Processing two complex pieces of data

Or, patterns of recursion revisited
Recall

- Previously we defined a type for Naturals.
- Different functions used different schemes (templates) for recursion:
  - `add`, `multiply`: on first
  - `exp`: on second
  - `minus`, `geq`: lockstep/compound
  - `divi`, `mod`: strong induction
More examples

We’ll perform a similar analysis on functions that consume multiple lists

Simple examples:

- **append** : one of the lists
- **n-th** : lockstop
- **list-pick** : combination
- **skip** : strong induction
Other functions

- **merge** : ? (for hwk 😊)
- **equality** : lockstep
- **cross** : induction on one
- **interleave** : lockstep
- **vector-add** : lockstep
- **sublist?** : induction on one
- **decode** : ? (for next hwk 😊)