CAL 6000 SNØBØL IS AVAILABLE UNDER BOTH BATCH AND INTERACTIVE PROCESSING. THIS DOCUMENT DESCRIBES CAL SNØBØL WITH RESPECT TO STANDARD SNØBØL4 AND ALSO DETAILS THE UNIVERSITY OF MINNESOTA VERSION OF CAL SNØBØL.
INTRODUCTION

This document contains information on the University of Minnesota version of the CAL 6000 SNOBOL interpreter written at the University of California, Berkeley. Also discussed are interactive SNOBOL facilities at the University of Minnesota. Please refer to the document "SNOBOL4 AT THE UNIVERSITY OF MINNESOTA" for a description of all SNOBOL facilities available and for specific information pertaining to running programs in a batch processing environment. Also included there is a complete list of references and a brief description of the SNOBOL4 language. Copies of the document above as well as this CAL SNOBOL description are available from the UCC Reference Librarian.

CAL SNOBOL is more or less a subset of standard SNOBOL4 as defined in the book:
"THE SNOBOL4 PROGRAMMING LANGUAGE", BY GRISWOLD, POAGE, AND POLANSKY, PRENTICE HALL, 1971 (SECOND EDITION)
(A GOOD BOOK FOR BEGINNING SNOBOLERS IS: "A SNOBOL4 PRIMER", GRISWOLD AND GRISWOLD, PRENTICE HALL, 1973)

The first book is an excellent reference and tutorial manual describing the macro implementation of SNOBOL4 for the IBM 360 computer. CAL SNOBOL is not a macro implementation, but instead a version tailored for efficiency on the CDC 6000/CYBER 70 series machines. It consists of a translator to convert SNOBOL source statements into an internal micro code, and an interpreter to execute the micro code. CAL SNOBOL was developed beginning in 1968-69 at Berkeley and has been subsequently maintained and modified there. The U of M version has been extended for the timesharing environment, allowing line numbers on program files and enabling interactive input/output, and has been modified for the Kronos 2.1 operating system. The section "U OF M MODIFICATIONS TO CAL SNOBOL" contains a summary of changes.

Following this introduction is a description of timesharing information for running CAL SNOBOL programs, and a section on running programs under the batch subsystem. Further sections describing differences in CAL SNOBOL over standard SNOBOL are: character sets, datatypes, operators, arithmetic, primitive functions, input/output functions, and keywords. Also included are lists of U of M modifications to CAL SNOBOL, known bugs, and SNOBOL simulated functions. An index is provided at the end of this document.

CAL SNOBOL has been available under the Meritss Timesharing System on the CDC 6400 at the University of Minnesota since 1972. In October, 1974 CAL SNOBOL's timesharing capabilities on the Cyber 74 machine at the University Computer Center were utilized under the MIRJE system. CAL SNOBOL is fully supported by UCC.
TIMESHARING INFORMATION

THE TIMESHARING USER HAS THE CHOICE OF RUNNING CAL SNOBOL PROGRAMS UNDER THE "SNOBOL SUBSYSTEM" OR THE "BATCH SUBSYSTEM" SUPERVISED BY TELEX - THE NAME OF THAT PART OF THE KRONOS OPERATING SYSTEM HANDLING TIMESHARING JOBS. TIMESHARING JOBS ARE THEREFORE KNOWN AS "TELEX ORIGIN" OR "RUN UNDER TELEX". CAL SNOBOL MAY ALTERNATIVELY BE RUN (USUALLY WITH PUNCHED CARDS) FROM REMOTE BATCH TERMINALS. THESE JOBS ARE CONSIDERED "BATCH ORIGIN" TO KRONOS. THE DIFFERENCES FOR SNOBOL USERS BETWEEN THE TWO TYPES ARE THE KIND OF PRINTED OUTPUT (PRINTER WIDTH) THAT IS RECEIVED, AND THE CHARACTER SET USED (TIMESHARING TERMINAL AS OPPOSED TO LINE PRINTER CHARACTERS).

CAL SNOBOL IS EASIEST TO USE UNDER THE SNOBOL SUBSYSTEM. AFTER LOGGING IN TO THE TIMESHARING SYSTEM, THE USER CAN ACCESS CAL SNOBOL BY TYPING:

```
SNOBOL XXXX
```

WHERE "XXXX" IS ONE OR MORE ADDITIONAL TELEX COMMANDS. IF "XXXX" IS NULL AND NO PRIMARY FILE IS CURRENTLY SPECIFIED, THE SYSTEM RESPONDS:

```
OLD, NEW, OR LIB FILE:
```

THE USER MAY THEN SPECIFY ONE OF THE THREE FILE TYPES AND A PRIMARY FILE NAME. THE ADDITIONAL COMMANDS CAN BE ANY LOGICAL SEQUENCE OF TIMESHARING COMMANDS THAT WOULD NORMALLY FOLLOW THE ABOVE COMMAND. EXAMPLES:

```
SNOBOL OLD A - BRINGS UP THE SNOBOL SUBSYSTEM AND SELECTS THE PERMANENT FILE A AS THE PRIMARY FILE.
SNOBOL RUN I=B - BRINGS UP THE SNOBOL SUBSYSTEM AND EXECUTES THE SECONDARY LOCAL FILE B.
SNOBOL STATUS - BRINGS UP THE SNOBOL SUBSYSTEM AND EXECUTES THE STATUS COMMAND.
```

ASSUMING THAT A NEW PROGRAM (MYPROG) IS TO BE RUN, THE COMMAND:

```
SNOBOL NEW MYPROG
```

MAY BE ENTERED SELECTING A NEW PRIMARY FILE. MYPROG. THEN THE AUTO COMMAND MAY BE USED TO ENTER THE TEXT FOR THE PROGRAM LINE BY LINE. LEADING LINE NUMBERS AND A BLANK SPACE ARE SUPPLIED FOR EACH LINE UNDER AUTO MODE. THE FIRST POSITION AFTER THE LINE NUMBER AND THE BLANK IS CONSIDERED COLUMN 1 FOR SNOBOL. THE SNOBOL STATEMENT LABEL FIELD BEGINS HERE AND USE OF THE CONTINUATION CHARACTERS + OR - MUST ALSO BE HERE. AN ASTERISK HERE DENOTES A COMMENT LINE. COLUMN 1 MUST BE LEFT BLANK IF THE SNOBOL STATEMENT DOES NOT CONTAIN A LABEL. AUTO MODE MAY BE TERMINATED VIA THE ESCAPE KEY. TO MAKE CORRECTIONS, ONE ONLY NEED TO RETYPE THE LINE NUMBER AND LEADING BLANK OF THE LINE TO BE CHANGED AND THEN CONTINUE WITH THE NEW LINE.

WHEN A CAL SNOBOL PROGRAM IS EXECUTED, THE LINE NUMBERS AND ONE
LEADING CHARACTER (USUALLY A BLANK, AS FROM "AUTO" MODE) ARE SKIPPED BY THE SNOBOL INTERPRETER IF THEY EXIST ON THE PROGRAM FILE, AND THE REST OF THE LINE IS CONSIDERED THE SNOBOL STATEMENT. THE MAXIMUM LINE LENGTH FROM A TIMESHIRING TERMINAL IS 150 CHARACTERS, WHILE THE SNOBOL INTERPRETER WILL ACCEPT ANY LENGTH (BUT IF THERE ARE NO LINE NUMBERS ON THE FILE, CAL SNOBOL WILL RECOGNIZE ONLY THE FIRST 72 CHARACTERS). AFTER EXITING AUTO MODE, A PROGRAM MAY BE LISTED WITH THE TELEX "LIST" OR "LNH" COMMAND. THE USER MAY START EXECUTION OF A CAL SNOBOL PROGRAM BY ENTERING THE "RUN" OR "RNH" COMMAND. LEGAL PARAMETERS FOR THE RUN COMMAND UNDER THE SNOBOL SUBSYSTEM ARE:

I=FILE - WHERE FILE IS A SECONDARY FILE TO BE EXECUTED. A PRIMARY FILE TO BE EXECUTED NEED NOT BE SPECIFIED.
MA-NNNNN - OR MI-NNNNN, TO REQUEST ADDITIONAL FIELD LENGTH. MA - USE NNNNN AS THE FIELD LENGTH (FL) FOR THE JOB (NNNNN IS IN OCTAL, UNLESS SUFFIXED WITH A "D"); MI - ADD NNNNN TO THE SYSTEM CALCULATED FL; OTHERWISE IT IS THE SAME AS MA.
X - SPECIFIES THE ALTERNATE CHARACTER SET FOR TIMESHIRING BASED ON CDC 6000/CYBER 70 SERIES LINE PRINTER EQUIVALENTS. SEE THE SECTION ON "CHARACTER SETS".

EXPLANATIONS OF COMPILE TIME ERRORS INCLUDE THE LINE NUMBER OF THE OFFENDING STATEMENT, AND A BRIEF DIAGNOSTIC OF ONE OR MORE ERRORS ON THAT LINE. EXPLANATIONS OF EXECUTION TIME ERRORS APPEAR AS FOLLOWS:

- ERROR TERMINATION IN STATEMENT N AT LEVEL M
< EXECUTION ERROR MESSAGE >

STATEMENT N IS NOT THE LINE NUMBER, BUT THE SNOBOL STATEMENT AT WHICH THE ERROR WAS DETECTED (COUNNT STATEMENTS BEGINNING WITH THE FIRST STATEMENT, NOT INCLUDING COMMENTS AND CONTINUATIONS.) LEVEL M IS THE FUNCTION LEVEL AT WHICH THE ERROR OCCURRED. THE MAIN PROGRAM IS LEVEL 0. THE EXECUTION ERROR MESSAGE DESCRIBES THE TYPE OF FATAL ERROR.

