NBSIR 78-1420-2

NBS Minimal BASIC Test Programs - Version 1 User's Manual

Volume 2 - General Program Structure, Output, Assignment, Simple Control Structures, Simple Expressions

David E. Gilsinn
Charles L. Sheppard

Systems and Software Division
Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

U.S. DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS

NBSIR 78-1420-2

NBS MINIMAL BASIC TEST PROGRAMS - VERSION 1 USER'S MANUAL

Volume 2 - General Program Structure, Output, Assignment, Simple Control Structures, Simple Expressions

David E. Gilsinn
Charles L. Sheppard

Systems and Software Division
Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

U.S. DEPARTMENT OF COMMERCE, Juanita M. Kreps, *Secretary*

Dr. Sidney Harman, *Under Secretary*

Jordan J. Baruch, *Assistant Secretary for Science and Technology*

NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Acting Director*

ABSTRACT

This volume is the second of four volumes that comprise the user's guide to the NBS Minimal BASIC test programs. The programs test whether a BASIC processor accepts the syntactical forms and produces semantically meaningful results according to the specifications given in BSR X3.60 Proposed American National Standard for Minimal BASIC. The object of this volume is to introduce elementary variable, control and expression forms and exercise them by using a large sample of variations of the appropriate statements. There are thirty three programs in this volume. They cover specifically the tests for output, simple assignment of variables, elementary control structures, general program structure, formulating simple expressions, and the hierarchy of operators. The entire set of test programs is available on magnetic tape.

Key Words: BASIC, BASIC standard, BASIC validation, compiler validation, computer programming language, computer standards.

TABLE OF CONTENTS

0.0 Introduction

This volume is the second in a set of four volumes that comprise the user's guide to the NBS Minimal BASIC test programs. There are thirty three routines in this volume that cover the topics of output, simple assignment of variables, some preliminary control structures, general program structure, formulating simple expressions and determining the hierarchy of operators. The user should be familiar with volume 1 of this series before executing the programs because it describes the general system logic and environmental assumptions made by the authors. Furthermore, in order to understand the terminology, the user should also be familiar with the Minimal BASIC standard BSR X3.60.

Except for routines that concentrate on evaluating processor diagnostic capabilities, most routines have more than one individual test within them. These tests are delineated within the body of the routine by a BEGIN TEST statement on input and are terminated by an END TEST statement on output. Each of the individual tests has been titled with a section number in its program listing that identifies it with the appropriate documentation section.

The authors, in using the tests themselves, have found it useful to proceed by first reading the test documentation paragraphs and noting the appropriate sections within the standard that are covered by them. This correspondence can be found in the overview volume. Next execute a program file and compare the results with the sample output. Then, proceed with the next routine. This allows orderly progression from one test to the next, with the user keeping track of discrepancies as they occur.

There is one source code for each major section in the documentation (e.g. 1.0, 2.0, etc.). The subsections (e.g. 1.2.1, 2.2, etc.) are descriptive paragraphs of specific tests within the one source code associated with the major section. For example, there is a portion of the code NBS FILE 1 that is subtitled with a subsection number 1.2.1. This means that portion of the code is associated with subsection 1.2.1 in the documention.

Within this and the subsequent volumes we have used the word "type" not in the sense of a variable type but in the sense of a difference in form. Thus we speak of one type of numerical constant over another in order to differentiate say between an integer or "E" format numerical constant. Our use of the word "type" does not imply different compiler representation.

1.0 OUTPUT AND ASSIGNMENT OF STRINGS

This test unit executes some of the basic print-list forms which use such print-items as quoted strings, string variables, tab-calls, and nulls. The test unit also executes the case of an empty-print-list. This section uses specifications from sections 3, 4, 5, 6, 9, and 12 of the Minimal BASIC standard X3.60.

1.1 An Empty Print List

The objective of this test is to verify that a completely empty print-list will generate an end-of-print-line, thereby completing the current line of output. That is, if a PRINT statement contains no print list, then a blank line results. The test generates the empty print-list by following the keyword PRINT with an end-of-line character. The output informs the user where the blank lines should be.

1.2 Printing and Assigning Quoted Strings

1.2.1 Allowed ASCII Characters

The objective of this test is to print the character set for BASIC which is described in terms of the ASCII character set. The letters are the set of upper-case Roman letters referenced by the ASCII character set in positions 4/1 through 5/10. The digits are the set of Arabic digits referenced by the ASCII character set in positions 3/0 through 3/9. The remaining string-characters consist of: space, exclamation-point, quote, number-sign, dollar, percent, ampersand, apostrophe, open-parenthesis, close-parenthesis, asterisk, plus, comma, minus, period, slant, colon, semicolon, less-than, equals, greater-than, question-mark, circumflex, and underline. The output should show the letters first, then the digits and finally the remaining quoted string-characters.

1.2.2 Spaces in Quoted Strings

The objective of this test is to show that an implementation will not ignore spaces embedded in quoted strings. The output of the test illustrates this by two lines of alphabetically ordered letters. As a comparison to show that the embedded spaces were retained, the second line has blank spaces where the letters B, E, F, I, J, K, N, O, P, Q, T, U, V, W, and X should be.

1.2.3 End-of-Line Test for Quoted Strings

This test determines whether omitting the print-separator at the end of a print-list will be recognized as an indicator to start the next-line. That is, an end- of-print-line should be generated and added to the characters generated by the evaluation of the print-list. The test is constructed to inform the user how each printed statement should begin and which lines should be blank as a result of an empty string or a null-print.

