Though syntax errors are the most common kind of error (especially for those learning a new programming language), they are the least dangerous kind of error. Every serious programming language does a complete job of detecting syntactic errors, and will not allow users to execute a program with even one syntactic error. Furthermore, in most cases the language system gives a sufficiently clear indication of the location of the error that it is obvious what needs to be done to fix it.

The situation with respect to static semantic errors is a bit more complex. Some programming languages, e.g., Java, do a lot of static semantic checking before allowing a program to be executed. Others, e.g., C and Python (alas), do relatively less static semantic checking. Python does do a considerable amount of static semantic checking while running a program. However, it does not catch all static semantic errors. When these errors are not detected, the behavior of a program is often unpredictable. We will see examples of this later in the book.

One doesn’t usually speak of a program as having a semantic error. If a program has no syntactic errors and no static semantic errors, it has a meaning, i.e., it has semantics. Of course, that isn’t to say that it has the semantics that its creator intended it to have. When a program means something other than what its creator thinks it means, bad things can happen. What might happen if the program has an error, and behaves in an unintended way?

It might crash, i.e., stop running and produce some sort of obvious indication that it has done so. In a properly designed computing system, when a program crashes it does not do damage to the overall system. Of course, some very popular computer systems don’t have this nice property. Almost everyone who uses a personal computer has run a program that has managed to make it necessary to restart the whole computer.

Or it might keep running, and running, and running, and never stop. If one has no idea of approximately how long the program is supposed to take to do its job, this situation can be hard to recognize.

Or it might run to completion and produce an answer that might, or might not, be correct.

Each of these is bad, but the last of them is certainly the worst, When a program appears to be doing the right thing but isn’t, bad things can follow. Fortunes can be lost, patients can receive fatal doses of radiation therapy, airplanes can crash, etc