A sublanguage of ALGOL 68

By P. G. Hibbard

This Report has been accepted by Working Group 2.1, reviewed by Technical Committee 2 on Programming and approved for publication by the General Assembly of the International Federation for Information Processing. Reproduction of the Report, for any purpose, but only of the whole text, is explicitly permitted without formality.

0. Introduction

At the meeting of Working Group 2.1 of I.F.I.P. held in Vienna in 1972, a definition of a sublanguage of ALGOL 68 was presented [1]. As a result of comments by members of the Working Group, an enhanced sublanguage was designed, and was discussed at the meeting in Dresden in 1973, at the meeting of the Subcommittee on ALGOL 68 Support in Cambridge in 1974, and in a paper presented at the 1974 International Conference on ALGOL 68 at Winnipeg [2]. A formal definition was prepared [3] and discussed by the Subcommittee in Boston in 1975. The Subcommittee appointed a Task Force to prepare the definition for submission to the full Working Group. At the Working Group meeting in Munich in 1975, this final report on the sublanguage was accepted.

0.1. Aims and principles of design

a) The sublanguage of ALGOL 68 here defined is intended for use primarily in numerical and related areas. Its field of application encompasses that of ALGOL 60; however, it possesses greater generality and more expressive power than that language permitting, for example, the definition and manipulation of structured values and the definition of operations upon them. These, and other features, are expected to lead to better structured and more efficient programs.

b) Sufficient restrictions have been imposed to allow the sublanguage to be implemented on small computers. Even "single-pass" compilation techniques allow efficient code to be generated and good diagnostic information to be provided, and essentially only "stack-oriented" storage allocation techniques are required at run time.

c) Even though the sublanguage lacks many features of the full language, it may be regarded as a complete language for its intended field of application; i.e., programs written for that application area are likely to be substantially the same as those which would be written in the full language.

0.2. Summary of contents

This report is an addendum to the Revised Report on the Algorithmic Language ALGOL 68 [4], hereafter termed "the Report". It contains modifications and additions to the Report which cause it then to define particular-programs both in the full language, without change, and in a sublanguage (Report 2.2.2.c) of the full language.

Section 1 gives an informal list of the differences between the full language and the sublanguage, and section 2 gives a commentary on the techniques which have been adopted to modify the Report, to assist the reader. The remaining sections of this report formally define the sublanguage. Section 3 gives the changes and additions to the syntax, and section 4 gives the changes and additions to the semantics. Sections 5 and 6 specify, respectively, the changes required in the standard- prelude of the Report, and the definitions which are to be included in the particular-prelude.

(No changes in the pragmatic remarks of the Report have been included, hence they will not be fully appropriate to the language described here.

A complete description of the sublanguage, using a less formal approach, is available as a companion volume [5].)

1. Sublanguage restrictions

1.1. Modes

UNITED-declarers may not occur in the sublanguage, though calls of routines which are created in the standard-prelude and have a parameter whose mode envelops some 'UNITED', e.g. print and upb, are permitted. Flexible-ROWS of MODE-declarators may not occur; however, an alternative is provided for the mode of strings (such that they may be regarded as plain values). ROWS1-of-ROWS2-of-MODE-declarators, and structured-with-
FIELDS-mode-declarators in which 'FIELDS' envelops a 'MODE' of the form 'ROWS of MODE2', may not occur in the sublanguage. The lower-bound of an actual-row-rower must be present. The number of different sizes of arithmetic values (including complex) and the number of different widths of bits and bytes values which are allowed is an implementation characteristic. Only a restricted number of the operators defined in the standard-prelude need be supplied for modes whose size or width is not zero. The optional local-symbol in a variable-declaration may not be omitted if the actual-declarer begins with a style-i-sub-symbol (/).

1.2. Clauses

Parallel-clauses, void-collateral-clauses and vacuums do not occur. Displays may not occur in certain balances (see 2.2) or as the first phrase inside a closed-clause. (Thus a collateral-clause cannot occur as the only phrase of a closed-clause.) Conformity-clauses do not occur.

1.3. Units

Only local-generators are allowed, and they may not occur as operands or ROWS-rowers, or occur in a local range before it has been possible to detect that it is local (Report 3.2.2.b) in a left-to-right scan. A jump to a label may not pass over a declaration in the same reach as that label. A go-to is obligatory in a jump, and procedure-yielding-MOID-jumps may not occur (and hence a jump may never yield a routine). Strings always have a lower bound of 1, and slices involving strings may not have revised-lower-bounds. The reference-to-STRING-PRIMARY of a slice is always dereferenced (thus transient names do not appear in the sublanguage). A void-denotation may not occur. The times-ten-to-the-power-symbol is excluded.

1.4. Coercions

A character may be widened to a string, and a string may be widened to a 'row of character' value. Rowing only changes a 'MODE' value into a 'row of MODE' value, and a 'ROWS of MODE' value into a 'row ROWS of MODE' value (and does not, for example, change a 'REF to MODE' value into a 'REF to row of MODE' value).

1.5. Independence and identification

A defining-indicator, except in a label-definition, must come textually before an applied-indicator which identifies it. An applied-mode-indication may not occur in the actual-ROWS-rower of the actual-declarer of its mode-definition. The priority-definition of a dyadic-operator must come before the first operation-definition of that operator, and the priority of an operator may not be redefined in an inner range. The test for independence of operation-definitions in the sublanguage is based upon a 'meekly related' condition. (For example, the modes 'procedure yielding MOID' and 'reference to MOID' are meekly related, though not firmly related.)

1.6. Symbols

A bold-TAG-token may not be defined as a mode-indication if that same token has been defined as an operator in an outer range, or vice versa. Only certain TAO-tokens may be defined as operators; these include all bold-TAG-tokens and a further set to be specified in the implementation characteristics (subject always to the syntax of the Report, 9.4.2.1.). Certain TAO-tokens are removed from the standard-prelude, so that the sets of TAO-tokens [e.g., IT, **7] used in the construction of extra versions of operators are mutually disjoint.

1.7. Transput

Channel and file enquiries (e.g. estab possible, Report 10.3.1.2.c, compressible, Report 10.3.1.3.e) are not permitted, and the conversion key of a file may not be altered. The only event routines provided are on logical file end, on physical file end, on page end and on line end. Files may be established, opened, scratched and closed, but not created or locked. All the layout routines, except backspace and set char number, are allowed. Conversion routines, formatless transput and binary transput are provided. Formatted transput is not provided.

