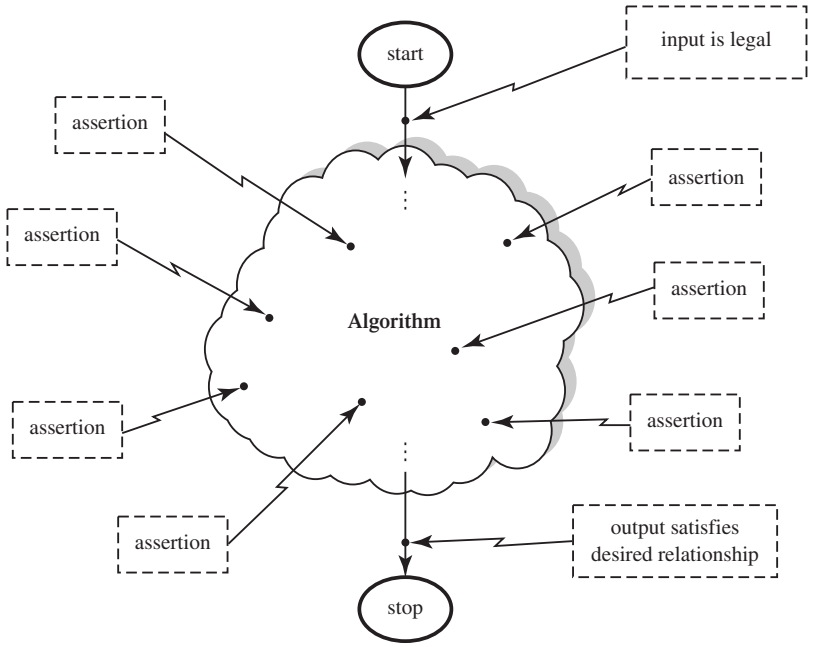

Figure 5.5

Annotating an algorithm with invariants.

Now, suppose we can establish that all the assertions we have attached are indeed invariants, meaning that they are true whenever reached. Then, in particular, the final assertion is also an invariant. But this means that the algorithm is partially correct. Hence all we have to do is establish the invariance of our assertions. This is done by establishing certain local properties of our assertions, sometimes called **verification conditions**, to the effect that proceeding locally from checkpoint to checkpoint does not bring about any violations of the invariance properties. This approach to proving correctness is sometimes called the **invariant assertion method**, or **Floyd's method**, after one of its inventors.

How do we go about choosing checkpoints and intermediate assertions, and how do we establish the verification conditions? The example given in the next section should shed some light on these questions.

Turning from partial correctness to termination, our main interest is in showing that something good eventually happens (not that bad things do not); namely, that the algorithm indeed reaches its endpoint and terminates successfully. To prove such a statement we use checkpoints as before, but we now find some quantity depending on the algorithm's variables and data structures, and show that it **converges**. By this we mean that the quantity keeps decreasing as execution proceeds from one checkpoint to another, but that it cannot decrease forever—we need to show that there is some bound below which it can never go. Hence there is no way for the algorithm to run forever, since the **convergent**, as it is sometimes called, would then have to decrease forever, contradicting this bound.

In a sorting algorithm, for example, the number of elements not yet in their final positions in the sorted list might be shown to decrease as execution proceeds, but never to be less than 0. When that number reaches 0, the algorithm presumably terminates.