7.3. Good Routine Names

Code Complete

- A good name for a routine clearly describes everything the routine does.

- In the routine's name, describe all the outputs and side effects.

  - ❑ ¿Que sucede si existen muchos efectos laterales?
    The cure is not to use less-descriptive routine names; the cure is to program so that you cause things to happen directly rather than with side effects.

- Avoid meaningless, vague, or wishy-washy verbs.
  Some verbs are elastic, stretched to cover just about any meaning. Routine names like `HandleCalculation()`, `PerformServices()`, `OutputUser()`, are OK.

- In other cases, the verb is vague because the operations performed by the routine are vague. The routine suffers from a weakness of purpose, and the weak name is a symptom.

- Don't differentiate routine names solely by number.
  One developer wrote all his code in one big function. Then he took every 15 lines and created functions named `Part1`, `Part2`, and so on. The numerals at the ends of these names provide no indication of the different abstractions the routines represent.

- Make names of routines as long as necessary.
  Research shows that the optimum average length for a variable name is 9 to 15 characters. Routines tend to be more complicated than variables, and good names for them tend to be longer. On the other hand, routine names are often attached to object names, which essentially provides part of the name for free.

- To name a function, use a description of the return value.
  A function returns a value, and the function should be named for the value it returns. For example: `cos()`, `customerId.Next()`, `printer.IsReady()`.

- To name a procedure, use a strong verb followed by an object.
  A procedure with functional cohesion usually performs an operation on an object. The name should reflect what the procedure does, and an operation on an object implies a verb-plus-object name. Example: `PrintDocument()`.

  - ❑ In object-oriented languages, you don't need to include the name of the object in the procedure name because the object itself is included in the call. You invoke routines with statements like document.Print(), orderInfo.Check(), and monthlyRevenues.Calc(). Names like document.PrintDocument() are redundant and can become inaccurate when they're carried through to derived classes.

- Use opposites precisely. Using naming conventions for opposites helps consistency, which helps readability. Opposite-pairs like first/last are commonly understood.

- Establish conventions for common operations. In some systems, it's important to distinguish among different kinds of operations.

  Example:
  In one of our projects, we neglected to establish a convention for naming the routines that would return the object identifier, so we had routine names like these:
  ```
  employee.id.Get()
  dependent.GetId()
  supervisor()
  candidate.id()
  ```
  for the same operation.