2. Commentary

Most of the restrictions in the sublanguage are expressed by means of changes to the hyper-rules of the Report. The only nontrivial changes in the semantics of the Report are in those parts which apply exclusively to the sublanguage. Several indicators which are defined in the standard-prelude are not available in sublanguage-particular-programs, and a different particular-prelude is used.
2.1. The nest

The hyper-rules of Section 5 place additional information in the nest of a construct. This information is as follows:

a) 'sublanguage' (1.2.3.BC). This occurs in the nest of a particular-program if it is descended from the second alternative of hyper-rule 10.1.1.g. The predicate 'where NEST indicates sublanguage' (2.2.1.aa) examines the nest for 'sublanguage', and is used to select alternatives of those notions whose productions in the sublanguage are different from those in the full language. Since the nest of the standard-prelude does not contain 'sublanguage', it is possible for the external environment of sublanguage-particular-programs to be the same as that of full-language-particular-programs.

b) 'mark' (1.2.3.B). At any point in the production tree of a construct, the 'mark's contained in the 'LAYER's divide the 'PROPS' into those which would have been encountered at that point in a left-to-right scan, and those which would not have been encountered. The 'mark's are used to ensure that certain sublanguage restrictions are satisfied (5.2.3.1.aa, 7.1.1.ac, 7.2.1.ca). The 'mark' is moved through the 'LAYER' by hyper-rule 3.2.1.ba.

c) Layers and environs (1.2.3.B,BB). A 'LAYER' is added to the nest of each of those constructs which, when elaborated, will cause an environ to be established (Report 3.2.2.b). The 'ENVIRON' of a 'LAYER' is either 'nonlocal', when the corresponding environ is nonlocal, or it is 'local' or 'restricted', when it is local. If the 'ENVIRON' is 'restricted' then a local-generator cannot occur. The 'ENVIRON' is determined by the syntax (3.2.1.aa, 3.4.1.i, 3.5.1,a,b,e, 4.2.1.b, 5.2.3.1.a,b, 5.4.2.1.a,b, 5.4.3.1.a), and is used to restrict the occurrences of local-generators in sublanguage-particular-programs (5.2.3.1.aa).

2.2. The syntactic positions

Restrictions on the positions in which collateral-clauses may occur have been imposed by introducing a further sort, viz, 'robust' (1.2.2.CA), and by modifying the rules for balancing (3.2.1.f). The metanotion 'PRACETY' (3.1.1.AA) is used to cause the first phrase of a closed-clause, if that phrase is a unit, to be robust in sublanguage-particular-programs. Certain positions are required to be robust (3.2.1.d,f). Displays may not occur in robust positions (Report 3.3.1.d,e).

2.3. Independence of properties

Some of the tests for independence in sublanguage-particular-programs require that all the 'PROPS' in the nest are examined, and not simply those 'PROPS' in some 'LAYER'. They are collected together by hyper-rules 7.1.1.af,ag. The independence tests differ according to the defining-indicator:

a) A 'DYADIC TAD' (7.1.1.ab) is required to be unrelated to all the collected 'PROPS' in the nest. If the 'TAD' is not a 'TAB', then it is also required to be one of the acceptable 'TAD's for the implementation (7.1.1.aj).

b) A 'DUO TAD' (7.1.1.ac) is required to be unrelated to all 'DUO TAD's in the collected 'PROPS' of the nest (see (f)), and to be declared textually after a priority-definition of that 'TAD' (7.1.1.ac).

c) A 'MONO TAM' (7.1.1.ad) is required to be unrelated to all 'MONO TAM's in the collected 'PROPS' of the nest (see (f)), and its 'TAM' is required to be one of the acceptable 'TAM's for the implementation (7.1.1.ad).

d) A 'TALLY TAB' (7.1.1.ae) is tested for independence by examining the 'PROPS' of the 'LAYER'. The collected 'PROPS' of the nest are examined to ensure that its 'TAB' does not occur as a 'TAO' in an outer range (7.1.1.ai).

e) A 'MABEL TAG' (7.1.1.aa) has the same tests for independence as in the full language.

f) If the sublanguage is indicated in the nest, the "firmly related" test for the independence of operators is replaced by a "meekly related" test (7.1.1.na).

2.4. Identification

a) In the sublanguage, it is required that a 'QUALITY TAX' which is identified in the 'PROPSETY's of a 'LAYER' and which is not a 'label TAG' can be identified in those 'PROPSETY's which precede the 'mark' in that 'LAYER' (7.2.1.ca). If it is a 'label TAG', it may identify 'PROP's after the 'mark', provided no 'DEC's intervene.

b) In the nest of the constituent actual-ROWS-rover, if any, of the actual-declarer of a mode-definition, the 'mark' enveloped by the 'LAYER' of the defining range of the applied-mode-indication is moved to before the 'PROP' which envelops that 'TAB' (4.2.1.c). (This prevents that TAB-symbol occurring in the actual-ROWS-rover.)

2.5. Strings

The mode 'STRING' is 'structured with row of character letter aleph digit one mode'. If the nest indicates the sublanguage, the mode of a string-denotation is 'STRING', otherwise 'row of character'. Additional alternatives of
hyper-rule 5.3.2.1.a deal with slices of strings. A "specially related" test (7.1.l.nb) prevents the definition of operators which attempt to distinguish between the modes 'STRING' and 'row of character'.

2.6. Elidible hypernotions

Several of the hypernotions which have been introduced to define sublanguage-particular-programs have been made elidible, to reduce the number of trivial changes in the semantics.

2.7. The preludes

a) In order to prevent the identification of certain defining-indicators in the standard- prelude by applied-indicators in a sublanguage-particular-program, the mechanism employed in the Report for hiding indicators [Report 10.1.3] is extended to allow different parts of the standard-prelude to be identified from full-language-particular-programs and from sublanguage-particular-programs. 'subprelude' appears in the nest of the preludes if the particular-program is a sublanguage-particular-program (10.1.1.a,g). The number of forms made in the steps of the Report, 10.1.3, is limited in sublanguage programs to allow for the implementer to restrict the number of lengths of 'INTREAL' values and widths of 'BITS' values which may be manipulated.

b) The particular-prelude contained in a program when the second alternative of hyper-rule 10.1.1.g is used contains definitions of string and several other indicators required by sublanguage-particular-programs.

3. Modifications to the syntax of the Report

The modified and additional rules are listed below. It is to be understood that a rule preceded by a single upper or lower case letter replaces the corresponding rule of the Report, and that a rule labelled with two letters is additional to the rules of the Report. All other rules of the Report are unchanged.

Report 1.2.2. [[Changes caused by the additional syntactic position]]

C) SORT :: RONG ; firm ; meek ; weak ; soft.
CA) RONG :: strong ; robust.

Report 1.2.3. [[Changes caused by the restrictions on identification, positions where local-generators may occur and general sublanguage restrictions]]

B) LAYER :: NEW SUBLETY PROPSETY1 mark PROPSETY2.
BA) NEW :: new ENVIRON.
BB) ENVIRON :: local ; restricted ; nonlocal.
BC) SUBLETY :: sublanguage ; subprelude ; EMPTY.

Report 1.3.1. [[Additional general predicates]]

fa) where THING1 implies THING2 : where THING1, where THING2 ; unless THING1.
na) where (NOTETY1) is not (NOTETY2) : unless (NOTETY1) is (NOTETY2).

Report 1.3.2. [Changes caused by the new form of 'LAYER']

a) program : strong void new local mark closed clause [31a].
aa) WHETHER NEST indicates sublanguage [31a,32f,33a,c,d,46e,g,h,j,s,523a,b,532a,544a,65d,da,66a,71ab,ac,ad,ae,i,j,72ca,80a,812h,A341a] : WHETHER (NEST) contains (sublanguage).