AN EXAMPLE EXISTS ON THE LIBRARY FILE SNOPIG WHICH TRANSLATES ENGLISH TEXT TO PIG LATIN, A TYPICAL EXAMPLE OF A SNOBOL APPLICATION. TO LIST THIS PROGRAM DO:
LIB SNOPIG
LNH
TO RUN THE PROGRAM DO:
LIB SNOPIG
RNH

A COPY OF THIS DOCUMENT (SNOINFO) CAN BE RETRIEVED FROM THE TIMESHIRING LIBRARY BY ENTERING:
LIB SNOINFO
LIST
SNOINFO IS MEANT TO BE PRINTED ON A TELETYPE BECAUSE OF THE CHARACTER SET USED; PRINTING TIME IS THEN ABOUT TWO HOURS. OFFSET COPIES OF A TELETYPE LISTING OF SNOINFO ARE AVAILABLE IN ROOM 140 EXPERIMENTAL ENGINEERING OR FROM THE UCC REFERENCE LIBRARIAN.
IT IS USEFUL TO RUN TIMESHARING PROGRAMS UNDER THE BATCH SUBSYSTEM IF
CONTROL CARD PARAMETERS OTHER THAN THE ONES SUPPLIED BY TELEX ARE
DESIRED OR IF MORE MEMORY SPACE IS NEEDED. FOR EXAMPLE, IF THE
PRIMARY FILE NAME IS "SNOFLAK", TELEX GENERATES:

$SN0B0L C, L=0, I=SNOFLAK.

WHEN "RUN" OR "RNH" IS ENTERED. IF "RNH,X" IS TYPED, TELEX
GENERATES:

$SN0B0L C, L=0, I=SNOFLAK, C=B.

THE COMMANDS BELOW ACCESS SN0B0L IN THE BATCH SUBSYSTEM:
BATCH, 40000. (FOR A FIELD LENGTH OF 40000)
SN0B0L C, PARAMETERS.

FOR EXAMPLE:

BATCH, 40000.
SN0B0L C, I=SNOFLAK, L=LIST.

VALID PARAMETERS FOR THE SN0B0L C CONTROL CARD ARE DESCRIBED BELOW;
ALL ARE ORDER INDEPENDENT.

I - SPECIFIES THE FILE ON WHICH THE SOURCE PROGRAM RESIDES;
DEFAULT IS "I=INPUT".

L - SPECIFIES ONTO WHAT FILE THE LISTING OF THE SOURCE PROGRAM IS
TO GO; DEFAULT IS "L=OUTPUT"). "L=0" SUPPRESSES THE LISTING.

D - SPECIFIES THE FILE TO WHICH THE INITIAL INPUT ASSOCIATION IS
MADE. "D=INPUT" IS THE DEFAULT. IF "D=DATAFIL" IS SELECTED,
INPUT('INPUT', 'DATAFIL', 80) IS PREDEFINED.

0 - SPECIFIES THE FILE TO WHICH THE INITIAL OUTPUT ASSOCIATION IS
MADE. "0=OUTPUT" IS THE DEFAULT. IF "0=RESULTS" IS SELECTED,
OUTPUT('OUTPUT', 'RESULTS', ' ') IS PREDEFINED.

B - SELECTS THE BUFFER SIZE TO BE USED FOR ALL I/O FILES DURING
EXECUTION. DEFAULT IS "B=401" (OCTAL). MINIMUM IS "B=101"
(OCTAL). INCREASING THE BUFFER SIZE SPEEDS UP PROGRAMS DOING
A SUBSTANTIAL AMOUNT OF INPUT/OUTPUT.

F - SETS THE UPPER LIMIT ON FIELD LENGTH (MEMORY) EXPANSION FOR
CAL SN0B0L DURING EXECUTION. DEFAULT IS "F=33000" (OCTAL).
THIS PARAMETER MAY BE USED FOR A FIELD LENGTH > 33000.

C - DETERMINES WHICH CHARACTER SET IS TO BE USED. "C=T"
SPECIFIES THE TIMESHARING TERMINAL CHARACTER SET, AND "C=B"
SPECIFIES THE BATCH (LINE PRINTER) CHARACTER SET. THE
DEFAULT IS DETERMINED BY THE JOB ORIGIN: T FOR TELEX; B FOR
BATCH.

P - DETERMINES THE PAGE LISTING FORMAT. "P=T" SPECIFIES TELETYPewriter
PAGING - NO PAGE EJECTS AND ONE OCCURRENCE OF A 72 COLUMN
HEADER. "P=B" SPECIFIES BATCH PAGING - NORMAL FORMAT. THE
DEFAULT IS DETERMINED BY JOB ORIGIN: T FOR TIMESHARING; B FOR
BATCH.

H - SETTING "H=1" ALLOWS HEAP BLOCK COMPACTATION. DEFAULT IS
"H=0". SEE THE SECTION ON "KNOWN BUGS".

NOTE: THE D, O, C, AND P CONTROL CARD PARAMETERS ARE UNIQUE TO THE
U OF M VERSION OF CAL SNOBOL.

THE KRØNØS RFL CONTROL CARD AND THE SNOBOLC CONTROL CARD F
PARAMETER CONTROL THE MAXIMUM FIELD LENGTH PARAMETER FOR THE CAL
SNOBOL RUN. THE MAXIMUM OF THE F PARAMETER AND THE INITIAL FIELD
LENGTH IS USED.

TELEX ESTIMATES AND SETS FIELD LENGTH FOR A RUN SOLELY ON THE BASIS
OF THE NUMBER OF CHARACTERS IN A FILE. WHILE A GENEROUS AVERAGE IS
ESTIMATED BY TELEX, THE SIZE OF A PROGRAM AND THE FIELD LENGTH THAT
IT REQUIRES VARY WIDELY WITH CAL SNOBOL.

WHEN EXECUTED, THE SNOBOL PROGRAM WILL BE LISTED (UNLESS "L=0" IS
SPECIFIED), AND LISTING DIRECTIVES INSERTED INTO THE SOURCE BECOME
USEFUL. THE LISTING DIRECTIVES AVAILABLE IN CAL SNOBOL ARE -UNLIST,
-LIST, -SPACE, AND -EJECT. -EJECT DOES NOT AFFECT TELETYPING PAGING.
ANY OTHER LINE STARTING WITH A DASH IS CONSIDERED TO BE A COMMENT.

-UNLIST TURNS THE LISTING OFF AT THE POINT IT APPEARS.
-LIST TURNS ON THE LISTING AFTER THE LINE ON WHICH IT APPEARS.
-SPACE CAUSES A BLANK LINE IN THE LISTING WHERE IT OCCURS.
-EJECT CAUSES A PAGE EJECT ("1") IN THE LISTING WHERE IT OCCURS.

CAL SNOBOL ASSUMES THAT THE SOURCE FILE HAS LEADING LINE NUMBERS IF
ITS FIRST CHARACTER IS A DIGIT, AND WILL ACCEPT LINES OF ANY LENGTH.
OTHERWISE, IF THERE ARE NO LINE NUMBERS ON THE SOURCE FILE, CAL
SNOBOL IGNORES COLUMNS 73-80, EVEN IF NON-BLANK, MOVING THEM 10
SPACES OVER TO THE RIGHT ON THE LISTING, AND COMPLETELY IGNORES ANY
CHARACTERS BEYOND COLUMN 80.
CHARACTER SETS

Two character sets are available when using CAL SNOBOL at the U of M. Because the timesharing terminal keyboard is similar to the character set described for standard SNOBOL4, this set is default. There are some differences, the only one affecting CAL SNOBOL being the character for alternation in patterns; this is an up arrow (↑). This set may be specified by "CT" on the SNOBOLC control card.

The other character set corresponds to the CDC 6000/Cyber 70 series line printer character equivalents on timesharing terminals. This set is selected by specifying an "X" on the RUN or RNH command in the SNOBOL subsystem, or "C=B" on the SNOBOLC control card in the batch subsystem.

