Programming Paradigms

Imperative Programming

- Modifying mutable variables
- Using assignments
- Control structures such as if-then-else, loops, break, continue, return
- Strong correspondence between:
  - ∗ Mutable variables – Memory Cells
  - ∗ Variable deferences – Load instructions
  - ∗ Variable assignments – Store instructions
  - ∗ Control structures – Jumps

Problem: Scaling up – *One tends to conceptualize data structures word-by-word.*
Ideally*:* Develop theories of collections, shapes, strings – normally a theory does not describe mutations(they can destroy useful laws in the theories)

⇨ Therefore, let's:
- ∗ Concentrate on defining theories for operators expressed as functions
- ∗ Avoid mutations
- ∗ Have powerful ways to abstract and compose functions

Functional Programming

- In a restricted sense, functional programming means programming WITHOUT **variables, assignments, loops** and **other imperative control structures**
- In a wider sense, functional programming means focusing on the functions and immutable data
- In particular, functions can be values that are produced, consumed and composed
  - ∗ They can be defined anywhere, including inside functions
  - ∗ They can be passed as parameters to functions and returned as results
  - ∗ There exists a set of operators to compose functions