1.2.4 Semicolon Separator for Quoted Strings

The objective of this test is to verify that the evaluation of the semicolon separator generates the null string (i.e. a string of zero length). The test generates as output the digits "123" on five successive lines. To generate the five lines, the test is constructed to perform:

(1) one print item followed by no separator,
(2) three print items separated by semicolons on one line,
(3) two print items separated by a semicolon on different lines,
(4) two print items separated by a semicolon but on different lines, with different string lengths from (3),
(5) three separate print statements separated by semicolons.

1.2.5 Printing 72 Characters

The objective of this test is to verify that up to 72 characters may be printed on one line. Since the total length of program lines, including the keyword, has been limited to 72 characters, it is not possible to create a single 72-character output string. Therefore, the test uses the semicolon separator to concatenate two sets of two strings each. The first string of each set has 50 characters and the second has 22 characters. On output the two lines of digits form column indices from 1 to 72. This test, of course, assumes the availability of a 72 column output device or terminal.

1.2.6 Comma Separator for Quoted Strings

The objective of this test is to verify that comma separators generate enough spaces to fill out the current print zone. The length of each zone is implementation dependent but should have a width of at least d+e+6 spaces, where d is the significance-width, and e is the exrad-width (see section 12.4 of BSR X3.60). All the print zones should have the same length, except possibly the last. The number of zones and length is implementation dependent. The test is constructed to output three sets of three letters XYZ which should appear in a consistently spaced periodic manner. Each triplet of letters should be separated by spaces and appear in separate print zones. The numbers printed above the letters specify print columns, and the difference between the column number of one of the letters, say X, and its next appearance on the line is the zone width. This number should be greater than or equal to d+e+6. The difference between the column number of the second and third X's should be the same as the first zone width. Three groups were used in order not to exceed the 72 character maximum output line for some implementations.

1.2.7 Tabbing Quoted Strings Within the Margin

The objective of this test is to show that TAB-calling will set the columnar position of the current line to the specified value prior to printing the next print-item. The TAB-calls are not made beyond the margin in this test. The term "current line" refers to the (possibly empty) string of characters generated since the last end-of-print-line was generated. The "margin" is the number of characters that can be printed on one line and is defined by the implementation. The "columnar position" of the current line is the print position that will be occupied by the next character output to that line. Print positions are numbered consecutively from left to right,

starting with position one. The test uses the semicolon print-separator in the print-list to effect the generating of the null string. The test output begins by printing a string of digits for counting the columnar positions. This string of digits should then be followed by the printing of the digits 1, 2, and 3 on three different consecutive lines. Each of the digits should be printed in columns 24, 48, and 72, respectively.

1.2.8 Assignment of Quoted Strings

The objective of this test is to verify the assignment of strings to string variables by assignments of the maximum assignable string length for a standard conforming program (18 characters). The test prints a phrase of 18 characters in length as a verification that the assignment of the maximum assignable string length in a standard conforming program is possible for all string-variables.

1.2.9 Semicolon Separator for Assigned Strings

This test is similar to test 1.2.4. For this test the print-items in the print-list are assigned to string-variables. The output should be the same as that in test 1.2.4.

1.2.10 Comma Separator for Assigned Strings

This test is similar to test 1.2.6. For this test the print-items in the print-list are assigned to string-variables. The output should be the same as that in test 1.2.6.

