The Design Recipe

1. The **purpose** describes what the function calculates. Explain the role of every parameter.
   
   ```scheme
   ;; (p? n) produce true if n is prime; false otherwise
   ```

2. The **contract** indicates the type of arguments the function consumes and the value it produces. Can be **Num**, **Int**, **Nat**, or other types.
   
   ```scheme
   ;; p?: Nat -> Bool
   ```

3. Choose **examples** which help the reader understand the purpose.
   
   ```scheme
   ;; Examples:
   (check-expect (p? 9) false)
   (check-expect (p? 17) true)
   ```

4. The **implementation** is interpreted by the computer.
   
   ```scheme
   (define (p? n)
     (and (not (= n 1)) (d-over n 2)))
   ```

5. The **tests** resemble examples, but are chosen to try to find bugs in the implementation.
   
   ```scheme
   ;; Tests
   (check-expect (p? 1) false)
   ```