Report 2.2.1. [Changes caused by the new form of 'LAYER']

a) SOlid NEST closed clause [22a,5D,551a,A341h,A349a] : SOlid NEST PRACETY serial clause defining LAYER [32a] PACK, where NEST indicates sublanguage [22aa] implies (PRACETY) is (prefaced).

AA) PRACETY :: prefaced ; EMPTY.

Report 3.1.1. [[Changes to make the first phrase of the serial-clause of a closed-clause, if a unit in the sublanguage, in a robust position]]

a) SOlid NEST closed clause [22a,5D,551a,A341h,A349a] : SOlid NEST PRACETY serial clause defining LAYER [32a] PACK, where NEST indicates sublanguage [22aa] implies (PRACETY) is (prefaced).
Report 3.2.1. ([Changes to determine 'ENVIRON', to move 'mark' through the 'LAYER', to make the first unit robust and to modify balancing])

a) SOID NESTI PRACETY serial clause defining new ENVIRON SUBLETY mark PROPSETY[31a,34f,l,35h] :
where ENVIRON determined by PROPSETY[aa],
SOID NESTI new ENVIRON SUBLETY mark PROPSETY PRACETY series(b).

aa) WHETHER ENVIRON determined by DECSETY LABSETY[a,aa,34c,523aa] :
where (DECSETY) is (EMPTY), WHETHER (ENVIRON) is (nonlocal) ;
where (DECSETY) is (DYADIC TAD DECSETY1),
WHETHER ENVIRON determined by DECSETY1 LABSETY[aa] ;
where (DECSETY) is (MOID TALLETY TAX DECSETY1),
WHETHER (ENVIRON) is (local).

b) SOID NEST2 PRACETY series[a,b,34c] :
strong void NEST2 PRACETY unit(d),
go on[94f] token, SOID NEST2 series(b) ;
where NEST2 advances past DECS2 to NEST3[ba],
NEST2 declaration of DECS2[41a],
go on[94f] token, SOID NEST3 series(b) ;
where NEST2 advances past LAB2 to NEST3[ba],
NEST3 label definition of LAB2[c], SOID NEST3 series(b) ;
where NEST2 advances past LAB2 to NEST3[ba],
where SOID NEST2 balances SOID1 and SOID2[e],
SOID1 NEST2 PRACETY unit[d], completion[94f] token,
NEST3 label definition of LAB2[c], SOID2 NEST3 series(b) ;
where (NEST2) is (NESTI NEW SUBLETY PROPSETY mark),
SOID2 NEST unit[d].

ba) WHETHER NEST LAYER1 advances past PROPS to NEST LAYER2[b,41a,b,c,42c] :
where (LAYER1) is
(NEW SUBLETY PROPSETY mark PROP Sety PROPS PROPSETY2),
WHETHER (LAYER2) is
(NEW SUBLETY PROPSETY1 PROPS mark PROPSETY2) ;
where (PROPS) is (FIELDS), WHETHER (LAYER2) is (LAYER1).

Report 3.3.1. ([Changes to restrict the occurrence of collateral-clauses])

a) strong void NEST collateral clause[5D,551a] :
unless NEST indicates sublanguage[22aa],
strong void NEST joined portrait[b] PACK.

b) SOID NEST joined portrait[a,b,c,d,34g] :
where SOID NEST balances SOID1 and SOID2[32e],
SOID1 NEST unit[32d], and also[94f] token,
SOID2 NEST unit[32d]
or alternatively SOID2 NEST joined portrait[b].

c) strong void NEST parallel clause[5D,551a] :
unless NEST indicates sublanguage[22aa], parallel[94f] token,
strong void NEST joined portrait[b] PACK.

d) strong ROWS of MODE NEST collateral clause[5D,551a] :
where (ROWS) is (row), strong MODE NEST joined portrait[b] PACK :
where (ROWS) is (row ROWS1),
strong ROWS1 of MODE NEST joined portrait[b] PACK :
unless NEST indicates sublanguage[22aa], EMPTY PACK.

e) WHETHER SORT MODI NEST balances
SORT1 MODI1 and SORT2 MODI2[b,33b,34d,h] :
WHERE SORT NEST balances SORT1 and SORT2[f]
and MODI balances MODI1 and MODI2[g].

f) WHETHER SORT NEST balances SORT1 and SORT2[e,522a] :
where (SORT) is (strong),
WHERE (SORT1 cum SORT2) is (strong cum strong) ;
unless (SORT) is (strong),
WHERE (SORT1 cum SORT2) is (RONG cum SORT) or (SORT1 cum SORT2) is (SORT cum RONG),
where NEST indicates sublanguage[22aa] implies (RONG) is (robust).

Report 3.4.1. ([Changes caused by the new form of 'LAYER'])

c) MODE NEST1 enquiry clause defining new ENVIRON mark DECSETY2[b,32i,35g] :
where ENVIRON determined by DECSETY2[32aa],
meek MODE NEST1 new ENVIRON mark DECSETY2 series[32b].
d) **SOID NEST2 alternate CHOICE STYLE clause**{b}:
SOID NEST2 in CHOICE STYLE clause{e},
where SOID NEST2 balances SOID1 and SOID2{32e},
SOID1 NEST2 in CHOICE STYLE clause{e},
SOID2 NEST2 out CHOICE STYLE clause{l}.

h) **SOID NEST2 in part of choice using UNITED{e,h}**:
SOID NEST2 case part of choice using UNITED[i],
where SOID NEST2 balances SOID1 and SOID2{32e},
SOID1 NEST2 case part of choice using UNITED[i], and also{94f} token,
SOID2 NEST2 in part of choice using UNITED[h].

i) **SOID NEST2 case part of choice using UNITED[h]**:
MOID NEST2 LAYER3 specification defining LAYER3{j,k,-},
where MOID unites to UNITED{64b},
where (LAYER3) is (new nonlocal mark DECSETY),
SOID NEST2 new nonlocal DECSETY mark unit{32d}.

j) **MODE NEST3 specification defining new nonlocal mark MODE TAG3{i}**:
NEST3 declarative defining new nonlocal MODE TAG3{541e} brief pack,
colon{94f} token.

k) **MOID NEST3 specification defining new nonlocal mark EMPTY[i]**:
formal MOID NEST3 declarer{46b} brief pack, colon{94f} token.

Report 3.5.1. ([Changes caused by the new form of 'LAYER' and the robust syntactic position])

a) **RONG void NEST1 loop clause**{5D,551a}:
NEST1 STYLE for part defining new nonlocal mark integral TAG2{b},
NEST1 STYLE intervals{c},
NEST1 STYLE repeating part with integral TAG2{e}.

b) **NEST1 STYLE for part defining new nonlocal mark integral TAG2{a}**:
STYLE for{94f} token,
integral NEST1 new nonlocal integral TAG2 mark
defining identifier with TAG2{48a} ;
where (TAG2) is (letter aleph), EMPTY.

c) **NEST1 STYLE repeating part with DEC2{a}**:
NEST1 new nonlocal DEC2 mark STYLE while do part{f} ;
NEST1 new nonlocal DEC2 mark STYLE do part{h}.

