# Functions

* Should be small (3-4 lines).
* Should do one thing. It should not have any side effects.
* if the keyword try exists in a function, it should be the very first word in the function and that there should be nothing after the catch/finally blocks.
* Should have descriptive names.
* The ideal number of arguments for a function is zero (niladic). Next comes one (monadic), followed closely by two (dyadic). Three arguments (triadic) should be avoided where possible.
* Do not use flag arguments, instead use two functions.
* Output arguments should be avoided.
* Prefer exceptions to returning error codes.
* The occasional multiple return, break, or continue statement does no harm and can sometimes even be more expressive than the single-entry, single-exit rule.

# Comments

* Minimize comments.
* Clear and expressive code with few comments is far superior to cluttered and complex code with lots of comments.
* Eventually the comments begin to lie as the code around them changes.

# Formatting

* Concepts that are closely related should be kept vertically close to each other.
* If one function calls another, they should be vertically close, and the caller should be above the callee.
* We expect the most important concepts to come first, and the low-level details to come last.

# Objects and Data Structures

* The Law of Demeter says that a method *f* of a class *C* should only call the methods of these:
  + *C*
  + An object created by *f*
  + An object passed as an argument to *f*
  + An object held in an instance variable of *C*
* The method should not invoke methods on objects that are returned by any of the allowed functions. In other words, talk to friends, not to strangers.

# Error Handling

* Mention the operation that failed and the type of failure.
* Page 140