**CHARACTER SET TABLE**

<table>
<thead>
<tr>
<th>STANDARD SNOBOL4 CHARACTERS</th>
<th>TTY DEFAULT CHARACTERS</th>
<th>LINE PRINTER EQUIVALENTS ON TTYS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-Z</td>
<td>A-Z</td>
<td>A-Z</td>
</tr>
<tr>
<td>0-9</td>
<td>0-9</td>
<td>0-9</td>
</tr>
<tr>
<td>PERIOD</td>
<td>·</td>
<td>·</td>
</tr>
<tr>
<td>COMMA</td>
<td>,</td>
<td>,</td>
</tr>
<tr>
<td>COLON</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>SEMICOLON</td>
<td>;</td>
<td>;</td>
</tr>
<tr>
<td>PLUS</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>MINUS</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>TIMES</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>DIVIDE</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>QUOTE</td>
<td>&quot;</td>
<td>&quot;</td>
</tr>
<tr>
<td>SINGLE QUOTE</td>
<td>'</td>
<td>'</td>
</tr>
<tr>
<td>OPEN PAREN</td>
<td>(</td>
<td>(</td>
</tr>
<tr>
<td>CLOSE PAREN</td>
<td>)</td>
<td>)</td>
</tr>
<tr>
<td>LESS THAN</td>
<td>&lt; ØR [</td>
<td>[ ØR &lt; ARRAY</td>
</tr>
<tr>
<td>GREATER THAN</td>
<td>&gt; ØR ]</td>
<td>] ØR &gt; BRACKETS</td>
</tr>
<tr>
<td>EQUAL SIGN</td>
<td>=</td>
<td>=</td>
</tr>
<tr>
<td>VERTICAL BAR</td>
<td>↑</td>
<td>// ALTERNATION OPERATOR</td>
</tr>
<tr>
<td>DOLLAR SIGN</td>
<td>$</td>
<td>$</td>
</tr>
<tr>
<td>AMPERSAND</td>
<td>&amp;</td>
<td>LINE FEED SEE NOTE BELOW</td>
</tr>
<tr>
<td>AT SIGN</td>
<td>#</td>
<td># SEE NOTE BELOW</td>
</tr>
<tr>
<td>LOGICAL NOT</td>
<td>NONE</td>
<td>NONE SEE NOTE BELOW</td>
</tr>
<tr>
<td>QUESTION MARK</td>
<td>?</td>
<td>&amp; SEE NOTE BELOW</td>
</tr>
<tr>
<td>EXCLAMATION</td>
<td>NONE</td>
<td>&quot; SEE NOTE BELOW</td>
</tr>
<tr>
<td>PERCENT</td>
<td>NONE</td>
<td>? SEE NOTE BELOW</td>
</tr>
<tr>
<td>NUMBER SIGN</td>
<td>NONE</td>
<td>NONE SEE NOTE BELOW</td>
</tr>
</tbody>
</table>

*NOTE:* These last 7 characters are invalid operators in CAL SNOBOL.
DATATYPES

---

STRINGS, NAMES, REALS, INTEGERS, ARRAYS, PATTERNS, CODE, AND PROGRAMMER-DEFINED ARE THE DATATYPES IN CAL SNOBOL; TABLES ARE NOT IMPLEMENTED.

THE PRIMITIVE PATTERN STRUCTURE SUCCEED IS NOT PREDEFINED.

OPERATORS

---

OF THE UNARY OPERATORS, CURSOR POSITION, INTERROGATION, AND NEGATION ARE NOT IMPLEMENTED. DEFERRED EVALUATION (UNEVALUATED EXPRESSIONS USING THE UNARY * ) MAY ONLY BE USED WITH VARIABLES, NOT FULL EXPRESSIONS OR FUNCTION CALLS.

OF THE BINARY OPERATORS, EXPONENTIATION IS NOT AVAILABLE. THE FUNCTION OPSYM (USED FOR CREATING SYNONYM SYMBOLS FOR EXISTING OPERATORS OR FUNCTIONS) IS NOT AVAILABLE.

PATTERN MATCHING IS PERFORMED IN QUICKSCAN MODE ONLY. FULLSCAN MODE IS NOT AVAILABLE. ALL OTHER STANDARD SNOBOL4 OPERATORS ARE VALID.

ARITHMETIC

---

1) THE MAXIMUM MAGNITUDE FOR AN INTEGER IS 10 DIGITS.
2) REAL NUMBERS EXIST IN THE RANGE 10**-294 TO 10**332 WITH 14 SIGNIFICANT DIGITS.
3) EXPONENTIATION OF INTEGERS OR REALS DOES NOT WORK.
4) ARITHMETIC EXPRESSIONS MIXING REAL AND INTEGER OPERANDS IS ALLOWED ONLY BY USING THE FUNCTION CONVERT EXPLICITLY. SEE "PRIMITIVE FUNCTIONS" BELOW.
5) THE FUNCTIONS REMDR (YIELDING THE REMAINDER UPON DIVISION) AND INTEGER (A PREDICATE TEST FOR AN INTEGER ARGUMENT) ARE NOT IMPLEMENTED.

PRIMITIVE FUNCTIONS

---

THE STANDARD SNOBOL4 FUNCTIONS ARG, CLEAR, COLLECT, COPY, DUMP, DUPL, EVAL, FIELD, INTEGER, LOAD, LOCAL, OPSYM, REMDR, REPLACE, ST0PTR, TABLE, TRACE, UNLOAD, AND VALUE ARE NOT IMPLEMENTED.

THE FOLLOWING FUNCTIONS ARE UNIQUE TO CAL SNOBOL (#1), HAVE DIFFERENT CALLS (#2), DIFFERENT DEFINITIONS OVER STANDARD SNOBOL4 (#3), OR ARE U OF M ADDITIONS TO CAL SNOBOL (#4).

APPLY(F,X1,X2,...,XN) - APPLY ASSUMES THAT EXTRA ARGUMENTS ARE THROWN AWAY; THOSE MISSING ARE ASSUMED TO BE NULL (#2).

ARRAY(STRING) - ARRAY TAKES ONLY ONE ARGUMENT -- AN ARRAY PROTOTYPE.
ALL ELEMENTS ARE INITIALIZED TO THE NULL STRING (#2).

CCIMAGE() - CCIMAGE RETURNS A STRING WHOSE VALUE IS THE CURRENT CONTROL CARD IMAGE; THIS IS EITHER THE SNØBØLC CONTROL CARD WITH ALL PARAMETERS AND ANY COMMENTS AFTER THE TERMINATOR CHARACTER, OR THE COMPLETE LOADER CARD IF LOADING A CAL SNØBØL FREEZE FILE. THERE MAY OR MAY NOT BE TRAILING BLANKS (#4).

CLOCK() - CLOCK RETURNS A STRING WHICH IS THE TIME OF DAY IN THE FORM "HH:MM:SS". ANY ARGUMENTS ARE IGNORED (#1).

CONVERT(X) - CONVERT IS ONLY USED FOR NUMERIC DATATYPE CONVERSION. IF X IS REAL, A STRING IS RETURNED; IF X IS INTEGER OR A STRING OF DIGITS, POSSIBLY CONTAINING A PERIOD AND/OR LEADING SIGN, THE CORRESPONDING REAL NUMBER IS RETURNED. CONVERT MUST BE USED IN ARITHMETIC EXPRESSIONS MIXING REALS AND INTEGERS AND TO OUTPUT REAL NUMBERS (#3).


FREEZE('CONCORD')

IS ENCOUNTERED IN A PROGRAM, A FILE CONCORD IS CREATED AND SNØBØL WILL END EXECUTION. THE SYSTEM COMMAND:

SAVE CONCORD

WILL SAVE THIS COPY OF THE FILE. TO RUN THE PROGRAM WHERE IT LEFT OFF, DO:

OLD CONCORD (TIMESHARING) OR GET(CONCORD) (BATCH)

FREEZE IS USEFUL IF A LARGE PART OF THE PROGRAM INITIALIZES STRINGS AND PATTERNS BEFORE COMPUTATION OF RESULTS (#1).

NOTE: FREEZE HAS BEEN MODIFIED AT THE U OF M TO WRITE TWO OVERLAYS WITH STANDARD TYPE *OVL* RECORD NAMES, THUS ALLOWING THE BINARY TO BE EXECUTED AT ANY KRØNØS INSTALLATION.

IF(E1, E2, ..., EN) - IF IS A PREDICATE FUNCTION (SIMILAR TO AN "AND" FUNCTION) ALLOWING AN ARBITRARY NUMBER OF EXPRESSIONS TO BE GROUPED FOR EVALUATION IN ONE STATEMENT (#1).

ITEM(A,S1,S2, ..., SN) - A IS AN ARRAY. IF THE OTHER ARGUMENTS ARE ALL INTEGERS AND A IS N-DIMENSIONAL, THEN ITEM DOES AN NRETURN
OF THE NAME OF THE ELEMENT OF A WHOSE SUBSCRIPTS ARE THE
SI. THE SI CAN ALSO BE STRINGS OF INTEGERS SEPARATED BY
COMMAS. ITEM SCANS THE SI UNTIL ENOUGH INTEGERS TO SATISFY
THE DIMENSIONALITY OF A HAVE BEEN FOUND - THE EXTRA ARE
IGNORED; NULL STRINGS ARE ASSUMED FOR MISSING SUBSCRIPTS.
ITEM FRETURNS IF ANY SUBSCRIPTS ARE OUT OF BOUNDS (#3).

NEXTVAR(X) - IN CAL SNOBOL, IT IS POSSIBLE TO SEQUENTIALLY ACCESS
"FAMILIES" OF VARIABLES. EXAMPLES OF FAMILIES INCLUDE ALL
ELEMENTS OF AN ARRAY, FIELDS IN A PROGRAMMER-DEFINED DATA
STRUCTURE, AND MOST IMPORTANT, ALL SIMPLE VARIABLES IN
EXISTANCE DURING A PROGRAM RUN, BOTH NATURAL AND CREATED.
NEXTVAR RETURNS THE NAME OF THE "NEXT" VARIABLE IN THE SAME
FAMILY AS THE VARIABLE NAMED BY X. X MUST BE OF TYPE NAME OR
STRING. NEXTVAR WILL TREAT A FAMILY OF VARIABLES CYCLICALLY.
FOR ARRAYS, ELEMENTS ARE SELECTED BY THE ORDERED SET OF
INTEGER SUBSCRIPTS (#1).