Report 4.1.1. ([Changes to move 'mark' through the 'LAYER'])

a) **NEST declaration of DECS{a,32b}**:
NEST COMMON declaration of DECS{42a,43a,44a,e,45a,-} ;
where (DECS) is (DECS1 DECS2),
where NEST advances past DECS1 to NEST1{32ba},
NEST COMMON declaration of DECS1{42a,43a,44a,e,45a,-},
and also{94f} token, NEST1 declaration of DECS2{a}.

b) **NEST COMMON joined definition of PROPS PROP**
{b,42a,43a,44a,e,45a,46e,541e} :
where NEST advances past PROPS to NEST1{32ba},
NEST COMMON joined definition of PROPS{b,c}, and also{94f} token,
NEST1 COMMON joined definition of PROP{c}.

c) **NEST COMMON joined definition of PROP**{b,42a,43a,44a,e,45a,46e,541e} :
where NEST advances past PROP to NEST1{32ba},
NEST1 COMMON definition of PROP{42b,43b,44c,f,45c,46f,541f,-}.

Report 4.2.1. ([Changes to prevent an applied-mode-indication occurring in the actual-declarer of its mode-definition])

AA) **NONROW :: PLAIN ; structured with FIELDS mode ; REF to MODE ; PROCEDURE ; UNITED.**

b) **NEST mode definition of MOID TALLY TAB{41c}**:
where (TAB) is (bold TAG) or (NEST) is (NEW mark LAYER),
MOID TALLY NEST defining mode indication with TAB{48a},
is defined as{94d} token,
actual MOID TALLY NEST new restricted mark declarer{c}.

c) **actual MOID TALLY1 NEST LAYER1 new restricted mark declarer{b}**:
where (TALLY1) is (i) and (MOID) is (ROWS of MODE),
actual ROWS NEST LAYER2 new restricted mark
rower{46i} STYLE bracket,
where NEST LAYER2 advances past MOID i TAB2
to NEST LAYER1{32ba},
actual MODE NEST LAYER1 new restricted mark declarer{46a} ;
where (TALLY1) is (i) and (MOID) is (NONROW),
actual NONROW NEST LAYER1 new restricted mark declarator{46c,d,g,h,o,s,-} ;
where (TALLY1) is (TALLY2 i),
MOID TALLY2 NEST LAYER1
new restricted mark applied mode indication with TAB2{48b}.
Report 4.6.1. (Changes to prevent ROWS1-of-MODE-declarators, flexible-ROWS-of-MODE-declarators, 'FIELDS' of the form 'ROWS of MODE' and UNITED-declarers, and to restrict the occurrence of local-generators)

B) VIRACT :: virtual ; actual ; actual STYLE.

e) VICTAL FIELDS NEST portrayer of FIELDS1[d,e]:
   VICTAL MODE NEST declarer[a,b],
   where NEST indicates sublanguage[22aa] implies
   (MODE) is (NONROW),
   NEST MODE FIELDS joined definition of FIELDS1[41b,c];
   where (FIELDS1) is (FIELDS2 FIELDS3),
   VICTAL MODE NEST declarer[a,b],
   where NEST indicates sublanguage[22aa] implies
   (MODE) is (NONROW),
   NEST MODE FIELDS joined definition of FIELDS2[41b,c],
   and also[94f] token, VICTAL FIELDS NEST portrayer of FIELDS3[e].

f) VIRACT flexible ROWS of MODE NEST declarator[a,42c]:
   unless NEST indicates sublanguage[22aa], flexible[94d] token,
   VIRACT ROWS of MODE NEST declarer[a].

h) VICTAL ROWS of MODE NEST declarator[a,b,42c]:
   where (VICTAL) is (virtual) or (VICTAL) is (actual)
   or (VICTAL) is (formal),
   where NEST indicates sublanguage[22aa] implies
   (MODE) is (NONROW),
   VICTAL ROWS NEST rower[i,j,k,l] STYLE bracket,
   VICTAL MODE NEST declarer[a,b];
   where (VICTAL) is (actual STYLE),
   where NEST indicates sublanguage[22aa] implies
   (MODE) is (NONROW),
   VICTAL ROWS NEST rower[i,j,k,l] STYLE bracket,
   actual MODE NEST declarer[a].

j) actual row NEST rower[h,i]:
   NEST lower bound[m], up to[94f] token, NEST upper bound[n];
   unless NEST indicates sublanguage[22aa], NEST upper bound[n].

s) VICTAL union of MOODS1 MOOD1 mode NEST declarator[a,b,42c]:
   unless NEST indicates sublanguage[22aa],
   unless EMPTY with MOODS1 MOOD1 incestuous[47f],
   union of[94d] token, MOODS NEST joined declarer[l,u] brief pack,
   where MOODS ravels to MOODS2[47g]
   and safe MOODS1 MOOD1 subset of safe MOODS2[73l]
   and safe MOODS2 subset of safe MOODS1 MOOD1[73l,m].

Report 4.8.1. (Changes caused by the identification conditions of the sublanguage requiring 'NEST' to be present in some hypernotions)

a) QUALITY NEST NEW SUBLETY
   PROPSETY1 QUALITY TAX mark PROPSETY2 defining INDICATOR
   with TAX[32c,35b,42b,43b,44c,f,45c,541f]:
   where QUALITY TAX NEST unrelated
   PROPSETY1 mark PROPSETY2[71aa,ab,ac,ad,ae],

b) QUALITY NEST applied INDICATOR with TAX[42c,46a,b,5D,542a,b,544a]:
   where QUALITY TAX NEST identified in NEST[72a],

c) MODE field PROPSETY1 MODE field TAG PROPSETY2
   defining field selector with TAG[46f]:
   where MODE field TAG NEW mark EMPTY independent
   PROPSETY1 PROPSETY2[71a,aa,ab],
   TAG[942A] token.

Report 5.2.3.1. (Changes to prevent the occurrence of heap-generators, and to restrict the occurrence of local-generators)

a) reference to MODE NEST2 LEAP generator[5C]:
   where NEST2 indicates sublanguage[22aa] implies
   LEAP NEST2 generatable[aa,-],
   LEAP[94d,-] token,
   actual MODE NEST2 new restricted mark declarer[46a].

aa) where local NEST1 new ENVIRON SUBLETY
   PROPSETY1 mark PROPSETY2 generatable[aa,aa]:
   where (ENVIRON) is (nonlocal), where NEST1 generatable[aa];
   where (ENVIRON) is (local),
   where local determined by PROPSETY1 PROPSETY2[32aa] implies
   local determined by PROPSETY1[32aa];
   where (ENVIRON) is (restricted), where false.

