Intro to Scheme

Elements of Programming

- Primitive expressions - simplest entities
- Means of combination - build
- Means of abstraction - manipulate (procedure → rules)

Expressions

- Primitive - e.g. 48
- Compound - application e.g. (+ 12 34)
  - Operator before operands; nesting - (+ (- 9 8) (+ 3 4))
  - (Run → evaluate → print) loop

Evaluating Combinations

- Evaluate the subexpressions of the combination - recursive
- Apply to others / work outwards

Compound Procedures

- Numbers and arithmetic & nesting of combinations
- Define - associate name with value
  - General: define (<name> <formal parameters>) (<body>)
  - (define (square x)(* x x))

Substitution Model

- Formal parameter replaced by the corresponding argument
  - Normal order: fully expand and then reduce
    - Takes arguments and passes it to procedure without actually evaluating it yet
  - Applicative: evaluate the arguments and then apply
    - Takes arguments and evaluates, then gives the result to the procedure

Conditional Expressions & Predicates

- Predicate: expression whose value is true or false
- Cond: take tests to perform different operations

  -
    ```
    (cond (<predicate 1> <consequent expression>)
          (<p2> <e2>)
          (<pn> <en>))
    ```

    - OR (if <predicate> <consequent> <alternative>)
- Logical compositions
    - And, or, not