**Functional versus Imperative Maps**

Goal is to map identifiers to definitions

We need efficient operations on *maps*

* look up identifier to obtain its definition
* update map to store declarations
* revert declarations

Functional style

* correspond to [Notation for Maps](http://lara.epfl.ch/w/cc09:notation_for_maps)
* [standard Map class in Scala](http://www.scala-lang.org/docu/files/api/scala/collection/immutable/Map.html)
* update returns new map, we get to keep the old

def + : (kv : (A, B)) : Map[A, B]

* if we write analysis recursively, we just keep values of old maps

Imperative style

* global map
* insert new values when entering scope
* undo changes on exit
  + need an undo stack to know what to remove
  + need data structure to support efficient removal