b) reference to MODINE NEST LEAP sample generator[44e]:
   where NEST indicates sublanguage[22aa] implies (LEAP) is (local),
   LEAP[94d,-] token,
   actual MODINE NEST new restricted mark declarer[44b,46a];
   where (LEAP) is (local),
   where NEST indicates sublanguage[22aa] implies
   (STYLE) is (brief),
   actual STYLE MODINE NEST new restricted mark declarer[44b,46a].
Report 5.2.4.1. [[Changes caused by the new strength of syntactic position]]

a) RONG reference to MODE.NEST nihil[5B] : nil[94f] token

Report 5.3.2.1. “Changes to allow slicing of strings in the sublanguage”

AA) STRING ::
   structured with row of character field letter aleph digit one mode.

a) REFETY MODE1 NEST slice[5D] :
   weak REFLEXETY ROWS1 of MODE1 NEST PRIMARY[5D],
   ROWS1 leaving EMPTY NEST indexer[b,c,-] STYLE bracket,
   where (REFETY) is derived from (REFLEXETY)[531b,c,,-] ;
   where (MODE1) is (ROWS2 of MODE2),
   weak REFLEXETY ROWS1 of MODE2 NEST PRIMARY[5D],
   ROWS1 leaving ROWS2 NEST indexer[b,d,-] STYLE bracket,
   where (REFETY) is derived from (REFLEXETY)[531b,c,,-] ;
   where NEST indicates sublanguage[22aa],
   where (REFETY MODE1) is (character),
   meek STRING NEST PRIMARY[5D],
   row leaving EMPTY NEST indexer[b,d,-] STYLE bracket ;
   where NEST indicates sublanguage[22aa],
   where (REFETY MODE1) is (STRING),
   meek STRING NEST PRIMARY[5D],
   row leaving row NEST strindexer[da] STYLE bracket.


fa) NEST strimmer[da] : NEST lower bound[46m] option,
   up to [94f] token, NEST upper bound[46n] option.

h) • trimscript : NEST subscript[e] ; NEST trimmer[f] ;
   NEST strimmer[fa] ; NEST revised lower bound[g] option.

i) • indexer : ROWS leaving ROWSETY NEST indexer[b,c,d] ;
   row leaving row NEST strindexer[da].

Report 5.4.1.1. [[Changes caused by the new form of 'LAYER']]

a) procedure yielding MOID NEST1 routine text[44d,5A] :
   formal MOID NEST1 declarer[46b], routine[94f] token,
   strong MOID NEST1 new local mark unit[32d].

b) procedure with PARAMETERS yielding MOID NEST routine text[44d,5A] :
   NEST1 new local mark DECS2 declarative
   defining new local mark DECS2[e] brief pack,
   where DECS2 like PARAMETERS[c,d,-],
   formal MOID NEST1 declarer[46b], routine[94f] token,
   strong MOID NEST1 new local DECS2 mark unit[32d].

e) NEST2 declarative defining NEW mark DECS2[b,e,34j] :
   formal MODE NEST2 declarer[46b],
   NEST2 MODE parameter joined definition of DECS2[41b,c] ;
   where (DECS2) is (DECS3 DECS4), formal MODE NEST2 declarer[46b],
   NEST2 MODE parameter joined definition of DECS3[41b,c],
   and also[94f] token,
   NEST2 declarative defining NEW mark DECS4[e].

Report 5.4.2.1. [[Changes caused by the new form of 'LAYER', and to restrict
the occurrence of local-generators]]

a) MOID NEST DYADIC formula[c,5B] :
   MODE1 NEST new restricted mark DYADIC TALLETY operand[c,,-],
   procedure with MODE1 parameter MODE2 parameter yielding MOID
   NEST applied operator with TAD[48b],
   where NEST DYADIC TAD identified in NEST[72a],
   MODE2 NEST new restricted mark DYADIC TALLY operand[c,-].

b) MOID NEST MONADIC formula[c,5B] :
   procedure with MODE parameter yielding MOID NEST
   applied operator with TAM[48b],
   MODE NEST new restricted mark MONADIC operand[c].

Report 5.4.3.1. [[Changes caused by the new form of 'LAYER']]

a) MOID NEST call[5D] :
   meek procedure with PARAMETERS yielding MOID NEST PRIMARY[5D],
   actual NEST new local mark PARAMETERS[b,c] brief pack.
Report 5.4.4.1. [[Changes caused by the new strength and to restrict the modes of jumps]]

a) RONG MOID NEST jump[5A] :
   where NEST indicates sublanguage[22aa],
   where (MOID) is (NONPROC)
   or (MOID) is (REF to procedure yielding MOID1),
goto[b], label NEST applied identifier with TAG[48b] ;
   unless NEST indicates sublanguage[22aa], goto[48b] option,
   label NEST applied identifier with TAG[48b].

Report 5.5.2.1. [[Changes caused by the new strength]]


Report 6.1.1. [[Changes caused by the new strength]]

a) RONG MOID FORM coercee[5A,B,C,D,A341i] :
   where (FORM) is (MORF), STRONG[A] MOID MORF ;
   where (FORM) is (COMORF), STRONG[A] MOID COMORF,
   unless (STRONG MOID) is (deprocedured to void).

Report 6.5.1. [[Changes to allow the widening of a 'STRING' value to a 'row of character' value]]

CA) COERCEE : selection ; slice ; routine text ; ADIC formula ; call ;
   applied identifier with TAG ; assignment ; identity relation ;
   LEAP generator ; cast ; denoter ; format text.

d) widened to row of character NEST COERCEE[61A] :
   MEEK[61C] BYTES NEST COERCEE ;
   where NEST indicates sublanguage[22aa],
   MEEK[61C] STRING NEST COERCEE.
da) widened to STRING NEST COERCEE[61A] :
   where NEST indicates sublanguage[22aa],
   MEEK[61C] character NEST COERCEE.

Report 6.6.1. [[Changes to restrict the forms of rowing]]

a) rowed to REFETY ROWS1 of MODE FORM[61A] :
   where NEST indicates sublanguage[22aa] implies (REFETY) is (EMPTY),
   where (ROWS1) is (row), STRONG[61A] REFLEXETY MODE FORM,
   where (REFETY) is derived from (REFLEXETY)[531b,c,-] ;
   where NEST indicates sublanguage[22aa] implies (REFETY) is (EMPTY),
   where (ROWS1) is (row ROWS2),
   STRONG[61A] REFLEXETY ROWS2 of MODE FORM,
   where (REFETY) is derived from (REFLEXETY)[531b,c,-].