1.2.11 Tabbing Assigned Strings Within the Margin

This test is similar to test 1.2.7. For this test the print-item following a TAB-call in a print-list is a string-variable. The output is the same as that of test 1.2.7.

```
******************
* PROGRAM FILE 1 *
******************
```

```
10 PRINT "PROGRAM FILE 1"
20 PRINT
30 PRINT
40 PRINT
50 PRINT
60 PRINT "                  * * * * * * * * * * * * * * * * * * *"
70 PRINT "                  *                                   *"
```

```
80 PRINT "                *   MINIMAL BASIC TEST PROGRAM    *"
90 PRINT "                *         PREPARED BY             *"
100 PRINT "                *  NATIONAL BUREAU OF STANDARDS   *"
110 PRINT "                *           VERSION 1             *"
120 PRINT "                *                                 *"
130 PRINT "                * * * * * * * * * * * * * * * * * *"
140 PRINT
150 PRINT
160 PRINT
220 PRINT "                      SECTION 1.1: NULL PRINT."
230 PRINT
240 PRINT "                                BEGIN TEST."
250 PRINT "           THIS IS LINE 1."
260 PRINT "           THIS IS LINE 2."
270 PRINT
280 PRINT "           THIS IS LINE 4, 3 SHOULD HAVE BEEN SKIPPED."
290 PRINT "           THIS IS LINE 5."
300 PRINT
310 PRINT
320 PRINT "         THIS IS LINE 8, 6 AND 7 SHOULD HAVE BEEN SKIPPED."
330 PRINT "                                END TEST."
340 PRINT
350 PRINT "   SECTION 1.2: PRINT OF AND ASSIGNMENT OF QUOTED STRINGS."
360 PRINT
370 PRINT "                  SECTION 1.2.1: ALLOWED CHARACTERS."
380 PRINT
390 PRINT "                                BEGIN TEST."
400 PRINT
410 PRINT "                       ABCDEFGHIJKLMNOPQRSTUVWXYZ"
420 PRINT "                              1234567890"
430 PRINT "                              ()&!# %'?*$"
440 PRINT "                              +,-/<=>^."
450 PRINT "                                 : ;_"
460 PRINT
470 PRINT
480 PRINT "                                 END TEST."
490 PRINT
500 PRINT "               SECTION 1.2.2: SPACES IN QUOTED STRINGS."
510 PRINT
520 PRINT "                                BEGIN TEST."
530 PRINT
540 PRINT "                       ABCDEFGHIJKLMNOPQRSTUVWXYZ"
550 PRINT "                       A CD  GH   LM    RS     YZ"
560 PRINT
570 PRINT "                                 END TEST."
580 PRINT
590 PRINT "         SECTION 1.2.3.: END-OF-LINE TEST, QUOTED STRINGS."
600 PRINT
610 PRINT "                                BEGIN TEST."
620 PRINT
630 PRINT "          THIS LINE COMES FROM ONE QUOTED STRING PRINT."
640 PRINT "          THIS LINE IS THE OUTPUT OF THE NEXT PRINT."
650 PRINT "        PAST TWO PRINT LINES MUST BEGIN WITH:  THIS LINE..."
660 PRINT " "
670 PRINT "          LINE ABOVE THIS MUST BE BLANK (AN EMPTY STRING)."
680 PRINT "          LINE BELOW THIS MUST BE BLANK (NULL PRINT)."
690 PRINT
```

```
700 PRINT "                                   END TEST."
710 PRINT
720 PRINT "        SECTION 1.2.4: SEMICOLON SEPARATOR, QUOTED STRINGS."
730 PRINT
740 PRINT "                                  BEGIN TEST."
750 PRINT
760 PRINT "                                   1. 123"
770 PRINT "                                   2. 1";"2";"3"
780 PRINT "                                   3. 1";
790 PRINT "23"
800 PRINT "                                   4. 12";
810 PRINT "3"
820 PRINT "                                   5. 1";
830 PRINT "2";
840 PRINT "3"
850 PRINT
860 PRINT "                                   END TEST."
870 PRINT
880 PRINT "                     SECTION 1.2.5: 72 CHARACTER PRINT."
890 PRINT
900 PRINT "                                  BEGIN TEST."
910 PRINT
920 PRINT "000000000111111111122222222223333333333444444444445";
930 PRINT "5555555556666666666777"
940 PRINT "123456789012345678901234567890123456789012345678901234567890";
950 PRINT "12345678901234567890"
960 PRINT
970 PRINT "                                   END TEST."
980 PRINT
990 PRINT "            SECTION 1.2.6: COMMA SEPARATOR, QUOTED STRINGS."
1000 PRINT
1010 PRINT "                                  BEGIN TEST."
1020 PRINT
1030 PRINT "000000000111111111122222222223333333333444444444445";
1040 PRINT "5555555556666666666777"
1050 PRINT "123456789012345678901234567890123456789012345678901234567890";
1060 PRINT "12345678901234567890"
1070 PRINT "XYZ","XYZ","XYZ"
1080 PRINT
1090 PRINT "                                    END TEST."
1100 PRINT
1110 PRINT
1120 PRINT " SECTION 1.2.7: TABBING QUOTED STRINGS WITHIN 72 CHARACTER"
1130 PRINT "                POSITIONS."
1140 PRINT
1150 PRINT "                                  BEGIN TEST."
1160 PRINT
1170 PRINT "000000000111111111122222222223333333333444444444445";
1180 PRINT "5555555556666666666777"
1190 PRINT "123456789012345678901234567890123456789012345678901234567890";
1200 PRINT "12345678901234567890"
1210 PRINT TAB(24);"1"
1220 PRINT TAB(48);"2"
1230 PRINT TAB(72);"3"
1240 PRINT
1250 PRINT "                                    END TEST."
1260 PRINT
```

```
1270 PRINT "             SECTION 1.2.8: ASSIGNMENT OF QUOTED STRINGS."
1280 PRINT
1290 PRINT "                                  BEGIN TEST."
1300 PRINT
1310 LET A$="18 CHARACTERS LONG"
1320 LET B$=A$
1330 LET C$=B$
1340 LET D$=C$
1350 LET E$=D$
1360 LET F$=E$
1370 LET G$=F$
1380 LET H$=G$
1390 LET I$=H$
1400 LET J$=I$
1410 LET K$=J$
1420 LET L$=K$
1430 LET M$=L$
1440 LET N$=M$
1450 LET O$=N$
1460 LET P$=O$
1470 LET Q$=P$
1480 LET R$=Q$
1490 LET S$=R$
1500 LET T$=S$
1510 LET U$=T$
1520 LET V$=U$
1530 LET W$=V$
1540 LET X$=W$
1550 LET Y$=X$
1560 LET Z$=Y$
1570 PRINT "                              ";Z$
1580 PRINT
1590 PRINT "                                   END TEST."
1600 PRINT
1610 PRINT "      SECTION 1.2.9: SEMICOLON SEPARATOR, ASSIGNED STRINGS."
1620 PRINT
1630 PRINT "                                  BEGIN TEST."
1640 PRINT
1650 LET A$="123"
1660 LET B$="1"
1670 LET C$="2"
1680 LET D$="3"
1690 LET E$="12"
1700 LET F$="23"
1710 PRINT "                                  1.";A$
1720 PRINT "                                  2.";B$;C$;D$
1730 PRINT "                                  3.";B$;
1740 PRINT F$
1750 PRINT "                                  4.";E$;
1760 PRINT D$
1770 PRINT "                                  5.";B$;
1780 PRINT C$;
1790 PRINT D$
1800 PRINT
1810 PRINT "                                   END TEST."
1820 PRINT
1830 PRINT "         SECTION 1.2.10: COMMA SEPARATOR, ASSIGNED STRINGS."
```

```
1840 PRINT
1850 PRINT "                                   BEGIN TEST."
1860 PRINT
1870 LET A$="XYZ"
1880 LET B$="XYZ"
1890 LET C$="XYZ"
1900 PRINT "0000000001111111111222222222233333333334444444445";
1910 PRINT "55555555566666666777"
1920 PRINT "12345678901234567890123456789012345678901234567890";
1930 PRINT "123456789012345678912"
1940 PRINT A$,B$,C$
1950 PRINT
1960 PRINT "                                    END TEST."
1970 PRINT
1980 PRINT "SECTION 1.2.11: TABBING ASSIGNED STRINGS WITHIN 72 CHARAC-"
1990 PRINT "                TER POSITIONS."
2000 PRINT
2010 PRINT "                                   BEGIN TEST."
2020 PRINT
2030 LET A$="1"
2040 LET B$="2"
2050 LET C$="3"
2060 PRINT "0000000001111111111222222222233333333334444444445";
2070 PRINT "55555555566666666777"
2080 PRINT "12345678901234567890123456789012345678901234567890";
2090 PRINT "123456789012345678912"
2100 PRINT TAB(24);A$
2110 PRINT TAB(48);B$
2120 PRINT TAB(72);C$
2130 PRINT
2140 PRINT "                                    END TEST."
2150 PRINT
2160 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 1

