Top Down Operator Precedence

Vaughan R. Pratt *

Massachusetts Institute of Technology 1973

1   Survey of the Problem Domain

There is little agreement on the extent to which syntax should be a consideration in the design and implementation of programming languages. At one extreme, it is considered vital, and one may go to any lengths [Wijngaarden, 1969, McKeeman et al., 1970] to provide adequate syntactic capabilities. The other extreme is the spartan denial of a need for a rich syntax [**?**]. In between, we find some language implementers willing to incorporate as much syntax as possible provided they do not have to work hard at it [**?**].

In this paper we present what should be a satisfactory compromise for a respectably large proportion of language designers and implementers. We have in mind particularly

1. those who want to write translators and interpreters (soft, firm or hard-wired) for new or extant languages without having to acquire a large system to reduce the labor, and

2. those who need a convenient yet efficient language extension mechanism accessible to the language user.

References

[McKeeman et al., 1970] McKeeman, W. M., Horning, J. J., and Wortman, D. B. (1970). *A Compiler Generator.* Publisher Englewood Cliffs, N.J., Prentice-Hall.

[Minsky, 1970] Minsky, M. (1970). Form and content in computer science (1970 acm turing lecture). *J. ACM*, 17(2):197–215.

[Wijngaarden, 1969] Wijngaarden, A. v. (1969). *Report on the Algorithmic Language ALGOL 68.* Printing by the Mathematisch Centrum.

[Wirth, 1971] Wirth, N. (1971). The programming language pascal. *Acta Inf.*, 1(1):35–63.