ALGORITHMS AND PROGRAMS

0.1 REQUIREMENTS TO ALGORITHMS

Requirements specify what's wanted overall by an application. They result in a set of problems that must be implemented by functions. This chapter briefly reviews the requirements concept.

The Meaning of Requirements

Every program is written to provide benefits for its stakeholders. These benefits are its *functional requirements*[1]. For a "Hello World" program, the functional requirement can be expressed as:

> *'Hello World' shall be[2] on the console.*

An example requirement for a climate-change application is
> *"The predicted number of years until the average sea-level rise since 2020 becomes one inch*
> *shall appear at http://abc ..."*

Functional requirements are concerned with *what* a program is intended to accomplish for its users, not *how* it will do so (e.g., *not* "using Python" or "by means of a neural net"). Every requirement statement is a *predicate*—a statement that can be either true or false. The statements "my yard is less than an acre" and "x == y + x", are predicates. The statements "put your foot on the gas" and "x = y + z" (as computer code) are not.

We formulate requirements as early as we can in the process because it's obviously better to know what you want an application to do for its users before setting out to build it. For small applications, it may be feasible to formulate requirements for the entire application at once. For other applications, it can be difficult to express up front all desired functionality—simply because envisioning a nontrivial system in its entirety can be much too demanding. Consequently, we typically traverse a development loop[3]

---

[1] Non-functional requirements *qualify* functional ones. For example, "Hello World" shall appear within one microsecond."

[2] It is customary to express requirements with "shall" because this envisions a future time, when the application is complete. Logically, the present tense is better, as in "Hello World" is on the console.

[3] –essentially the agile methodology

specify some requirements → build code that satisfies them → observe the partial application in action → specify more requirements →…

Requirements are decomposed into problems that must be implemented by functions that, in turn, use algorithms.