```
* * * * * * * * * * * * * * * * * * *
*                                     *
*   MINIMAL BASIC TEST PROGRAM        *
*          PREPARED BY                *
*  NATIONAL BUREAU OF STANDARDS       *
*            VERSION 1                *
*                                     *
* * * * * * * * * * * * * * * * * * *
```

SECTION 1.1: NULL PRINT.

BEGIN TEST.

THIS IS LINE 1.
THIS IS LINE 2.

THIS IS LINE 4, 3 SHOULD HAVE BEEN SKIPPED.
THIS IS LINE 5.

THIS IS LINE 8, 6 AND 7 SHOULD HAVE BEEN SKIPPED.
END TEST.

SECTION 1.2: PRINT OF AND ASSIGNMENT OF QUOTED STRINGS.

SECTION 1.2.1: ALLOWED CHARACTERS.

BEGIN TEST.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890
()&!# %'?*$
+,-/<=>^.
: ;_

END TEST.

SECTION 1.2.2: SPACES IN QUOTED STRINGS.

BEGIN TEST.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
A CD GH LM RS YZ

END TEST.

SECTION 1.2.3.: END-OF-LINE TEST, QUOTED STRINGS.

BEGIN TEST.

THIS LINE COMES FROM ONE QUOTED STRING PRINT.
THIS LINE IS THE OUTPUT OF THE NEXT PRINT.
PAST TWO PRINT LINES MUST BEGIN WITH: THIS LINE...

LINE ABOVE THIS MUST BE BLANK (AN EMPTY STRING).
LINE BELOW THIS MUST BE BLANK (NULL PRINT).

END TEST.

SECTION 1.2.4: SEMICOLON SEPARATOR, QUOTED STRINGS.

BEGIN TEST.

1. 123
2. 123
3. 123

4. 123
5. 123

END TEST.

SECTION 1.2.5: 72 CHARACTER PRINT.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666667777
123456789012345678901234567890123456789012345678901234567890123456789012
```

END TEST.

SECTION 1.2.6: COMMA SEPARATOR, QUOTED STRINGS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666667777
123456789012345678901234567890123456789012345678901234567890123456789012
XYZ           XYZ           XYZ
```

END TEST.

SECTION 1.2.7: TABBING QUOTED STRINGS WITHIN 72 CHARACTER POSITIONS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666667777
123456789012345678901234567890123456789012345678901234567890123456789012
                        1
                                                2
                                                                        3
```

END TEST.

SECTION 1.2.8: ASSIGNMENT OF QUOTED STRINGS.

BEGIN TEST.

18 CHARACTERS LONG

END TEST.

SECTION 1.2.9: SEMICOLON SEPARATOR, ASSIGNED STRINGS.

BEGIN TEST.

1.123
2.123
3.123
4.123
5.123

END TEST.

SECTION 1.2.10: COMMA SEPARATOR, ASSIGNED STRINGS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
XYZ           XYZ           XYZ
```

END TEST.

SECTION 1.2.11: TABBING ASSIGNED STRINGS WITHIN 72 CHARACTER POSITIONS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
                        1
                                                2
                                                                        3
```

END TEST.

2.0 EXCEPTION TEST FOR PRINTING TAB BEYOND THE LEFT MARGIN

The objective of this routine is to execute error tests on improper numeric arguments for the TAB-call (see section 12.5 of BSR X3.60). This testing is performed as follows:

2.1 TAB Argument is 0

First, it verifies that when the rounded argument of the TAB-call is less than one, the implementation supplies an argument value of one and continues program execution. This part of the test uses an argument value of zero in the TAB-call. On output there should be an X in the first column position.