PROTOTYPE(X) - EXISTS IN CAL SNOBOL AS A GENERALIZATION OF THE
FUNCTIONS ARG, FIELD, AND LOCAL IN STANDARD SNOBOL. HENCE
PROTOTYPE IS DESIGNED TO RETURN THE "PROTOTYPE" DEFINITION
FOR VARIABLES OF NEARLY ALL DATATYPES (#1).

1) IF X IS AN ARRAY, PROTOTYPE RETURNS A STRING OF THE FORM
L1:U1:L2:U2:...:LN:UN WHERE L1:U1 ARE THE LOWER AND UPPER
BOUNDS OF THE ITH DIMENSION OF THE ARRAY. THE L1 AND U1 ARE
INTEGERS WITH LEADING ZEROS AND PLUS SIGN SUPPRESSED.

2) IF X IS A PROGRAMMER-DEFINED DATA STRUCTURE, PROTOTYPE
RETURNS THE <DATA PROTOTYPE> THAT WAS USED IN THE DATA
FUNCTION CALL WHEN X WAS DEFINED.

3) IF X IS A PATTERN, PROTOTYPE RETURNS ONE OF A FIXED SET OF
PATTERN "PROTOTYPES". THESE STRINGS ARE IN THE FORM OF
PROTOTYPES FOR PROGRAMMER-DEFINED DATA STRUCTURES. TO
FURTHER THE ANALOGY BETWEEN PATTERNS AND PROGRAMMER-DEFINED
DATA STRUCTURES, A SET OF PSEUD0-FIELD FUNCTIONS,
CORRESPONDING TO THE FIELD NAMES IN THE PATTERN PROTOTYPES,
HAVE BEEN IMPLEMENTED. THESE PROCEDURES ALLOW A PATTERN TO
BE DECOMPOSED INTO THE OBJECTS USED TO BUILD THE PATTERN.

IF X IS ARB, REM, BAL, FAIL, ABORT, OR FENCE, PROTOTYPE
RETURNS THE STRING ARB(), REM(), ..., NO PSEUD0-FIELD
FUNCTIONS EXIST FOR THESE PATTERNS.

IF X IS A LEN, POS, RP0S, TAB, RTAB, ANY, NOTANY, SPAN,
BREAK, OR ARENØ PATTERN, PROTOTYPE RETURNS THE STRING:
Y(PARAM)
WHERE Y IS EITHER LEN, POS, ETC. THE PSEUD0-FIELD FUNCTION
PARAM MAY THEN BE USED ON ANY OF THESE PATTERNS. E.G.:
PARAM(LEN('3')) RETURNS THE INTEGER 3.
PARAM(ANY('CBAC')) RETURNS THE STRING ABC.
PARAM(ARBNO('CBAC')) RETURNS THE STRING CBAC.

IF X IS A CONCATENATED OR ALTERNATED PATTERN, PROTOTYPE RETURNS THE STRING CAT(FIRST,REST) OR ALT(FIRST,REST). THE PSEUDØ FIELD FUNCTIONS FIRST AND REST ARE DEFINED ON THE CLASS OF CONCATENATED AND ALTERNATED PATTERNS. E.G.:
FIRST('A' + 'B' + 'C') RETURNS THE STRING A.
REST('A' + 'B' + 'C') RETURNS THE PATTERN WHICH IS THE VALUE OF THE EXPRESSION ('B' + 'C').
FOR EXAMPLE, FIRST(BAL REM) RETURNS THE PATTERN BAL.

E.G.: THE PSEUDØ FIELD FUNCTIONS LEFT(P $ R) RETURNS THE PATTERN P, AND RIGHT(P $ R) RETURNS THE VARIABLE NAME R.

IF X IS A DEFERRED EVALUATION PATTERN, I.E. *V, PROTOTYPE RETURNS THE STRING STAR(RIGHT). THE PSEUDØ FIELD FUNCTION RIGHT(X) RETURNS THE VARIABLE NAME V.

4) X CAN ALSO BE A NAME OF A VARIABLE. SEE THE EXPLANATION OF FAMILIES OF VARIABLES IN THE DEFINITION OF THE FUNCTION NEXTVAR ABOVE.
   IF X IS A SIMPLE VARIABLE, PROTOTYPE RETURNS THE STRING INDIRECT(RIGHT). THE PSEUDØ FIELD FUNCTION RIGHT(X) RETURNS THE STRING CORRESPONDING TO THE SIMPLE VARIABLE NAMED BY X.
   E.G.:
   RIGHT(VAR) RETURNS THE STRING VAR.

5) IF X IS AN ARRAY ELEMENT OR A PROGRAMMER-DEFINED DATA STRUCTURE FIELD, PROTOTYPE RETURNS THE STRING ITEM(FAMILY,SELECTOR) OR APPLY(SELECTOR,FAMILY). FAMILY(X) RETURNS THE ARRAY OR OBJECT WHICH CONTAINS THE VARIABLE NAMED BY X; SELECTOR RETURNS A STRING WHICH CAN BE USED TO SELECT THIS VARIABLE IN FAMILY(X).
   ITEM(FAMILY(A[1..J]),SELECTOR(A[1..J])) IS EQUIVALENT TO A[I,J].
   APPLY(SELECTOR(SON(NODE)),FAMILY(SON(NODE))) IS EQUIVALENT TO SON(NODE).

REMARK STRING - REMARK PLACES STRING IN THE DAYFILE. REMARK RETURNS THE NULL STRING. THE STRING MAY BE A MAXIMUM OF 80 CHARACTERS LONG (#2).

TYPE(X) - TYPE IS EQUIVALENT TO DATATYPE(X) EXCEPT FOR THE CASE WHEN X IS A PROGRAMMER-DEFINED DATATYPE (DATA STRUCTURE). TYPE THEN RETURNS THE STRING "DATA", WHEREAS DATATYPE RETURNS THE DATA STRUCTURE NAME STRING WHICH WAS DEFINED IN THE PROTOTYPE TO THE FUNCTION DATA (#1).
INPUT/OUTPUT FUNCTIONS

The function BACKSPACE is not available. ENDCRD AND ENDFIE replace ENDGRUP, and EOR AND EOF replace EORLEVEL, to be consistent with Kronos file structure. (ENDGRUP AND EORLEVEL were Cal Snobol functions designed for the Scope operating system.) The numbering conventions of the previous section are again used below.

DETACH(VNAME) - Removes the I/O association that the string VNAME (a variable) might have (#2).

ENDRECORD(FNAME) - Writes an END OF RECORD on the file specified by the value of FNAME (#4).

ENDIFIE(FNAME) - Writes an END OF FILE on the file specified by the value of FNAME (#2), (#4).

EOR(FNAME) - Succeeds and returns the NULL string if the file identified by the value of FNAME is positioned at an END OF RECORD. EOR fails otherwise (#4).

EOF(FNAME) - Succeeds and returns the NULL string if the file identified by the value of FNAME is positioned at an END OF FILE. EOF fails otherwise (#4).

E0I(FNAME) - Succeeds and returns the NULL string if the file identified by the value of FNAME is positioned at an END OF INFORMATION. E0I fails otherwise (#1).

INPUT(VNAME,FNAME,LENGTH) - INPUT associates the variable represented by the string VNAME to the file represented by the string FNAME, such that any time VNAME is evaluated, the next line (unit record) is read from file FNAME. LENGTH is an integer (default value zero) specifying the number of characters to read from each line on the file. Trailing blanks are supplied to lines read which are shorter than LENGTH. If VNAME already has an I/O association, it is first detached. Input returns the NULL string as its value. The TRIM function can be used subsequently on VNAME to delete trailing blanks (#2).

OUTPUT(VNAME,FNAME,PREFIX) - OUTPUT associates the variable represented by the string VNAME to the file represented by FNAME, such that any new value given to VNAME causes a line (unit record) to be written on the file. PREFIX is a string of length 0 or 1 (normally used as a line printer carriage
CONTRôL) WHICH IS CONCATENATED TO THE BEGINNING OF EACH LINE WRITTEN TO THE FILE. OUTPUT DETACHES ANY PREVIOUS I/O ASSOCIATION THAT VNAME HAS AND RETURNS THE NULL STRING.

OUTPUT-ASSOCIATED VARIABLES ARE NOT AUTOMATICALLY FORMATTED INTO MORE THAN ONE LINE. TO WRITE STRINGS LONGER THAN THE PRINTER WIDTH, "BUFFER" THE RESULT. E.G.: IF VARIABLE X CONTAINS THE STRING TO BE WRITTEN TO FILE OUTPUT, THEN

```
OUTLOO P X LEN(N) . OUTPUT = S(OUTLOO P) OUTPUT = DIFFER(X) X
```

PRINTS THE VALUE OF X ON ONE OR MORE LINES OF LENGTH N (#2).

REWIND(FNAME) - REWIND REPOSITIONS A FILE REPRESENTED BY THE STRING FNAME TO ITS BEGINNING OF INFORMATION. REWIND WRITES AN END OF RECORD ON AN OUTPUT TYPE FILE (IF IT HAS BEEN WRITTEN UPON) BEFORE REWINDING. REWIND RETURNS THE NULL STRING (#2).

