Data and Values

- Program’s make extensive use of data
- They manipulate values—they are stored, evaluated, and passed around
Type

- Values are often grouped into types if there can be defined operations that can act on them uniformly.
- Type is a set of values.
- $v$ is of type $T$ means $v \in T$.
- An expression or function is of type $T$ their result $\in T$.

Data Types

- Primitive Types—can’t be decomposed (int, double, etc.).
  - Functional languages provide atoms.
- Composites (Array, Union, Object).
  - Functional languages have special treatment of lists.
- Recursive (has other values of same type).
List Handling

- Functional Languages provide special handling of lists
- Constant performance to fetch head
- Operations: head, rest or tail

Static vs. Dynamic

- Static typing is when type check or verification happens at compile time
- Dynamic typing is when it happens at runtime
**Strong vs. Weak**

- Strong typing is where the type is known and ultimately verified.
- Weak typing is where the language does not really enforce types at all even at run time.
Capability vs. Contract

- Static languages often provide design by contract (Bertrand Meyers)
  - Useful to do checks and verification
- Dynamic languages often prefer design by capability
  - More flexible