2.2 TAB Argument is Negative

This part of the test further verifies the procedure specified in test 2.1. However, for this test -10 is the argument of the TAB-call. The output should be the same as in 2.1.

```
******************
* PROGRAM FILE 2 *
******************
```

```
10 PRINT "PROGRAM FILE 2"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                SECTION 2.0: ERROR TEST ON PRINT TAB."
100 PRINT
110 PRINT
120 PRINT
130 PRINT "     THIS SECTION USES TWO DIFFERENT TAB ARGUMENTS TO TEST"
140 PRINT "THE CASE WHEN A ROUNDED TAB ARGUMENT HAPPENS TO HAVE A VALUE"
150 PRINT "LESS THAN ONE.  IN SUCH A CASE THIS SHOULD BE HANDLED AS A"
160 PRINT "NON-FATAL ERROR WHEREBY A VALUE OF 1 IS TO BE SUPPLIED FOR"
170 PRINT "THE TAB ARGUMENT AND THE PROGRAM SHOULD CONTINUE."
180 PRINT
190 PRINT
200 PRINT
210 PRINT "                  SECTION 2.1: TAB ARGUMENT IS 0."
220 PRINT
230 PRINT "                               BEGIN TEST."
240 PRINT
250 PRINT "0000000001111111111222222222233333333334444444445";
```

```
260 PRINT "555555555666666666777"
270 PRINT "12345678901234567890123456789012345678901234567890";
280 PRINT "123456789012345678901"
290 PRINT TAB(0);"X"
300 PRINT
310 PRINT "     IF AN X LIES IN COLUMN 1, TEST PASSED."
330 PRINT
340 PRINT "                                 END TEST."
350 PRINT
360 PRINT
370 PRINT
380 PRINT "                 SECTION 2.2: TAB ARGUMENT IS NEGATIVE."
390 PRINT
400 PRINT "                                 BEGIN TEST."
410 PRINT
420 PRINT "00000000011111111112222222222333333333344444444445";
430 PRINT "555555555666666666777"
440 PRINT "12345678901234567890123456789012345678901234567890";
450 PRINT "123456789012345678901"
460 PRINT TAB(-10);"X"
470 PRINT
480 PRINT "     IF AN X LIES IN COLUMN 1, TEST PASSED."
500 PRINT
510 PRINT "                                 END TEST."
520 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

PROGRAM FILE 2

SECTION 2.0: ERROR TEST ON PRINT TAB.

THIS SECTION USES TWO DIFFERENT TAB ARGUMENTS TO TEST THE CASE WHEN A ROUNDED TAB ARGUMENT HAPPENS TO HAVE A VALUE LESS THAN ONE. IN SUCH A CASE THIS SHOULD BE HANDLED AS A NON-FATAL ERROR WHEREBY A VALUE OF 1 IS TO BE SUPPLIED FOR THE TAB ARGUMENT AND THE PROGRAM SHOULD CONTINUE.

SECTION 2.1: TAB ARGUMENT IS 0.

BEGIN TEST.

```
000000000011111111112222222222333333333344444444445555555555666666666677712345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012X
```

IF AN X LIES IN COLUMN 1, TEST PASSED.

END TEST.

SECTION 2.2: TAB ARGUMENT IS NEGATIVE.

BEGIN TEST.

```
000000000011111111112222222222333333333344444444445555555555666666666677712345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012X
```

IF AN X LIES IN COLUMN 1, TEST PASSED.

END TEST.

3.0 USING EMPTY PRINT ITEMS TO SPACE OVER PRINT ZONES

The objective of this test is to verify that the comma separator allows users to produce automatically positioned output in a tabular format as determined by the print zones (see section 12.4 of BSR X3.60). Furthermore, it verifies that null print items can be used in the print-list. The term "null" refers to the space character. The test uses a print-list which has three single space characters and a quoted string as print-items. Each print-item is separated by a comma print-separator. The output for this test should show an A in the first column position of the fourth print zone assuming the implementation has four print zones. This is a reasonable assumption for most systems.

```
******************
* PROGRAM FILE 3 *
******************
```

```
10 PRINT "PROGRAM FILE 3"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                    SECTION 3.0: NULL PRINT ITEMS."
100 PRINT
110 PRINT
120 PRINT
130 PRINT "*****NOTE:  THE FIRST THREE PRINT ITEMS ARE NULL, THEREFORE"
140 PRINT "THE FOURTH PRINT ITEM WHICH IS A STRING EXPRESSION SHOULD"
150 PRINT "APPEAR IN THE BEGINNING COLUMN OF THE FOURTH PRINT ZONE FOR"
160 PRINT "THIS SYSTEM (THAT IS, IF THE SYSTEM PARTITIONS FOUR OR MORE"
170 PRINT "ZONES TO A LINE.)*****"
180 PRINT
190 PRINT
200 PRINT
210 PRINT "                                BEGIN TEST."
220 PRINT
230 PRINT "00000000011111111112222222222333333333344444444445";
240 PRINT "55555555566666666777"
250 PRINT "12345678901234567890123456789012345678901234567890";
260 PRINT "1234567890123456789012"
270 PRINT  , , ,"A"
280 PRINT
290 PRINT "     IF THE A IS IN THE BEGINNING COLUMN OF THE FOURTH"
300 PRINT "PRINT ZONE FOR THIS SYSTEM, THEN THE SYSTEM WILL HAVE PASSED"
310 PRINT "THE TEST."
320 PRINT
330 PRINT "                                  END TEST."
340 PRINT
```

```
350 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 3

SECTION 3.0: NULL PRINT ITEMS.