FURTHER NOTES:
1) FILE ASSOCIATIONS TO THE VARIABLE NAMES INPUT AND OUTPUT ARE PREDEFINED. THE DEFAULTS ARE INPUT('INPUT', 'INPUT', 80) AND OUTPUT('OUTPUT', 'OUTPUT', ' '). PREDEFINED ASSOCIATIONS FOR THE VARIABLE NAMES INPUT AND OUTPUT CAN BE MADE INITIALLY TO OTHER FILES BY USING THE D AND 0 PARAMETERS ON THE SNOBOL4 CONTROL CARD. (SEE THE DISCUSSION OF CONTROL CARD PARAMETERS IN THE SECTION "TIMESHARING INFORMATION").

2) THE FILE "OUTPUT" HAS SPECIAL MEANING UNDER TIMESHARING. INFORMATION WRITTEN TO IT IS PRINTED BEFORE THE NEXT INPUT REQUEST IS MADE, THEREBY ENABLING INTERACTIVE INPUT/OUTPUT. READING AN END OF RECORD (EOR) CAUSES FAILURE BUT SUBSEQUENT READS CAN BE MADE WITHOUT REDEFINING THE INPUT ASSOCIATION FOR THE INPUT FILE.

3) THE VARIABLE "PUNCH" DOES NOT HAVE A PREDEFINED ASSOCIATION IN CAL SNOBOL.

```
OUTPUT('PUNCH', 'PUNCH')
```

MAKES THE STANDARD SNOBOL4 PUNCH ASSOCIATION.

```
OUTPUT('NEWPAGE', 'OUTPUT', 1)
```

PROVIDES A MEANS OF PAGE EJECTING ON LINE PRINTER OUTPUT IF A SUBSEQUENT ASSIGNMENT IS MADE TO NEWPAGE.

KEYWORDS

KEYWORDS ARE IMPLEMENTED AS PRIMITIVE FUNCTIONS IN CAL SNOBOL, AND THEREFORE THE UNARY & OPERATOR IS INVALID. THE KEYWORD FUNCTIONS AVAILABLE ARE DISCUSSED HERE.

ALPHABET() RETURNS AS ITS VALUE A STRING CONTAINING THE 63 CHARACTERS OF THE CDC 6000/CYBER 70 SERIES CHARACTER SET IN THEIR STANDARD COLLATING SEQUENCE (IN NUMERICAL ORDER BY DISPLAY CODE). TWO OF THESE (DISPLAY CODES 74B AND 76B) ARE
NORMALLY USED AS ASCII ESCAPE CHARACTERS ON TIMESHARING TERMINALS AND GIVE UNUSUAL OR UNDESIRABLE EFFECTS WHEN USED ALONE.

ANCHOR(x) TURNS ON THE ANCHORED MODE OF PATTERN MATCHING IF X IS ANY NON-NULL EXPRESSION; TURNS OFF ANCHORED MODE OTHERWISE. INITIALLY SNOBOL PATTERN MATCHING IS NOT IN ANCHORED MODE. EXAMPLE: ANCHOR(1) TURNS IT ON; ANCHOR(0) TURNS IT OFF.

FNLEVEL() RETURNS AN INTEGER CORRESPONDING TO THE NUMBER OF FUNCTIONS CALLED WHICH HAVE NOT YET RETURNED.

MAXLENGTH(I) RETURNS THE VALUE CORRESPONDING TO THE MAXIMUM ALLOWABLE LENGTH STRING IN SNOBOL IF I IS NULL. FOR I AN INTEGER, THIS MAXIMUM IS RESET TO THE VALUE I. INITIALLY MAXLENGTH() HAS THE VALUE 131070.

STCOUNT() RETURNS THE RUNNING TOTAL OF THE NUMBER OF STATEMENTS WHICH HAVE BEEN EXECUTED SO FAR.

STLIMIT(I) RETURNS THE VALUE CORRESPONDING TO THE MAXIMUM NUMBER OF STATEMENTS ALLOWED TO BE EXECUTED BEFORE EXECUTION IS HALTED IF I IS NULL; FOR I AN INTEGER, THIS VALUE IS RESET TO I. INITIALLY STLIMIT() HAS THE VALUE 1000000.

THERE ARE NO PROTECTED KEYWORDS IN SNOBOL (E.G. &ERRTYPE).

UNIVERSITY OF MINNESOTA MODIFICATIONS TO SNOBOL

ALL MODIFICATIONS ARE NOTED AS SUCH IN THE SECTION WHERE THEY ARE PRIMARILY DESCRIBED. USE THE INDEX AT THE END OF THIS DOCUMENT TO LOCATE A PARTICULAR SECTION. MODIFICATIONS ARE GAUGED HERE AS UNIQUE TO THE U OF M VERSION (#1); EXTENDED AND ALTERED (NOT COMPATIBLE) (#2); AND EXTENDED (COMPATIBLE) (#3).

1) LINE NUMBERS (#1)
2) INPUT AND OUTPUT ASSOCIATIONS (#2)
3) INTERACTIVE INPUT/OUTPUT (#1)
4) TTY CHARACTER SET (#1)
5) CCIMAGE FUNCTION (#1)
6) FREEZE FUNCTION (#3)
7) ENDRECORD FUNCTION (#1)
   (ENDRECORD REPLACES EN DGROUP FUNCTION)
8) ENDFILE FUNCTION (#1)
   (ENDFILE REPLACES EN DGROUP FUNCTION)
9) EOR FUNCTION (#1)
   (EOR REPLACES EORLEVEL FUNCTION)
10) EOF FUNCTION (#1)
   (EOF REPLACES EORLEVEL FUNCTION)
11) CONTROL CARD PARAMETERS (#1)
KNOWN BUGS

1) Garbage Collection occurs in two phases, list storage compaction and heap block compaction. List storage compaction works properly. However, heap block compaction still contains numerous bugs and is turned off (see No. 4, below). When string space fills up, garbage collection is invoked. If there is not enough space to garbage collect, CAL SNOBOL will error terminate with the message "the maximum field length has been exceeded." An increase in field length usually fixes this. When running a program, extra memory (up to the maximum allowable) can be allotted by using the MI or MA parameter on the RUN command under the SNOBOL subsystem, or by running under the batch subsystem, using the F parameter on the SNOBOLC card.

2) Exponentiation of reals or integers does not work.

3) Deferred evaluation (unary * operator for unevaluated expressions) applied to anything other than a variable name is not detected as an inappropriate CAL SNOBOL expression. To perform deferred evaluation of a function, for example: *F(X), where F does not return a variable name, use an NRETURN in F.

4) Documentation from Berkeley accompanying the present version of CAL SNOBOL states that heap block compaction is a major source of bugs ("and may always be"). Heap block compaction is therefore turned off. It may be enabled by specifying H=1 on the SNOBOL control card under the batch subsystem, and calling the primitive function HBC (HBC has no arguments) at the place in the program where heap block compaction is desired. The message "heap block compaction called" is printed in the dayfile every time HBC is called. However, use of heap block compaction is not advised except for experimentation, and results cannot be guaranteed.

5) Concatenation in the subject field of a statement (using parentheses) is a compile error under some conditions.

6) Redefining standard primitive functions causes abnormal termination.

7) Values of literals can be changed when they should not. Example: 'HI THERE' 'HI' *XXX'; OUTPUT = 'XXX' writes the letters:
   HI

8) Embedded alternation symbols in a number cause an abort.

9) Duplicate labels resulting from compiling more code at execution time are not handled properly by CAL SNOBOL.
SIMULATED FUNCTIONS ON SNOLIB

---------

Because several standard SNOBOL4 functions are not predefined in CAL SNOBOL, an attempt has been made at the University of Minnesota to write some of these functions in SNOBOL in an efficient manner and to place them on a file for easy access to SNOBOL users. Other useful non-standard functions are made available also. This file may be gotten before a SNOBOL program is run and then read in and coded at execution time in the SNOBOL program using the CODE function. At the present time, on the library file SNOLIB, the functions

<table>
<thead>
<tr>
<th>GET</th>
<th>ANCHR</th>
<th>DUPL</th>
<th>EVAL</th>
<th>INTEGER</th>
<th>REPLACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE</td>
<td>REMDR</td>
<td>TRACE</td>
<td>STØPTR</td>
<td>DUMP</td>
<td>CLEAR</td>
</tr>
<tr>
<td>REVERSE</td>
<td>LFN</td>
<td>SØRTI</td>
<td>SØRTØ</td>
<td>STATUS</td>
<td>RANDOM</td>
</tr>
</tbody>
</table>

are available. In addition, the primitive pattern structure SUCCEED is given a value. To access these functions enter the following lines of code at the beginning of your program: (line numbers are shown for timesharing)

00100 INPUT('SNØ..1','SNOLIB',1000)
00110 REWIND('SNOLIB')
00120 SNØ.LIB = CODE(TRIM(SNØ..1)) ;S<SNØ.LIB>
00130 OUTPUT = 'SNOLIB NOT FOUND' ;(END)
00140 SNØLIB.CODED GET( )
00150 * CONTINUE WITH YOUR PROGRAM BELOW

The identifiers 'SNØ..1', 'SNOLIB', 'SNØ.LIB', and the label 'SNØLIB.CODED' should be used exactly as given here for best results. The input association length must be 1000. Also, the function GET is needed as the library manager.

