5.7. THE PROGRAMMING LANGUAGE PL/0

The remaining sections of this chapter are devoted to the development of a compiler for a language to be called PL/0. The necessity of keeping this compiler reasonably small in order to fit into the framework of this book and the desire to be able to expose the most fundamental concepts of compiling high-level languages constitute the boundary conditions for the design of this language. There is no doubt that either an even simpler or a much more complicated language could have been chosen; PL/0 is one possible compromise between sufficient simplicity to make the exposition transparent and sufficient complexity to make the project worthwhile. A considerably more complicated language is PASCAL, whose compiler was developed using the same techniques, and whose syntax is shown in Appendix B.

As far as program structures are concerned, PL/0 is relatively complete. It features, of course, the assignment statement as the basic construct on the statement level. The structuring concepts are those of sequencing, conditional execution and repetition, represented by the familiar forms of **begin/end-**, **if-**, and **while** statements. PL/0 also features the subroutine concept and, hence, contains a procedure declaration and a procedure call statement.

In the realm of data types, however, PL/0 adheres to the demand for simplicity without compromise: integers are its only data type. It is possible to declare constants and variables of this type. Of course, PL/0 features the conventional arithmetic and relational operators.

The presence of procedures, that is, of more or less "self-contained" partitions of a program offers the opportunity to introduce the concept of *locality* of objects (constants, variables, and procedures). PL/0 therefore

features declarations in the heading of each procedure, implying that these objects are understood to be local to the procedure in which they are declared.

This brief introduction and overview provide the necessary intuition to understand the syntax of PL/0. This syntax is presented in Fig. 5.4 in the form of seven diagrams. The task of transforming the diagrams into a set of equivalent BNF-productions is left to the interested reader. Fig. 5.4 is a convincing example of the expressive power of these diagrams which allow formulation of the syntax of an entire programming language in such a concise and readable form.

The following PL/0 program may demonstrate the use of some features that are included in this mini-language. The program contains the familiar algorithms for multiplication, division, and finding the greatest common divisor (gcd) of two natural numbers.

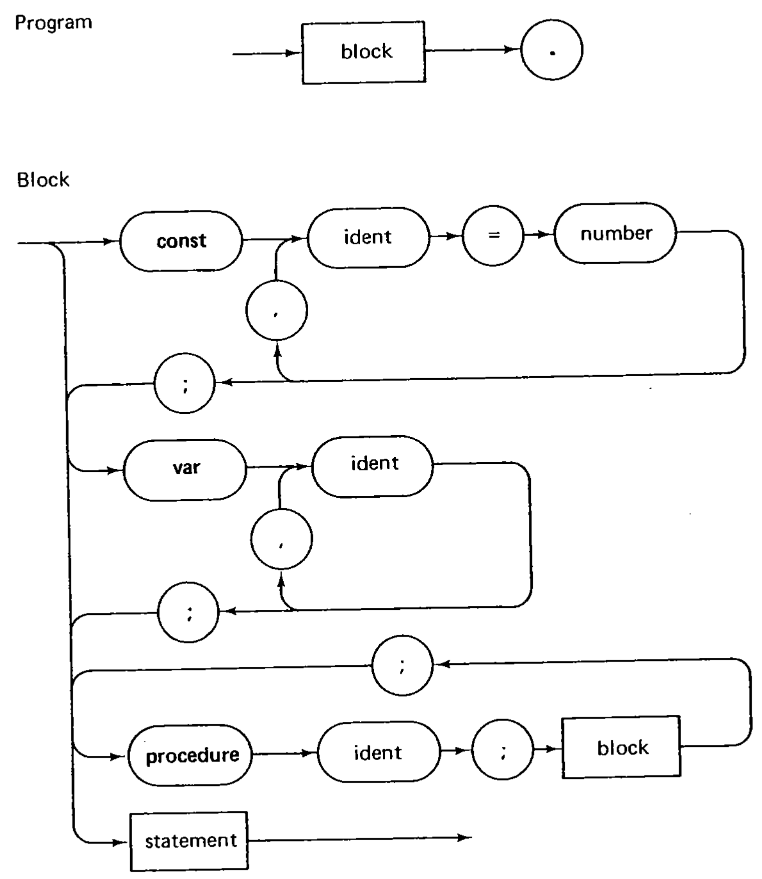
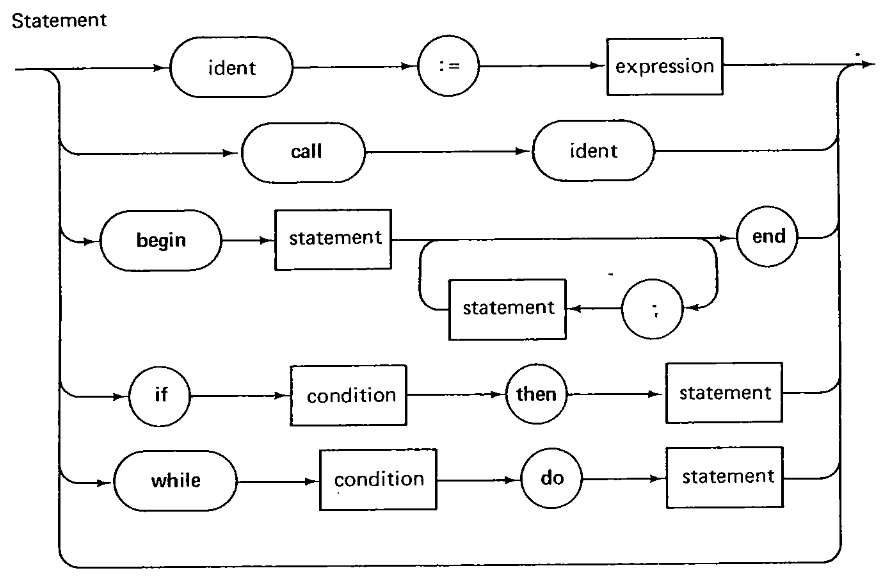


Fig. 5.4 Syntax of PL/0.