*****NOTE: THE FIRST THREE PRINT ITEMS ARE NULL, THEREFORE THE FOURTH PRINT ITEM WHICH IS A STRING EXPRESSION SHOULD APPEAR IN THE BEGINNING COLUMN OF THE FOURTH PRINT ZONE FOR THIS SYSTEM (THAT IS, IF THE SYSTEM PARTITIONS FOUR OR MORE ZONES TO A LINE.)*****

BEGIN TEST.

```
0000000001111111111222222222233333333334444444444555555555566666666667777
1234567890123456789012345678901234567890123456789012345678901234567890123
                                                A
```

IF THE A IS IN THE BEGINNING COLUMN OF THE FOURTH PRINT ZONE FOR THIS SYSTEM, THEN THE SYSTEM WILL HAVE PASSED THE TEST.

END TEST.

4.0 PRINTING INTEGER AND FIXED POINT CONSTANTS

4.1 Printing Constants with Six Significant Digits

The Minimal BASIC standard specifies that a program can contain numeric constants that have an arbitrary number of digits. However, the specification allows implementations to round the values of the constants to an implementation-defined precision of not less than six significant decimal digits. Therefore in general we have restricted our concern to the proper handling of numbers with six significant digits. The user is referred to section 5 and 12 of BSR X3.60 for the required specifications.

In order to clarify some of the notation used we take NR1 to mean an integer constant or implicit point representation form and NR2 to mean a fixed point form or explicit point unscaled representation. We must make it clear that the standard does not specify internal machine representation so that NR1 and NR2 refer only to external character appearance or graphical appearance and not to internal machine representation.

4.1.1 Constants in NR1 Form (Integers)

This section verifies that the test implementation recognizes and prints numbers that appear in the integer or NR1 format. In recognizing the NR1 format, the particular implementation should maintain at least six significant digits of precision on output. The test uses columnar output to show, in the first column, how the numbers should appear according to standard form, and in the second column, how the numbers are printed by the system being tested. The print-lists, in the source code, which contain the print-items for the numerical output use both signed and unsigned NR1 constants.

4.1.2 NR1 Constants Separated by Commas

The objective of this test is similar to that of test 1.2.6. However, for this test, the print-items in the print-list are numerical constants in NR1 form. On output there is a string of digits used as column counters. This string of digits is followed by two lines of integers. The integers on the first line are 1, -12, and 123 in their respective print zones. The second line of integers contains -1234, 12345, and -123456 in their respective print zones.

4.1.3 Space Allotment for Integers, in Compact Form

The objective of this test is to verify that there is a sufficient amount of spacing upon output of NR1 constants. That is, the evaluation of each NR1 constant should produce a string of characters consisting of a leading space if the number is positive or a leading minus sign if the number is negative, followed by the decimal representation of the number and a trailing space. This implies that d+2 spaces are the minimal required for printing NR1 constants. The "d" is the significance-width for the number of significant decimal digits printed in numeric representations. For NR1 constants "d" is required to have a minimal value of at least six digits.

The output of the test generates four columns of NR1 constants. Each print-list, contains numbers to be printed, and quoted strings containing asterisks. These strings are separated from the constants by semicolons and are used to delineate spacing generated around the numeric output. Since semicolons generate no spaces, the four columns of numbers should be surrounded by asterisks to identify the number of spaces allocated for an NR1 representation by the implementation. The output is headed by strings of digits that can be used as column counters. Beneath these strings of digits should be the four columns of numbers. The first column is an example of standard conformance for the implementation's second column of numbers. The third column is an example of standard conformance for the fourth column of numbers which are printed by the implementation.

4.1.4 NR1 Constants Separated by Semicolons

The objective here is two-fold: the test should show that the evaluation of the semicolon separator generates the null string, and at the same time, the test should show on output a minimum of two spaces between each pair of numbers. On output there should be a string of digits which can be used as column counters in determining whether the printed numbers have a minimum of two spaces between each pair.

4.1.5 Constants in NR2 Form (Reals)

There are two conditions under which numbers are to be printed in NR2 format. Let d be the number of significant digits of accuracy of the machine. Then, the first condition arises when the number is not representable as an integer, i.e., NR1, yet falls in the interval

$$0.1-(0.5)*(10\hat{}(-d-1)) \quad \text{to} \quad (10\hat{}d)-(1/2).$$

Notice that as d increases both endpoints increase. The left hand endpoint is never greater than 0.1, however. Next, since the minimal precision requirement is d=6, then (10^6)-(1/2) is the smallest right hand endpoint. Therefore, all numbers on all implementations that are not representable as integers, yet fall between 0.1 and 999999 in absolute value, must be printed in NR2 format. Notice that with systems having six digits of precision one can only get numbers with fractional parts up to 99999.9 without rounding. All numbers from 100000 to 999999 should be rounded to the nearest integer and output in NR1 if only 6 digits of precision are being carried by the system.

The second case occurs when a number is less than $0.1-(0.5)*(10\hat{}(-d-1))$ in absolute value yet can be represented exactly with d digits after the decimal point. For example, if d=6 then 10^(-6) must be printed as .000001.