Your program may now operate under the assumption that the functions requested through the GET function are available as part of CAL SNOBOL. See the description of the GET function, below, for its purpose and operation, and for the description of the structure of SNOLIB.

Obtain SNOLIB before you use your program by doing:

GET SNOLIB/UN=LIBRARY.

The statement in line 00110 in the code given above ensures positioning of SNOLIB at the beginning for every run of your program. SNOLIB must be on a local file named "SNOLIB".

The SNOLIB functions are described below. The immediately following list is in the same order as the functions in the library. For functions identical in operation to standard SNOBOL, the reader is directed to the book:

"THE SNOBOL4 PROGRAMMING LANGUAGE" by GRISWOLD ET AL.

--- PAGE 16 ---
"STANDARD" FUNCTIONS

GET(STRING) [CONTAINS ANCHR()]
DUPL(STR,N)
EVAL(STR)
INTEGER(STR)
REPLACE(STR1,STR2,STR3)
TABLE(VAR,CHAR)
REMDR(X,Y)
TRACE(VAR,FILE,TAG)
ST0PTR(VAR)
DUMP(MESG,FILE,TERM)
CLEAR()
REVERSE(STR)

[ THE PATTERN "SUCCEED" ]

"EXTENDED" FUNCTIONS

LFN(STRING)
S0RTI(STRING,DELMITER,PROTECT0R)
S0RT0(DELETE)
STATUS(MESG,FILE)
RAND0M(RANGE,COUNT,DELIM,SEED)

GET(STRING) - THE GET FUNCTION, ALTHOUGH NOT A STANDARD SN0B0L FUNCTION, FORMS THE BASIS OF SN0LIB USAGE AND IS USED TO DEFINE AND CODE SUBSEQUENT FUNCTIONS. GET IS ANALOGOUS TO "LOAD" IN STANDARD SN0B0L.

THE ARGUMENT "STRING" MAY Be A LIST OF SN0LIB FUNCTION NAMES SEPARATED BY COMMAS OR IT MAY BE NULL. IF STRING IS NULL, ALL "STANDARD" FUNCTIONS IN SN0LIB ARE DEFINED AND CODED. ALTERNATELY, IF THE ARGUMENT GIVEN IS AN ASTERISK ('*'), ALL FUNCTIONS (INCLUDING "EXTENDED" FUNCTIONS) ARE READ AND CODED.

"STRING" MAY BE A LIST OF AVAILABLE FUNCTION NAMES SEPARATED BY COMMAS AND THEN GET WILL ONLY DEFINE AND CODE THOSE FUNCTIONS, RETURNING AS ITS VALUE THE LIST OF ANY FUNCTION NAMES THAT WERE NOT FOUND IN SN0LIB (COMPARING THE USER'S LIST AGAINST AN INTERNAL LIST).

SN0LIB IS ORGANIZED WITH THE NAME OF A FUNCTION ON ONE LINE FOLLOWED BY THE ENTIRE FUNCTION DEFINITION AND BODY ON THE NEXT LINE. THE MOST FREQUENTLY USED AND SMALLER FUNCTIONS ARE AT THE BEGINNING. GET DOES NOT READ THROUGH ALL OF SN0LIB IN ALL CASES. ONLY THE FUNCTIONS NEEDED ARE CODED IN ORDER TO SAVE MEMORY SPACE. OTHERS MAY BE OBTAINED THROUGH GET LATER IN THE PROGRAM. IN ACCESSING SN0LIB IN YOUR PROGRAM, THE LINE:

00140 SN0LIB CODED GET()

PROVIDES THE RE-ENTRY LABEL FOR THE DEFINITION OF THE GET FUNCTION, AND THIS EXACT LABEL MUST BE PRESENT. THE CALL TO GET MUST BE IN THE SAME STATEMENT AS THE RE-ENTRY LABEL OR SUBSEQUENT TO IT. AS DISCUSSED ABOVE, THE ARGUMENT TO GET MAY BE NULL, OR MAY CONTAIN A STRING OF FUNCTION NAMES AS IN:

00140 SN0LIB CODED GET('DUPL,LFN,DUMP')
GET MAY BE CALLED MORE THAN ONCE; IN FACT SOME OF THE SNOLIB
FUNCTIONS THEMSELVES MAKE CALLS TO GET TO COMPLETE THEIR
DEFINITIONS (SEE "SNOLIB NOTES"). GET WILL NOT CODE THE
SAME FUNCTION TWICE IN THE SAME PROGRAM RUN.

ANCHR() - GET DEFINES ANCHR (WHICH HAS NO ARGUMENTS) FOR ITS OWN
NEEDS, AND ANCHR MAY BE USED FREELY BY THE SNOLIB USER. ANCHR
RETURNS THE NULL STRING IF THE ANCHOR IS OFF, AND RETURNS "1"
IF THE ANCHOR IS ON (SEE "KEYWORDS" FOR THE ANCHOR FUNCTION).

DUPL(STR,N) - RETURNS A STRING WHICH IS N REPETITIONS OF STR; RETURNS
THE NULL STRING IF N=0; RETURNS IF N<0 (SEE GRISWOLD ET AL.).

REVERSE(STR) - RETURNS AS ITS VALUE THE STRING ARGUMENT STR WITH ITS
CHARACTERS IN REVERSE ORDER.

EVAL(STR) - RETURNS THE VALUE OF STR (ANY EXPRESSION) BY EXECUTING
THE CODE FUNCTION UPON THE STRING. EVAL RETURNS IF THE
EVALUATION OF THE EXPRESSION FAILS. UNEVALUATED EXPRESSIONS
(USE OF THE UNARY OPERATOR "**" BEFORE AN EXPRESSION) CANNOT
BE HANDLED BY EVAL. UNLIKE THE EVAL IN STANDARD SN0B0L4, N0R
DOES EVAL CHECK SYNTAX.

INTEGER(STR) - THIS PREDICATE TEST SUCCEEDS IF AND ONLY IF THE VALUE
OF STR IS A SIGNED OR UNSIGNED INTEGER; THE NULL STRING IS
EQUIVALENT TO ZERO AND SUCCEEDS. (SEE GRISWOLD ET AL.)

REPLACE(STR1,STR2,STR3) - PERFORMS A ONE FOR ONE CHARACTER
REPLACEMENT. (SEE GRISWOLD ET AL. FOR A FULL DISCUSSION.)

TABLE(VAR,CHAR) - CREATES A TABLE NAMED VAR. TABLE ELEMENTS ARE
SUBSEQUENTLY REFERENCED AS FUNCTION CALLS WITH PARENTHESIS
("(" AND "))". BECAUSE TABLES ARE SIMULATED BY USING INDIRECT
REFERENCE, A PREFIX CHARACTER IS SUPPLIED TO PROTECT TABLE
ENTRIES. CHAR ALLOWS THE USER TO SPECIFY THIS CHARACTER; THE
DEFAULT IS AN EXCLAMATION POINT ("!"). FOR EXAMPLE,
TABLE('TEXT') CREATES A TABLE WHOSE ENTRIES MIGHT BE
TEXT('W0RD') , TEXT('AN'). (AVOID CREATING TABLES OF THE
SAME NAME AS OTHER FUNCTIONS AND NOTE THAT THE "CONVERT"
FUNCTION IN CAL SN0B0L D0ES NOT OPERATE ON ARRAYS OR TABLES.)

REMDR(X,Y) - RETURNS THE INTEGER REMAINDER OF THE DIVISION OF X BY Y.
(SEE GRISWOLD ET AL.)

SUCCEED - NOT AN ACTUAL FUNCTION, BUT A PRIMITIVE PATTERN. SUCCEED
USED IN A PATTERN MATCH ALWAYS ENSURES SUCCESS AT THE POINT
IT IS ENCOUNTERED. (SEE GRISWOLD ET AL.)

STATUS(MESG,FILE) - DISPLAYS THE VALUES OF THE CURRENT TIME, CLOCK,
DATE, FNCLEVEL, STCOUNT, STLIMIT, AND MAXLENGTH UPON THE FILE
(DEFAULT IS OUTPUT); TOGETHER WITH THE OPTIONAL MESSAGE
STRING MESG. STATUS RETURNS THE NULL STRING. (FNCLEVEL IS

--- PAGE 18 ---
ADJUSTED (FNCLEVEL() - 1) FOR THE LEVEL OF STATUS ITSELF.

TRACE(VAR, FILE, TAG)
ST0PTR(VAR) - TRACE PRINTS VALUES OF NATURAL VARIABLES ONLY (BY
OUTPUT ASSOCIATION). ST0PTR MAY BE CALLED TO DETACH THE
VARIABLE FOR WHICH TRACE HAS BEEN CALLED.