Report 7.1.1. [[Changes to the tests for independence, etc., in the sublanguage]]

C) PREFSETY :: PREF PREFSETY ; EMPTY.
CA) MABEL :: MODE ; label.
CB) PRADIC :: PRAM ; DYADIC.
CC) FEEKLY :: firmly ; meekly.
CD) A metaproduction rule is to be added for the metanotation "SUBTAO" (for which no metaproduction rule is given in this Report), each of whose hypernotions is some 'DYAD BECOMESETY' or 'DYAD cum NOMAD BECOMESETY'.

a) WHETHER PROPI NEST independent PROPS2 PROP2 {a,c,72a,aa,ab,ac,ad,ae} :
   WHETHER PROPI NEST independent PROPS2 [a,c]
   and PROPI NEST independent PROP2 [c].
aa) WHETHER MABEL TAG NEST unrelated
   PROPSETY2 mark PROPSETY3 [48a,c] :
   WHETHER MABEL TAG NEST independent
   PROPSETY2 PROPSETY3 [a,b,c].
ab) WHETHER DYADIC TAD NEST unrelated
   PROPSETY2 mark PROPSETY3 [48a,c] :
   where NEST indicates sublanguage[22aa],
   where PROPSETY1 collected properties from NEST[af,ag],
   WHETHER DYADIC TAD NEST independent
   PROPSETY1 PROPSETY2 PROPSETY3 [a,b,c]
   and TAD NEST acceptable caption [aj] ;
   unless NEST indicates sublanguage[22aa],
   WHETHER DYADIC TAD NEST independent
   PROPSETY2 PROPSETY3 [a,b,c].
WHETHER DUO TAD NEST unrelated PROPSETY2 mark PROPSETY3{48a}:
where NEST indicates sublanguage{22aa},
where PROPSETY1 collected properties from NEST[af,ag],
where DUO TAD NEST independent
PROPSETY1 PROPSETY2 PROPSETY3{a,b,c}
and DYADIC TAD contained in PROPSETY1 PROPSETY2;
unless NEST indicates sublanguage{22aa},
WHETHER DUO TAD NEST independent
PROPSETY2 PROPSETY3{a,b,c}.

WHETHER MONO TAM NEST unrelated PROPSETY2 mark PROPSETY3{48a}:
where NEST indicates sublanguage{22aa},
where PROPSETY1 collected properties from NEST[af,ag],
WHETHER MONO TAM NEST independent
PROPSETY1 PROPSETY2 PROPSETY3{a,b,c}
and TAM NEST acceptable caption{aj}:
unless NEST indicates sublanguage{22aa},
WHETHER MONO TAM NEST independent
PROPSETY2 PROPSETY3{a,b,c}.

WHETHER MOI D TALLY TAB NEST unrelated PROPSETY2 mark PROPSETY3{48a}:
where NEST indicates sublanguage{22aa},
where PROPSETY1 collected properties from NEST[af,ag],
WHETHER MOI D TALLY TAB NEST independent
PROPSETY2 PROPSETY3{a,b,c}
and TAB not operator in PROPSETY1[ah,ai];
unless NEST indicates sublanguage{22aa},
WHETHER MOI D TALLY TAB NEST independent
PROPSETY2 PROPSETY3{a,b,c}.

WHETHER EMPTY collected properties
from NEW SUBLETY mark{ab,ac,ad,ae,ag}:
WHETHER true.

WHETHER PROPSETY1 PROPSETY2 collected properties
from NEST NEW SUBLETY PROPSETY2 mark PROPSETY3
{ab,ac,ad,ae,ag}:
WHETHER PROPSETY1 collected properties from NEST[af,ag].

WHETHER TAB not operator in EMPTY[ae,ai]: WHETHER true.

WHETHER TAB not operator in PROPSETY PROP[ae,ai]
where (PROP) is (PRADIC TAB), WHETHER false;
where (PROP) is (MOI D TALLY TAB), WHETHER true
unless (PROP) contains (TAB),
WHETHER TAB not operator in PROPSETY[ah,ai].
Report 7.2.1. (Changes caused by the new form of 'LAYER')

a) WHETHER PROP NEST1 identified in
NEST2 NEW SUBLETY PROPSETY1 mark PROPSETY2 [a,48b,542a] :
where PROP resides in PROPSETY1 PROPSETY2 [b,c,,-],
WHETHER PROP NEST1 acceptable from
NEST2 NEW SUBLETY PROPSETY1 mark PROPSETY2 [ca] :
where PROP NEST1 independent PROPSETY1 PROPSETY2 [71a,b,c],
WHETHER PROP NEST1 identified in NEST2 [a,-].

c) WHETHER QUALITY TAX NEST acceptable from
NEST2 NEW SUBLETY PROPSETY1 mark PROPSETY2 [a] :
where NEST indicates sublanguage [22aa], unless (QUALITY) is (label),
WHETHER QUALITY TAX resides in PROPSETY1 [b,c,-] ;
where NEST indicates sublanguage [22aa] and (QUALITY) is (label),
WHETHER (PROPSETY2) is (LABSETY2) ;
unless NEST indicates sublanguage [22aa], WHETHER true.

Report 8.0.1. (Changes caused by the restrictions on the number of sizes of 'INTREAL' values and widths of 'BITS' values)

AA) SUBMODE :: SUBINTREAL ; SUBBITS ; character ; boolean ; STRING.
AB) A metaproduction rule is to be added for the metanotion "SUBINTREAL" ("SUBBITS") (for which no metaproduction rule is given in this report), each of whose hypernotions is some 'INTREAL' ('BITS').

c) MOID NEST denoter {5D,A341i} :
where NEST indicates sublanguage [22aa] implies (MOID) is (SUBMODE),
fragment [92a] sequence option,
MOID denotation [810a,811a,812a,813a,814a,815a,82a,b,c,83a,aa,-] ;
unless NEST indicates sublanguage [22aa] or (MOID) is (STRING),
fragment [92a] sequence option,
MOID denotation [810a,811a,812a,813a,814a,815a,82a,b,c,83a,,-] ;
or MOID NEST denotation [810a,812a].

Report 8.1.0.1. (Changes caused by the exclusion of the times-ten-to-the-power-symbol)

A) NESTETY :: NEST ; EMPTY.

c) SIZE INTREAL NESTETY denotation [a,80a] :
SIZE symbol [94d], INTREAL NESTETY denotation [a,811a,812a].
b) plain denotation : PLAIN NESTETY denotation [a,811a,812a,813a,814a,815a].

Report 8.1.2.1. (Changes caused by the exclusion of the times-ten-to-the-power-symbol)

a) real NEST denotation [80a,810a] :
variable point numeral [b] ; NEST floating point numeral [e].
e) NEST floating point numeral [a] :
stagnant part [f], NEST exponent part [g].
g) NEST exponent part [e] :
NEST times ten to the power choice [h], power of ten [i].
h) NEST times ten to the power choice [g] :
unless NEST indicates sublanguage [22aa],
times ten to the power symbol [94b] ;
letter e symbol [94a].