The actual test is divided into two parts. In the first part a set of numbers, both signed and unsigned, with absolute values between 0.1 and 99999.9 are printed for testing the NR2 printing format. The numbers printed should have at least six significant figures. The numbers printed should all be representable exactly without roundoff. Specific roundoff tests will be conducted in a subsequent test. The distribution of numbers is as follows: 1) 14 numbers of absolute value between 0.1 and 1.0 were chosen. Of these, 5 have been entered with trailing zeroes. The outputs should have trailing zeroes deleted. 2) 5 numbers of absolute value between 1.0 and 10000. 3) 10

numbers of absolute value between 10000 and 99999.9. Columns one and three should fall in print zones one and three. These represent the minimal required output formats. Column three should have a leading minus sign, as well as column four. The first and third columns are the standard output forms requiring d+3 characters counting minus sign or leading blank for positive number, period and trailing space. For systems with d>6 the standard numbers as given would be numbers with the 6 characters as specified and d-6 trailing zeroes which would be required to be suppressed. Columns two and four may have more than d+3 spaces allocated. The point of the test is that however many spaces are allocated by the implementation, the minimal requirement is d+3. For example, implementations may choose to line up decimal points for all NR2 numbers and allow more than d spaces on either side of the decimal. This is acceptable. The listing of column two contains numbers that are alternately signed and unsigned positively. The output of column two should be unsigned. The second part of the test outputs numbers less than 0.1-(0.5)*(10^(-7)) that are exactly representable with d digits after the decimal. Since d=6 is the minimal requirement, all numbers chosen are based on this. The test for other values of d will appear in the roundoff test.

4.1.6 NR2 Constants Separated by Commas

The objective of this test is similar to test 1.2.6. However, for this test the print-items in the print-list are numeric constants, in NR2 form. On output there should be a string of digits which can be used as column counters. This string of digits should be followed by two lines of numerals. The numerals on the first line should be: 123456, -99999.9, and 91234.5 in their respective print zones. The second line of numerals should be -1.23456, 89123.4, and -2.34567 in their respective print zones.

4.1.7 Space Allotment for Reals, in Compact Form

The object of this test section is to print a set of numbers in packed form under a column index scheme to assess the conformance with the minimal d+3 character rule for NR2 numbers. As in the test for section 4.1.3, there are four columns of numbers. Columns one and three are minimal conformance numbers for the case d=6. These numbers have 9 characters. Columns two and four which would be generated by the test system might have more spaces allocated for the number than the minimal conformance requirement. The rule in general is that there should be at least d+3 characters allocated. In order to determine the actual number of characters allowed by the system the columns have been bracketed by asterisks. Again, columns two and four are the system generated columns. These are the columns to which the rule must be applied. There must be at least one trailing space for all numbers and a leading space if the number printed is positive.

