0 → TOTAL;
FOR X IN HYBRID H:
X + TOTAL → TOTAL
ENDFOR

(The code generated is not given—it is left to the reader to realise that it is not particularly attractive.)

Summary
What does, and what does not, constitute ‘convenience’ in programming languages is somewhat a question of individual choice. The author feels that coding for iteration is more easily understood if, whenever a data path is involved, the nature of the data path is immediately apparent.

Like many language features, the above facility is decidedly no ‘cure-all’, but its use appears advantageous for a certain range of commonly occurring situations.

The techniques presented here are only suitable for iterations involving a single data path. This is quite a serious limitation (e.g. how does one handle element-by-element assignment?). More general techniques are being developed to handle such cases (this paper involves the manipulation of text which represents program actions having multiple entries and/or multiple exits—and the use of ‘connector functions’ to combine these components in a more general manner than is allowed in ‘conventional’ structured programming—however that is a matter for further research).

One would expect the above techniques to produce (source) program code which is relatively insensitive to choices of data structure. This may allow convenient implementation of decisions concerning the mapping of abstract data structures to physical data structures when implementing very high level languages. However, this is again a topic of further research.

The author would like to thank the referee for useful suggestions on improving the presentation of this paper.

References

Book reviews
Computer Science and Technology and their Application, General Editors: N. Metropolis, E. Piore and S. Ulam, 1975; 310 pages.
Administrative Editors: Mark I. Halpern, William C. McGee; Contributing Editors: Louis Bolliet, Andrei P. Ershov, J. P. Laski. (Pergamon Press, £15.00)

Contents
A Tutorial on Data-Base Organization, R. W. Engles.
General Concepts of the Simula 67 Programming Language, J. D. Ichbiah and S. P. Morse.
Incremental Compilation and Conversational Interpretation, M. Berthaud and M. Griffiths.
An Introduction to ALGOL 68, H. Bekic.
A General Purpose Conversational System for Graphical Programming, O. Lecarme.
Automatic Theorem Proving Based on Resolution, A. Picotte.
A Survey of Extensible Programming Languages, N. Solinseff and A. Yezerski.

It appears immediately that this volume is not annual, nor is it a review; it is hardly automatic programming, and the contributing editors did not contribute. Nevertheless, it is a selection of articles on topics closely related to high-level programming languages. They might have been contributed to a learned journal; but instead they have been collected in a book. On the whole they deserve to be: the general standard of the papers is distinctly higher than the average, and they are likely to appeal more consistently to a reader interested in high-level programming languages.

But it would be a rash reviewer who would venture to pass comment on all the papers individually; and to do so within the space allotted would be even more foolhardy.

C. A. R. HOARE (Belfast)