The industry of pure ideas

1.1 Their machines and ours

Engineers design and build machines. A car is a machine for traveling; an electronic circuit is a machine for transforming signals; a bridge is a machine for crossing a river. Programmers — “software engineers” — design and build machines too. We call our machines programs or systems.

There is a difference between our machines and theirs. If you drop one of their machines, it will hurt your feet. Ours won’t.

Programs are immaterial. This makes them closer, in some respects, to a mathematician’s theorems or a philosopher’s propositions than to an airplane or a vacuum cleaner. And yet, unlike theorems and propositions, they are engineering devices: you can operate a program, like you operate vacuum cleaners or planes, and get results.

Since one cannot operate a pure idea you will need some tangible, material support to operate programs or, using the more common terms, to run or execute them. That support is another machine: a computer. Computers and related devices are called hardware, indicating that — although they are getting ever lighter — computers are the kind of machine that will hurt your feet. Programs and all that relates to them are by contrast called software, a word made up in the 1950s when programs emerged as topic of interest.

Here is how things work. You dream up a machine, big or small, and describe your dream in the form of a program. The program can then be fed into a computer for execution. The computer by itself is a general-purpose machine, but when equipped with your program it becomes a specialized machine, a material realization of the immaterial machine that you defined through your program.
The person who writes the program — “you” in the previous paragraph — is predictably called a programmer. Others, known as users, can then run your program on your computer, or theirs.

If you have used a computer, you have already run some programs, for example to browse the Web or play a DVD, so you already are a user. This book should help you make it to the next step: programmer.

Cynics in the software industry pronounce “user” as “loser”. It’s one of the goals of this book that users of your program will pronounce themselves winners.

The immaterial nature of the machines we build is part of what makes programming so fascinating. Given a powerful enough computer you can define any machine you want, whose operation will require billion upon billion of individual steps; and the computer will run it for you. You do not need wood or clay or iron or a hammer or anything that could wear you out carrying it up the stairs, burn you, or damage your clothes. State what you want, and you will receive it. The only limit is your imagination.

All right, it is one of two limits; we avoid mentioning the other in genteel company, but you will likely encounter it before long; it is your own fallibility. Nothing personal: if you are like me and the rest of us, you make mistakes. Lots of mistakes. In ordinary life they are not all harmful, as most human activities are remarkably error-tolerant. You can press your fork a little too intensely, swallow water a little too fast, push the accelerator a little too hard, use the wrong word; this happens all the time and in most cases does not prevent you