpose
   - "produce the abstract fold function for `<Type>`
4. Signature
   - Assign type parameters to each function and argument used in your template. This makes writing the signature much easier.
   - `@HtDF fold-<Type>`

We do not need a stub.

Notes

- For `use-abstract-fn` functions, we do not tag the types it consumes
  - i.e. for a fn with `@signature Region -> Region` that uses an abstract-fn, our template is just `@template use-abstract-fn` (assuming we don't have `fn-composition` or anything else)
- When filling in arguments for a `fold` function, ask yourself: "Is there a built-in function for what I want to do?"
  - If the answer is yes, use that function!
  - If the answer is no, write a local function to do it!