Report 8.3.1. (Changes caused by the mode 'STRING')

aa) STRING denotation [80a,83c] :

bc) string denotation :
row of character denotation [a] ; STRING denotation [aa].
Report 10.1.1. ([Changes to introduce 'subprelude' and 'sublanguage' into
the nest])

a) program text :
  STYLE begin [94f] token, LAYER new local mark DECSI preludes [b],
  parallel [94f] token,
  LAYER new local DECSI mark tasks [d] PACK, STYLE end [94f] token,
  where (LAYER) is (new local EMPTY mark)
  or (LAYER) is (new subprelude local EMPTY mark).

b) LAYER new local mark DECS preludes [a] :
  where (DECS) is (DECSI DECSY2 DECSY3),
  LAYER new local mark DECSI DECSY2 DECSY3 standard prelude
  with DECSI [c],
  LAYER new local DECSI mark DECSY2 DECSY3
  library prelude with DECSY2 [c],
  LAYER new local DECSI DECSY2 mark DECSY3
  system prelude with DECSY3 [c].

f) NEST1 user task [d] :
  NEST1 new local mark DECS STOP particular prelude with DECS [c],
  NEST1 new local DECS mark STOP particular program [g] PACK,
  go on [94f] token,
  NEST1 new local DECS mark STOP particular postlude [i].

g) NEST2 particular program [f] :
  where (NEST2) contains (subprelude),
  NEST2 sublanguage particular program [ga] ;
  unless (NEST2) contains (subprelude),
  NEST2 full language particular program [gb].

ga) NEST2 sublanguage particular program [g] :
  NEST2 new sublanguage local mark LABSY3
  joined label definition of LABSY3 [b],
  strong void NEST2 new sublanguage local LABSY3 mark
  ENCLOSED clause [31a,33a,c,34a,35a].

gb) NEST2 full language particular program [g] :
  NEST2 new local mark LABSY3
  joined label definition of LABSY3 [b],
  strong void NEST2 new local LABSY3 mark
  ENCLOSED clause [31a,33a,c,34a,35a].

4. Modifications to the semantics of the Report

Each of the subsections in this section specifies a change required in the
semantics of the Report. It is to be understood that the text between # and => is
to be replaced, in the Report, by that between => and #.

a) ([Additional elidible hypernotions are required. The following change is to
be made to the list in the Report, 1.1.4.2.c;])
  # "defining LAYER".
  "defining LAYER" • "ENVIRON" • "mark" • "SUBLETY". #

b) ([The PRIMARY of a slice may be a 'STRING' value, and a change is
required in 5.3.2.2.a and b. In section a, the following change is required:])
  # the value referred to by V; =>
  the value referred to by V or of the value selected in V by 'letter aleph digit
  one'; #

  [[Also in section a, the following bullet is to be replaced:]]
  # W is the value ... be replaced:]]
  # W is the value ... by (11, ..., In). =>
  If the mode of V is 'STRING',
  then W is the value selected by (11, ..., In) in [2.1.3.4.a,i] the field of V
  selected by (2.1.3.3.a) 'letter aleph digit one';
  otherwise, W is the value selected in [2.1.3.4.a,g,i] or the name generated
  from [2.1.3.4.j] V by (11, ..., In). #

  [[In section b the following change is required:]]
  # Case B: the i-th trimscript is a trimmer T: =>
  Case B: the i-th trimscript is a trimmer or a strimmer T: #

c) ([An additional case is required for the widening of a 'STRING' value into a
'row of character' value. This is added to the end of section 6.5.2.])
  #. =>;
  Case D: 'MODE' is 'STRING':
  W is the structured value whose [only] field is a multiple value
  composed of a descriptor ((1, 1)) and V. #

d) ([A string-denotation is either a STRING-denotation or it is a row-of-
character-denotation. Modifications are required to the semantics of
section 8.3.2. The following replacement is required:]]
  # The yield of a string-denotation D =>
  a) The yield of a row-of-character-denotation D #

  [[An additional subsection deals with STRING-denotations:]]
  # =>
  b) The yield of a STRING-denotation D is a structured value whose [only]
  field is the yield of a row-of-character-denotation akin [1.1.3.2.k] to D. #

Report 10.3.4.1.1. ([Change caused by the exclusion of formatted transput])

a) FORMAT NEST format text [5D]:
  unless NEST indicates sublanguage [22a], formatter [94f] token,
  NEST collection [b] list, formatter [94f] token.
e) [In order to hide certain indicators in the EXTERNAL- preludes in sublanguage-particular-programs, the mark + is introduced. This is either replaced by ? or it is erased, according to whether the protonotion 'subprelude' is contained in the nest [3.0.2] of the EXTERNAL-prelude. The first change, in section 10.1.3, reintroduces 'NEST', which had been elided:]

# A representation ... following steps: =>

A representation of a NEST-EXTERNAL-prelude, NEST-system-task or NEST-particular-postlude is obtained by altering each form in the relevant section of this chapter in the following steps: #

(An additional step, inserted between steps 1 and 2 of section 10.1.3 causes the mark + to be replaced or removed:)

Step 1.5: If, in some form, as possibly made in the step above, the mark + occurs, then, if the predicate 'where (NEST) contains (subprelude)' holds, the mark + is replaced by the mark ?: otherwise, (the predicate does not hold and) the mark + is deleted; Step 1.5 is then taken again; #

(Throughout the steps the term "sufficient number" is modified so that in the sublanguage the implementer is not required to have declarers for all sizes and widths of values, nor even all the operators, etc., involving those which he does allow. The following change is to be made twice in Step 4 and twice in Step 5:))

# sufficient number => some number #

(The following paragraph is to be inserted after Step 9, to specify the number of forms which are to be made:)

# =>

If the predicate 'where (NEST) contains (subprelude)' holds, then the number of new forms made in steps 4 and 5 above is a characteristic of the implementation, and must be at least one for all forms except those derived from 10.2.3.3.q, 10.2.3.4.n, 10.2.3.7.n, 10.2.3.8.n, and 10.2.3.9.d (leng), and 10.2.3.3.r, 10.2.3.4.o, 10.2.3.7.o, 10.2.3.8.o, and 10.2.3.9.e (shorten) (for which no replacements need be supplied); otherwise, (the predicate does not hold and) the number of new forms is some sufficient number. #

f) [Changes are required in section 10.5.1 to specify that different particular-preludes are incorporated into a program for a sublanguage-particular-program and for a full-language-particular-program. This is done by specifying a different "base set" of definitions for each. The changes are as follows:]

# The representation of the particular-prelude ... following forms, =>

aa) The representation of the NEST-particular-prelude of each user-task is obtained from the "basis" forms (ab), #

b) The "basis" forms of a NEST-particular-prelude are obtained as follows:

If the predicate 'unless (NEST) contains (sublanguage)' holds, then the forms are those given in 10.5.1.a,b,c,d,e,f,g,h,i,j only; otherwise, they are all the forms in section 10.5.1. #

5. Modifications to the standard preludes of the Report

This section gives the changes required to the forms in the standard environment of the Report [10.2, 10.3, 10.4, 10.5]. In each subsection the change which is made is the insertion of the mark + between mode (the mode-symbol), op (the operator-symbol), proc (the procedure-symbol) or the representation of some declarer, and the indicator which follows it. In order to abbreviate this section, only the reference within the Report, the relevant indicator and an indication of the reason for the change are given.