VAR - THE VARIABLE TO BE TRACED (OR ST0PTRACED); MUST BE A
NATURAL VARIABLE EXPRESSED AS A STRING.
FILE - THE FILE TO RECEIVE THE TRACE OUTPUT, DEFAULT
"OUTPUT".
TAG - OPTIONAL ONE CHARACTER (N0N-ASCII) TAG. IF NOT GIVEN,
THE FIRST CHARACTER OF THE VARIABLE NAME IS USED. THE TAG
IS APPENDED TO THE FRONT OF A NEWLY ASSIGNED VALUE USING
THE PREFIX ARGUMENT OF THE OUTPUT FUNCTION AND THIS STRING
IS OUTPUT. SPECIFYING THE TAG IS USEFUL IF THE FIRST
CHARACTERS OF TWO OR MORE VARIABLES TO BE TRACED ARE THE
SAME.

ST0PTR IS DEFINED WHENEVER TRACE IS. UNFORTUNATELY, TRACE
CANNOT PROVIDE MORE INFORMATION ABOUT THE STATE OF A
VARIABLE SUCH AS TRACE IN STANDARD SN0B0L4.

LFN(TESTFN) - RETURNS THE SAME LOGICAL FILE NAME AS THE ARGUMENT
TESTFN IF TESTFN IS A LEGAL KR0N0S OPERATING SYSTEM FILE
NAME. OTHERWISE LFN RETURNS.

SORTI STRING, DELIMITER, PROTECTOR
SORT0 DELETE - SORTI AND SORT0 ARE USED TOGETHER TO DO IN-CORE
LEXICAL SORTING OF STRINGS. ONLY SORTI NEEDS TO BE OBTAINED
VIA GET, AS SORTI AND SORT0 ARE DEFINED TOGETHER. THE
PARAMETERS DELIMITER, PROTECTOR, AND DELETE ARE OPTIONAL
ONES USED TO ALLOW FLEXIBILITY. SORTI AND SORT0 ARE FIRST
DISCUSSED FOR THE CASE IN WHICH THESE PARAMETERS REMAIN NULL.

IN GENERAL, STRINGS ARE INSERTED ONE AT A TIME VIA SORTI INTO
A MASTER LIST; WHEN NO MORE STRINGS ARE TO BE ADDED, SORT0
MAY BE REFERENCED TO RETURN STRINGS ONE AT A TIME IN LEXICAL
ORDER. AFTER ALL STRINGS HAVE BEEN RETRIEVED, SORT0 FAILS;
INSERTION AND RAdIX METHODS ARE USED.

THIS SORT HAS BEEN FOUND TO BE EFFICIENT IN TIME AND STRING
SPACE USED.

THE PARAMETERS WHICH ALLOW CHANGES IN THE USE OF SORTI AND
SORT0 SPECIFY SPECIAL CHARACTERS, SELECT DELETION OF MEMBER
STRINGS RETURNED THROUGH SORT0, AND REINITIALIZE THE SORT
FUNCTION PAIR FOR ANOTHER SORT JOB IN THE SAME PROGRAM RUN.

STRING - SPECIFIES THE STRING TO BE SORTED (INTO THE LIST),
THE LEADING CHARACTER OF WHICH MUST BE ALPHANUMERIC. THE
CHARACTERS FOLLOWING MAY BE ANY CHARACTER IN THE CDC
6000/CYBER 70 SERIES CHARACTER SET OTHER THAN THE DELIMITER CHARACTER. THE STRINGS MAY BE OF ANY LENGTH. THE SORT IGNORES STRINGS WHICH ARE EXACT DUPLICATES OF STRINGS ALREADY IN THE SET, STRINGS WITH NON-ALPHANUMERIC FIRST CHARACTERS, AND THE NULL STRING.

DELMITER - USED DURING THE SORT TO DELIMIT SUBSTRINGS; SHOULD NOT OCCUR AS ANY CHARACTER IN THE STRINGS TO BE SORTED. IF NOT SPECIFIED, A RIGHT PARENTHESIS ("(") IS USED. DELIMITER MUST BE A SINGLE CHARACTER.

PROTECTOR - USED IN AN INDIRECT REFERENCE DURING THE SORT TO SAFELY MAINTAIN UP TO 36 DISTINCT SUB-LISTS. IF NOT SPECIFIED, A BACKSLASH (\"\") IS USED.

DELETE - ANY NON-NULL ARGUMENT, WHICH IN THE FIRST CALL TO S0RT0, WILL SET THE CONDITION FOR DELETING STRINGS RETURNED DURING THE SORT.

S0RTI AND S0RT0 MAY BE USED AGAIN DURING A PROGRAM ONCE THE LIST HAS BEEN EXHAUSTED (S0RT0 RETURNS) BY ADDING TO THE LIST (IF IT WAS NOT DELETED) OR STARTING ON A NEW LIST (HAVING DELETED STRINGS IN THE OLD LIST WITH THE DELETE PARAMETER TO S0RT0).

RAND0M(RANGE, COUNT, DELIM, SEED) - RETURNS A STRING OF ONE OR MORE RANDOM INTEGERS UNIFORMLY DISTRIBUTED ON THE INTERVAL 0..9 OR IN THE RANGE SPECIFIED BY THE CALL. EACH NUMBER IS SEPARATED BY A SPACE OR THE DELIMITER GIVEN AS A LITERAL IN THE CALL.

RANGE - SPECIFIES THE RANGE OF RANDOM INTEGERS (SIMILAR IN FORM TO AN ARRAY INDEX RANGE). THAT IS, A SIMPLE INTEGER WILL SET THE EXCLUSIVE UPPER LIMIT OF THE VALUE OF THE NUMBER(S) RETURNED, AND AN ARGUMENT OF THE FORM 'J+K' IS USED TO SET THE LOWER BOUND OTHER THAN ZERO AND AN INCLUSIVE UPPER LIMIT (BOUNDS MAY BE NEGATIVE OR POSITIVE OR BOTH). THE DEFAULT RANGE IS '0:9'.

COUNT - TELLS HOW MANY RANDOM NUMBERS TO RETURN WITH EACH CALL. IF COUNT > 1, RAND0M WILL RETURN A STRING OF RANDOM NUMBERS (NOT AN ARRAY) DELIMITED BY A BLANK (' ') OR BY AN OPTIONAL CHARACTER SPECIFIED BY DELIM. IF THE COUNT IS LESS THAN OR EQUAL TO ZERO, THE COUNT IS SET TO ONE.

DELIM - SPECIFIES THE DELIMITER CHARACTER (DEFAULT VALUE IS A BLANK) TO BE USED TO SEPARATE THE RANDOM NUMBERS RETURNED AS A STRING. A SPECIAL CHARACTER - 'N' - IS A CUE TO USE THE NULL STRING AS THE DELIMITER. IF COUNT IS ONE, DELIM IS SET TO NULL.

SEED - RAND0M USES THE TIME FUNCTION TO SEED ITSELF; IF AN INTEGER IS PRESENT AS THE FOURTH ARGUMENT, IT IS USED TO SEED THE RANDOM NUMBER ALGORITHM. IF ANYTHING ELSE NON-NULL IS PRESENT, RAND0M RE-SEEDS ITSELF FROM THE TIME FUNCTION. A SEQUENCE OF THE ALGORITHM CONTINUES UNTIL A NEW SEED VALUE OR RESEED PARAMETER IS GIVEN.

FOR EXAMPLE; RAND0M('1:52',10) RETURNS A STRING OF TEN NUMBERS IN THE RANGE 1 TO 52 DELIMITED BY BLANs.
DUMP(MESG,FILE,TERM) - DUMP USES THE FUNCTION NEXTVAR TO CYCLE THROUGH THE NATURAL VARIABLES IN THE CAL SNOBOL SYSTEM AND LIST THE VALUE OF STRINGS AND NUMBERS OR THE TYPES OF VARIABLES OF OTHER DATATYPES.

MESG - ANY STRING OF CHARACTERS USED AS A LABEL TO IDENTIFY THE DUMP (DEFAULT NULL).

FILE - THE NAME OF THE FILE (DEFAULT OUTPUT) TO RECEIVE THE DUMP.

TERM - SPECIFIES PROGRAM TERMINATION IF NONNULL, OTHERWISE TRANSFER IS BACK TO THE CALLING STATEMENT.

DUMP REFERENCES ALL NATURAL VARIABLES. IF IT ENCOUNTERS A VARIABLE WITH AN INPUT ASSOCIATION, A NEW INPUT ASSIGNMENT IS MADE TO THE VARIABLE. THEREFORE IT IS IMPORTANT THAT INPUT VARIABLES BE DETACHED TO AVOID UNEXPECTED RESULTS. (THE CONTENTS OF THE INPUT BUFFER MAY BE LOST.) THE VARIABLE "INPUT" IS DETACHED UPON ENTRY TO DUMP, AND REDEFINED AS ('INPUT','INPUT',80) IF TRANSFERRING BACK TO THE CALLING STATEMENT. DUMP DOES NOT YIELD THE VALUES OF THE VARIABLES USED IN SNOLIB ITSELF AS THEY CAUSE EXCESSIVELY LONG OUTPUT.

CLEAR() - CLEAR SEARCHES FOR ALL NATURAL VARIABLES AND SETS THEM TO THE NULL STRING. THE PRIMITIVE PATTERNS ABORT, ARB, BAL, FAIL, FENCE, REM, SUCCEED ARE INCLUDED AS LOCAL VARIABLES IN THE DEFINITION OF CLEAR TO PROTECT THEM. CLEAR HAS NO ARGUMENTS.