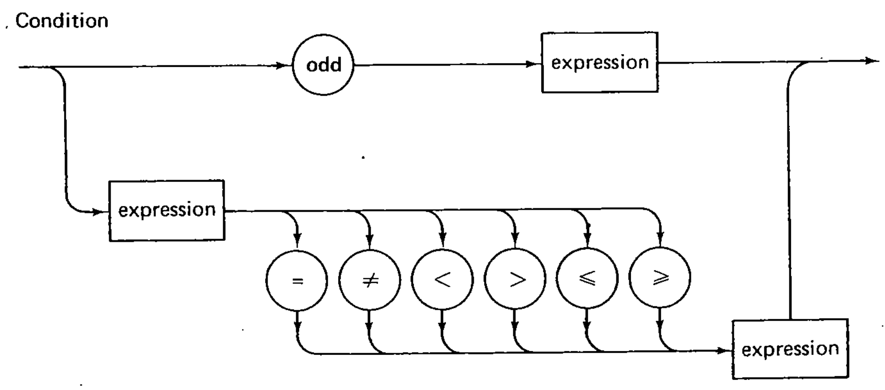
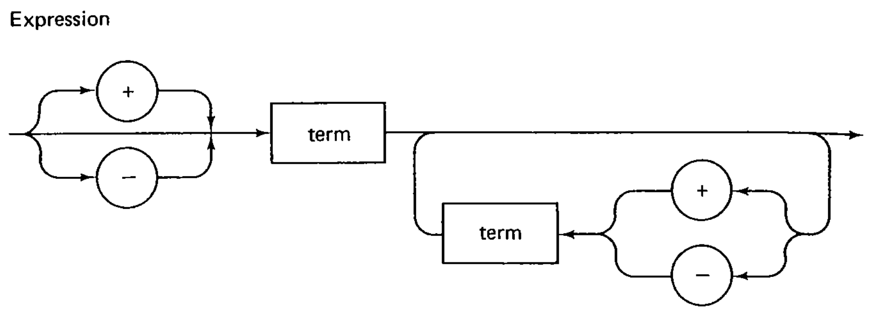
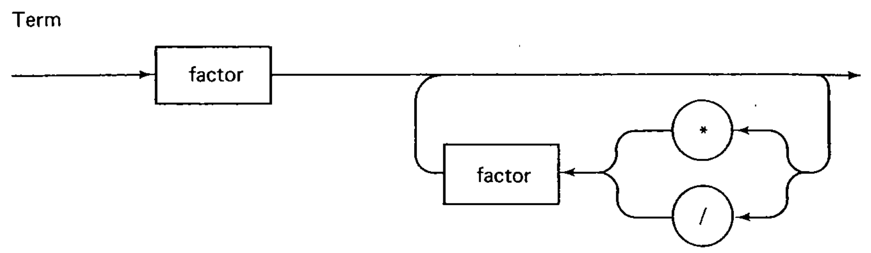


Fig. 5.4 (Continued)





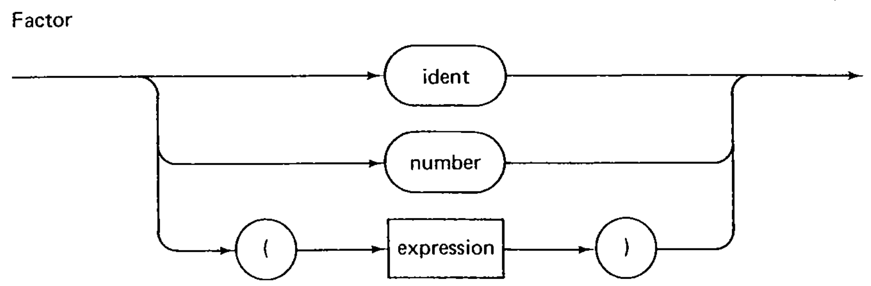


Fig. 5.4 (Continued)

```
const m = 7, n = 85;
var x,y,z,q,r;
procedure multiply;
  var a,b;
begin a := x; b := y; z := 0;
    while b > 0 do
    begin                                  (5.14)
      if odd b then z := z + a;
      a := 2*a; b := b/2;
    end
end ;
```

```
procedure divide;
    var w;
begin r := x; q := 0; w := y;
    while w ≤ r do w := 2*w;
    while w > y do
        begin q := 2*q; w := w/2;                    (5.15)
            if w ≤ r then
                begin r := r-w; q := q+1
                end
        end
end ;
procedure gcd;
    var f,g;
begin f := x; g := y;
    while f ≠ g do
        begin if f < g then g := g-f;                (5.16)
              if g < f then f := f-g;
        end ;
    z := f
end ;
begin
    x := m; y := n; call multiply;
    x := 25; y := 3; call divide;
    x := 84; y := 36; call gcd;
end .
```