(Thus, since the list includes, under 10.2.2., i) (D) string the form in 10.2.2.i is to be altered to mode+: string = flex [1:0] char; )

(The letter in braces indicates the reason for the change, as follows:

A - The indicator is not available in sublanguage-particular-programs.
B - The presence of the declarer string in the form necessitates that a copy of the form is included in each sublanguage-particular-prelude in order that it may identify the revised mode-declaration for string in that particular-prelude. No language restriction is implied by this change.
C - The form defines a group of operators (e.g. #t, **, up) one of which is, in some other form, associated with some different group (e.g. #t, up, shl). A copy of the form, without that operator, is included in the particular-prelude.
D - A modified copy of the form is present in the particular-prelude.)

a) 10.2.1.

a) (A) int lengths b) (A) int shorths d) (A) real lengths
e) (A) real shorths h) (A) bits lengths i) (A) bits lengths
j) (A) L bits width k) (A) bytes lengths l) (A) bytes shorths
m) (A) L bytes width q) (A) null character r) (A) flip
s) (A) flop t) (A) error char u) (A) blank

b) 10.2.2.
i) (D) string
c) 10.2.3.0.
   a) {C} up, down, L, r

   d) 10.2.3.1.
      b) {C} \texttt{\textit{thwb}, L \dagger}  
      c) {C} \texttt{\textit{upb, r \dagger}}  
      d) {C} \texttt{\textit{thwb, L \dagger}}

   e) 10.2.3.3.
      p) {C} \texttt{\textit{t}, \ast, up \dagger}

   f) 10.2.3.4.
      r) {C} \texttt{\textit{sentler, r \dagger}}

   g) 10.2.3.5.
      g) {C} \texttt{\textit{t}, \ast, up \dagger}

   h) 10.2.3.6.
      b) {B} +

   i) 10.2.3.7.
      t) {C} \texttt{\textit{t}, \ast, up \dagger}

   j) 10.2.3.8.
      g) {C} \texttt{\textit{t}, up, shl \dagger}  
      h) {C} \texttt{\textit{t}, down, shr \dagger}

   k) 10.2.3.9.
      c) {B} \texttt{\textit{L bytes pack}}

   l) 10.2.3.10.
      a) {B} \texttt{\textit{<, ll \dagger}}
      b) {B} \texttt{\textit{<, le \dagger}}
      c) {B} \texttt{\textit{==, eq \dagger}}
      d) {B} \texttt{\textit{>, ne \dagger}}
      e) {B} \texttt{\textit{>=, ge \dagger}}
      f) {B} \texttt{\textit{>, gt \dagger}}
      g) {B} \texttt{\textit{R}}
      h) {B} \texttt{\textit{R}}
      i) {B} +
      j) {B} +
      k) {B} +
      l) {B} \texttt{\textit{bx, x \dagger}}
      m) {B} \texttt{\textit{bx, bx \dagger}}
      n) {B} \texttt{\textit{bx, bx \dagger}}
      o) {B} \texttt{\textit{bx, bx \dagger}}

   m) 10.2.3.11.
      q) {B} \texttt{\textit{plussb, += \dagger}}  
      r) {B} \texttt{\textit{plusto, += \dagger}}
      s) {B} \texttt{\textit{plussb, += \dagger}}  
      t) {B} \texttt{\textit{plusto, += \dagger}}  
      u) {B} \texttt{\textit{timesab, x::, e:: \dagger}}

   n) 10.2.4.
      a) {A} \texttt{\textit{sena}}
      b) {A} \texttt{\textit{level}}
      c) {A} \texttt{\textit{level}}
      d) {A} \texttt{\textit{down}}
      e) {A} \texttt{\textit{up}}

   o) 10.3.1.2.
      c) {A} \texttt{\textit{estab possible}}
      d) {A} \texttt{\textit{stand conv}}

   p) 10.3.1.3.
      b) {A} \texttt{\textit{get possible}}
      c) {A} \texttt{\textit{put possible}}
      d) {A} \texttt{\textit{bin possible}}
      e) {A} \texttt{\textit{compressible}}
      f) {A} \texttt{\textit{reset possible}}
      g) {A} \texttt{\textit{set possible}}
      h) {A} \texttt{\textit{reid possible}}
      j) {A} \texttt{\textit{make conv}}
      k) {B} \texttt{\textit{make term}}
      p) {A} \texttt{\textit{on format end}}
      q) {A} \texttt{\textit{on value error}}
      r) {A} \texttt{\textit{on char error}}
      s) {A} \texttt{\textit{reidf}}

   q) 10.3.1.4.
      b) {B} \texttt{\textit{establish}}
      c) {A} \texttt{\textit{create}}
      d) {B} \texttt{\textit{open}}

   o) {A} \texttt{\textit{lock}}
   p) {A} \texttt{\textit{scratch}}

   r) 10.3.1.6.
      b) {A} \texttt{\textit{backspace}}
      k) {A} \texttt{\textit{set char number}}

   s) 10.3.2.1.
      b) {B} \texttt{\textit{whole}}
      c) {B} \texttt{\textit{fixed}}
      d) {B} \texttt{\textit{float}}
      l) {A} \texttt{\textit{char in string}}
      m) {A} \texttt{\textit{Lint width}}
      n) {A} \texttt{\textit{L real width}}
      o) {A} \texttt{\textit{L exp width}}

   t) 10.3.5.
      a) {A} \texttt{\textit{format}}

   u) 10.3.5.1.
      a) {A} \texttt{\textit{putf}}
6. The sublanguage particular prelude

The forms in this section are to be added to those of the particular- prelude of the Report [10.5.1]. In addition, a sufficient number of other forms, not listed below, which are copies of forms in sections 10.2 and 10.3 of the Report, are also to be included in order that every applied-indicator in the forms in the particular- prelude and particular-postlude may identify a defining-indicator in the EXTERNAL-preludes. (These forms will all include the mark a. before the defining-indicator.) The list is in two parts. Part a contains those forms which are identical to the forms in the Report (and which are indicated by B in section 4); only the reference to the Report of the corresponding form is given. Part b contains the definitions of operators (which are indicated by C in section 4) and of the mode-indication string. In these forms, a routine-text may have been replaced by a pseudo-comment. It is to be understood that the pseudo-comment is to be replaced by the routine-text of the form whose (Report) reference is given in the pseudo-comment.

a) Forms which are copies of forms in the standard- prelude of the Report (as indicated by B in section 4 above)

10.2.3.6.b
10.2.3.9.c
10.2.3.10.a, b, c, d, e, f, g, h, i, j, k, l, m, n, o
10.2.3.11.q, r, s, t, u
10.3.3.3.k
10.3.1.4.b, d
10.3.2.1.b, c, d

b) Additional forms (as indicated by C and D in section 4 above)

a) mode string = struct:flex [1:0] char Fl;
b) op lwb = c 10.2.3.1.b c;
c) op upb = c 10.2.3.1.c c;
d) op upb = (string a) Int : upb Fl of a;
e) op lwb = c 10.2.3.1.d c;
f) op upb = c 10.2.3.1.e c;
g) op lwb = c 10.2.3.1.f c;
h) op entler = c 10.2.3.1.g c;
i) op upb = (string a) Int : upb Fl of a;
j) op lwb = c 10.2.3.1.h c;
k) op upb = (string a) Int : upb Fl of a;
l) op lwb = c 10.2.3.1.i c;
m) op upb = (string a) Int : upb Fl of a;

7. Acknowledgements

The assistance of the members of the Subcommittee for ALGOL 68 support, especially C. H. Lindsey and P. Knueven, and the comments and criticisms of the members of the full Working Group are gratefully acknowledged.

8. References