VARIABLES WITH AN OUTPUT ASSOCIATION SHOULD BE DETACHED BEFORE CALLING CLEAR. ("OUTPUT" IS DETACHED AND REDEFINED AS OUTPUT('OUTPUT','OUTPUT','')) CLEAR WILL NOT AFFECT THE VARIABLES IN SNOLIB ITSELF, BECAUSE OF A PROTECTION ARRANGEMENT.

IF THERE ARE VARIABLES DIFFERENT FROM THOSE LISTED ABOVE TO PROTECT, CLEAR SHOULD BE REDEFINED AFTER THE CALL TO GET WHICH FIRST DEFINES CLEAR, DECLARING ALL THOSE VARIABLES TO BE RESTORED UPON RETURN AS LOCAL VARIABLES.

EXAMPLE:
DEFINE('CLEAR(REM,BAL,MYVAR,N0JUNK,GOODSTUFF')
CLEAR()
SNOLIB NOTES

1) THE FOLLOWING FUNCTIONS MAKE REFERENCE TO OTHER FUNCTIONS IN
SNOLIB:

GET - NEEDS ANCHR
RANDOM - NEEDS INTEGER
DUMP - NEEDS LFN
STATUS - NEEDS LFN
TRACE - NEEDS LFN

AND MAKE CALLS TO GET TO COMPLETE THE DEFINITION OF THE
NEEDED FUNCTIONS. IF A FUNCTION IS ALREADY DEFINED AND
CODED, GET DOES NOT DO SO AGAIN.

2) CLEAR IS THE ONLY FUNCTION WITH NO ARGUMENTS. THE FOLLOWING
FUNCTIONS WILL OPERATE ON THEIR DEFAULTS UPON A NULL CALL:

GET() - DEFINES THE STANDARD FUNCTIONS.
EVAL() - THE NULL STRING CAN BE CODED.
INTEGER() - THE NULL STRING IS EQUIVALENT TO ZERO.
RANDOM() - EQUIVALENT TO RANDOM('0:9','1:10').
DUMP() - EQUIVALENT TO DUMP('OUTPUT').
CLEAR() - HAS NO ARGUMENTS.
SORT0() - RETURNS THE NEXT MEMBER OF A SORTED SET.

3) INTERNAL TO THE FUNCTIONS, CERTAIN CONVENTIONS ARE OBSERVED.

A) ALL FUNCTION NAMES ARE ENTRY POINTS. DUPLICATE LABELS ARE
NOT TAKEN CARE OF PROPERLY BY CAL SNOBOL (AS DESCRIBED IN
GRISWOLD ET AL.) AND ARE AN ERROR.

B) ALL GLOBAL VARIABLES AND OTHER LABELS USED IN SNOLIB BEGIN
WITH THREE ALPHABETIC CHARACTERS FOLLOWED BY TWO PERIODS
AND AN INTEGER (EXAMPLE: "DIM..3") TO AVOID CONFLICT WITH
USER VARIABLES. IN ADDITION, INDIRECT REFERENCE (USING
ASCII CHARACTERS) IS USED IN SOME OF THE FUNCTIONS (EVAL,
TRACE, GET, SORTI, SORT0).

4) IF YOUR PROGRAM REACHES MAXIMUM FIELD LENGTH (WHICH IS SET TO
33000 UNDER THE RUN OR RNH COMMAND IN TIMESHARING), SEE THE
SECTION "TIMESHARING INFORMATION" FOR HELP IN EXTENDING
YOUR FIELD LENGTH. ALSO SEE "KNOWN BUGS".

5) FAILURE (FRETURN) WILL RESULT FROM A CALL TO ANY FUNCTION IN
SNOLIB THAT IS GIVEN AN ILLEGAL FILE ARGUMENT (TRACE, DUMP,
STATUS).

6) TO EXAMINE A FILE THAT A FUNCTION CREATES, REWIND THE FILE
AND COPY IT TO OUTPUT. Thus:

REWIND, FILENAM (TIMESHARING) REWIND(FILENAM) (BATCH)
LNIH, F=FILENAM CBF(FILENAM, OUTPUT)
INDEX TO SNOINFO 3.3

13 - ALPHABET(
11 - ALT(FIRST, REST)
14 - ANCHOR(X)
18 - ANCHR() [SN0LIB]
8 - APPLY(F, X1, X2, ..., XN)
8 - ARITHMETIC
8 - ARRAY(STRING)
13 - ASCII CHARACTERS
5 - BATCH CONTROL CARD PARAMETERS
5 - BATCH SUBSYSTEM
14 - BUGS, KNOWN
11 - CAT(FIRST, REST)
9 - CCIMAGE() [ U 0F M ]
7 - CHARACTER SETS
7 - CHARACTER SET TABLE
21 - CLEAR()
9 - CLOCK()
5 - CONTROL CARD PARAMETERS [ U 0F M ]
9 - CONVERT()
8 - DATATYPES
12 - DETACH(VNAME)
11 - DOFL(LEFT, RIGHT)
20 - DUMP(MESG, FILE, TERM) [SN0LIB]
18 - DUPL(STR, N) [SN0LIB]
12 - ENDFILE(FNAME) [ U 0F M ]
12 - ENDRECORD(FNAME) [ U 0F M ]
12 - EOF(FNAME) [ U 0F M ]
12 - E0I(FNAME)
12 - E0R(FNAME) [ U 0F M ]
4 - ERROR MESSAGES
18 - EVAL(STR) [SN0LIB]
4 - EXAMPLE PROGRAM (SNOPIG)
8 - EXPONENTIATION (SEE ALSO KNOWN BUGS)
11 - FAMILY(X)
11 - FIRST(PATTERN)
14 - FNLEVEL()
9 - FREEZE(FNAME, RECCNAME) [ U 0F M ]
8 - FULLSCAN MODE
12 - FUNCTIONS, INPUT/OUTPUT
13 - FUNCTIONS, KEYWORD
8 - FUNCTIONS, PRIMITIVE
16 - FUNCTIONS, SIMULATED [SN0LIB]
15 - GARBAGE COLLECTION
17 - GET(STRING) [SN0LIB]
15 - HEAP BLOCK COMPACTOR BUGS
9 - IF(E1, E2, ..., EN)
12 - INPUT/OUTPUT FUNCTIONS
12 - INPUT(VNAME, FILENAME, LENGTH)
13 - INPUT AND OUTPUT ASSOCIATIONS [ U 0F M ]
8 - INTEGERS
18 - INTEGER(STR) [SN0LIB]
13 - INTERACTIVE INPUT/OUTPUT [ U 0F M ]
2 - INTRODUCTION
INDEX TO SNOLIB 3.3  75/04/07. 23.21.43.

9 - ITEM(A,S1,S2,...,SN)
13 - KEYWORDS
15 - KNOWN BUGS
11 - LEFT(NAMING,PATTERN)
19 - LFN(TESTFN) [SNOLIB]
3 - LINE NUMBERS FOR TIMESHARING [ U OF M ]
6 - LISTING DIRECTIVES
14 - MAXLENGTH(I)
8 - MIXED TYPES IN ARITHMETIC EXPRESSIONS
10 - NEXTVAR(X)
8 - OPERATORS
12 - OUTPUT(VNAME,FNAME,PREFIX)
10 - PARAM(PRIM,FNC)
11 - PRD(LEFT.RIGHT)
3 - PRIMITIVE FUNCTIONS
3 - PROGRAM PREPARATION (TIMESHARING)
10 - PROTOTYPE(X)
10 - PSEUDOFIELD FUNCTIONS
8 - QUICKSCAN MODE
20 - RANDOM(RANGE,COUNT,DELIM,SEED) [SNOLIB]
8 - REAL NUMBERS
2 - REFERENCE BOOKS
11 - REMARK STRING
18 - REMDR(X,Y) [SNOLIB]
18 - REPLACE(STR1,STR2,STR3) [SNOLIB]
11 - REST(PATTERN)
18 - REVERSE(STR) [SNOLIB]
13 - REWIND(FNAME)
11 - RIGHT(NAMING,PATTERN)
4 - RUN (OR RHN) COMMAND PARAMETERS
16 - SIMULATED FUNCTIONS ON SNOLIB
5 - SNOBOLC CONTROL CARD PARAMETERS
3 - SNOBOL SUBSYSTEM (CAL SNOBOL)
5 - SNOBOL UNDER THE BATCH SUBSYSTEM (CAL SNOBOL)
17 - SNOLIB FUNCTIONS AND THEIR PARAMETERS
22 - SNOLIB NOTES
19 - SORTI(STRING,DELIMITER,PROTECTOR) [SNOLIB]
19 - SORT0(DELETE) [SNOLIB]
11 - STAR(RIGHT)
18 - STATUS(MESG,FILE) [SNOLIB]
14 - STCOUNT()
14 - STLIMIT(I)
18 - STOPTR(VAR) [SNOLIB]
8 - SUCCEED
18 - SUCCEED (SNOLIB)
18 - TABLE(VAR,CHAR) [SNOLIB]
3 - TELex
3 - TIMESHARING INFORMATION
18 - TRACE(VAR,TAG,FILE) [SNOLIB]
7 - TTY CHARACTER SET [ U OF M ]
11 - TYPE(X)
8 - UNEVALUATED EXPRESSIONS (SEE ALSO KNOWN BUGS)
14 - U OF M MODIFICATIONS TO CAL SNOBOL

--- PAGE 24 ---