Hope (programming language)

Hope is a small functional programming language developed in the 1970s at the University of Edinburgh.[1][2] It predates Miranda and Haskell and is contemporaneous with ML, also developed at the University. Hope was derived from NPL,[3] a simple functional language developed by Rod Burstall and John Darlington in their work on program transformation.[4] NPL and Hope are notable for being the first languages with call-by-pattern evaluation and algebraic data types.[5]

Hope was named for Sir Thomas Hope (c. 1681–1771), a Scottish agricultural reformer, after whom *Hope Park Square* in Edinburgh, the location of the Department of Artificial Intelligence at the time of the development of Hope, was also named.

Contents

Language details

A factorial program in Hope is:

```
dec fact : num -> num;
--- fact 0 <= 1;
--- fact n <= n*fact(n-1);
```

Changing the order of the clauses does not change the meaning of the program, because Hope's pattern matching always favors more specific patterns over less specific ones. Explicit type declarations in Hope are required; there is no option to use a type-inference algorithm in Hope.

Hope provides two built-in data structures: tuples and lists.[6]

Implementations

The first implementation of Hope was strict, but since that one there have been lazy versions and strict versions with lazy constructors. British Telecom embarked on a project with Imperial College to implement a strict version. The first release was coded by Thanos Vassilakis in 1986. Further releases were coded by Mark Tasng of British Telecom. A successor language Hope+ (developed jointly between Imperial College and International Computers Limited (ICL) added annotations to dictate either strict or lazy evaluation.[7]

Roger Bailey's Hope tutorial in the August 1985 issue of *BYTE* references an interpreter for IBM PC DOS 2.0.[6]

References

1. Burstall R.M, MacQueen D.B, Sannella D.T. (1980) *Hope: An Experimental Applicative Language*. Conference Record of the 1980 LISP Conference, Stanford University, pp. 136-143.
2. Bailey, Roger (1 April 1990). *Functional Programming with Hope*. Ellis Horwood Series in Computers and Their Applications. Ellis Horwood Ltd.
3. R.M. Burstall. Design considerations for a functional programming language. Invited paper, Proc. Infotech State of the Art Conf. “The Software Revolution”, Copenhagen, 45–57 (1977)
4. R.M. Burstall and J. Darlington. A transformation system for developing recursive programs. Journal of the Association for Computing Machinery, 24(1):44–67 (1977)
5. Hudak, Paul; Hughes, John; Peyton Jones, Simon; Wadler, Philip (2007-06-09). *A history of Haskell: being lazy with class*. ACM. pp. 12–1. doi:10.1145/1238844.1238856 (https://doi.org/10.1145%2F1238844.1238856). ISBN 9781595937667. S2CID 52847907 (https://api.semanticscholar.org/CorpusID:52847907).
6. Bailey, Roger (August 1985). "A Hope Tutorial" (https://archive.org/stream/BYTE_Vol_10-08_1985-08_The_Amiga#page/n241/mode/2up). *BYTE*. Vol. 10, no. 8. Retrieved 1 April 2015.
7. John Kewley and Kevin Glynn. Evaluation Annotations for Hope+. In Kei Davis and R. J. M. Hughes, editors, Functional Programming: Proceedings of the 1989 Glasgow Workshop, Workshops in Computing, pages 329-337, London, UK, 1990. Springer-Verlag.

External links

- Hope Interpreter for Windows (https://web.archive.org/web/20130801064002/http://www.hopemachine.co.uk/)
- Entry for Hope in the online Dictionary of Programming Languages (http://cgibin.erols.com/ziring/cgi-bin/cep/cep.pl?_key=Hope)