4.1.8 NR2 Constants Separated by Semicolons

The objective of this test is the same as test 4.1.4. However, for this test NR2 constants are being used as the print-items. The output should be similar in format to that of test 4.1.4 except that explicit point unscaled constants should appear. The user should examine the spacing between numbers in a line. There should be a trailing space for all numbers and a leading space if the number is positive.

```
                    ******************
                    * PROGRAM FILE 4 *
                    ******************
```

```
10 PRINT "PROGRAM FILE 4"
70 PRINT
80 PRINT
100 PRINT
110 PRINT "SECTION 4.1: PRINT/CONSTANTS, WITHIN 6 SIGNIFICANT DIGITS."
120 PRINT
130 PRINT "         SECTION 4.1.1: CONSTANTS IN NR1 FORM (INTEGERS)."
140 PRINT
150 PRINT "                                BEGIN TEST."
160 PRINT
170 PRINT "00000000011111111112222222222333333333344444444445";
180 PRINT "55555555566666666777"
190 PRINT "123456789012345678901234567890123456789012345678890";
200 PRINT "1234567890123456789012"
210 PRINT "STANDARD","TEST SYSTEM","STANDARD","TEST SYSTEM"
220 PRINT " 1",+1,"-1",-1
230 PRINT " 12",12,"-12",-12
240 PRINT " 123",+123,"-123",-123
250 PRINT " 1234",1234,"-1234",-1234
260 PRINT " 12345",+12345,"-12345",-12345
270 PRINT " 123456",123456,"-123456",-123456
280 PRINT " 999999",+999999,"-999999",-999999
290 PRINT
300 PRINT "                                  END TEST."
310 PRINT
320 PRINT "        SECTION 4.1.2: NR1 CONSTANTS SEPARATED BY COMMAS."
330 PRINT
340 PRINT "                                BEGIN TEST."
350 PRINT
360 PRINT "00000000011111111112222222222333333333344444444445";
370 PRINT "55555555566666666777"
380 PRINT "123456789012345678901234567890123456789012345678890";
390 PRINT "1234567890123456789012"
400 PRINT 1,-12,123
410 PRINT -1234,12345,-123456
420 PRINT
430 PRINT "                                  END TEST."
440 PRINT
450 PRINT "SEC. 4.1.3: SPACE ALLOTMENT FOR INTEGERS, IN COMPACT FORM."
460 PRINT
470 PRINT "                                BEGIN TEST."
480 PRINT
490 PRINT "00000000011111111112222222222333333333344444444445";
500 PRINT "55555555566666666777"
510 PRINT "123456789012345678901234567890123456789012345678890";
520 PRINT "1234567890123456789012"
530 PRINT "STANDARD";"  SYSTEM";" STANDARD";"  SYSTEM"
540 PRINT "* 1 ******";1;"*-1 ******";-1;"******"
550 PRINT "* 12 *****";+12;"*-12 *****";-12;"*****"
```

```
560 PRINT "* 123 ****";123;"*-123 ****";-123;"****"
570 PRINT "* 1234 ***";+1234;"*-1234 ***";-1234;"***"
580 PRINT "* 12345 **";12345;"*-12345 **";-12345;"**"
590 PRINT "* 123456 *";+123456;"*-123456 *";-123456;"*"
600 PRINT "* 999999 *";999999;"*-999999 *";-999999;"*"
610 PRINT
620 PRINT "                                   END TEST."
630 PRINT
640 PRINT "      SECTION 4.1.4: NR1 CONSTANTS SEPARATED BY SEMICOLONS."
650 PRINT
660 PRINT "                                   BEGIN TEST."
670 PRINT
680 PRINT "00000000011111111112222222222333333333344444444445";
690 PRINT "5555555556666666666777"
700 PRINT "12345678901234567890123456789012345678901234567890";
710 PRINT "123456789012345678912"
720 PRINT 1;-12;123
730 PRINT -1234;12345;-123456
740 PRINT
750 PRINT "                                   END TEST."
760 PRINT
770 PRINT "           SECTION 4.1.5: CONSTANTS IN NR2 FORM (REALS)."
780 PRINT
790 PRINT "                                   BEGIN TEST."
800 PRINT
810 PRINT
820 PRINT
830 PRINT "00000000011111111112222222222333333333344444444445";
840 PRINT "5555555556666666666777"
850 PRINT "12345678901234567890123456789012345678901234567890";
860 PRINT "123456789012345678912"
870 PRINT "STANDARD","TEST SYSTEM","STANDARD","TEST SYSTEM"
880 PRINT " .1 ",.1,"-.1 ",-.1
890 PRINT " .12 ",+.12,"-.12 ",-.12
900 PRINT " .123 ",.123,"-.123 ",-.123
910 PRINT " .1234 ",+.1234,"-.1234 ",-.1234
920 PRINT " .12345 ",.12345,"-.12345 ",-.12345
930 PRINT " .123456 ",+.123456,"-.123456 ",-.123456
940 PRINT " .234567 ",.234567,"-.234567 ",-.234567
950 PRINT " .345678 ",+.345678,"-.345678 ",-.345678
960 PRINT " .456789 ",.456789,"-.456789 ",-.456789
970 PRINT " .56789 ",+.567890,"-.56789 ",-.567890
980 PRINT " .6789 ",.678900,"-.6789 ",-.678900
990 PRINT " .789 ",+.789000,"-.789 ",-.789000
1000 PRINT " .89 ",.890000,"-.89 ",-.890000
1010 PRINT " .9 ",+.900000,"-.9 ",-.900000
1020 PRINT " 1.23456 ",1.23456,"-1.23456",-1.23456
1030 PRINT " 9.87654 ",+9.87654,"-9.87654 ",-9.87654
1040 PRINT " 12.3456 ",12.3456,"-12.3456 ",-12.3456
1050 PRINT " 123.456 ",+123.456,"-123.456 ",-123.456
1060 PRINT " 1234.56 ",1234.56,"-1234.56 ",-1234.56
1070 PRINT " 12345.6 ",+12345.6,"-12345.6 ",-12345.6
1080 PRINT " 23456.7 ",23456.7,"-23456.7 ",-23456.7
1090 PRINT " 34567.8 ",+34567.8,"-34567.8 ",-34567.8
1100 PRINT " 45678.9 ",45678.9,"-45678.9 ",-45678.9
1110 PRINT " 56789.1 ",+56789.1,"-56789.1 ",-56789.1
1120 PRINT " 67891.2 ",67891.2,"-67891.2 ",-67891.2
```

```
1130 PRINT " 78912.3 ",+78912.3,"-78912.3 ",-78912.3
1140 PRINT " 89123.4 ",89123.4,"-89123.4 ",-89123.4
1150 PRINT " 91234.5 ",+91234.5,"-91234.5 ",-91234.5
1160 PRINT " 99999.9 ",99999.9,"-99999.9 ",-99999.9
1170 PRINT
1180 PRINT "                                END PART I."
1190 PRINT
1200 PRINT
1210 PRINT "                                    PART II"
1220 PRINT
1230 PRINT " .022222 ",+.022222,"-.022222 ",-.022222
1240 PRINT " .004444 ",.004444,"-.004444 ",-.004444
1250 PRINT " .000888 ",+.000888,"-.000888 ",-.000888
1260 PRINT " .000044 ",.000044,"-.000044 ",-.000044
1270 PRINT " .000002 ",+.000002,"-.000002 ",-.000002
1280 PRINT
1290 PRINT
1300 PRINT "                                  END TEST."
1310 PRINT
1340 PRINT "        SECTION 4.1.6: NR2 CONSTANTS SEPARATED BY COMMAS."
1330 PRINT
1340 PRINT "                                BEGIN TEST."
1350 PRINT
1360 PRINT "00000000011111111112222222222333333333344444444445";
1370 PRINT "5555555556666666666777"
1380 PRINT "12345678901234567890123456789012345678901234567890";
1390 PRINT "1234567890123456789012"
1400 PRINT .123456,-99999.9,91234.5
1410 PRINT -1.23456,89123.4,-2.34567
1420 PRINT
1430 PRINT "                                  END TEST."
1440 PRINT
1450 PRINT " SECTION 4.1.7: SPACE ALLOTMENT FOR REALS, IN COMPACT FORM."
1460 PRINT
1470 PRINT "                                BEGIN TEST."
1480 PRINT
1490 PRINT "00000000011111111112222222222333333333344444444445";
1500 PRINT "5555555556666666666777"
1510 PRINT "12345678901234567890123456789012345678901234567890";
1520 PRINT "1234567890123456789012"
1530 PRINT "STANDARD";"   SYSTEM";"  STANDARD";"   SYSTEM"
1540 PRINT "** .234567 **";+.234567;"**-.234567 **";-.234567;"**"
1550 PRINT "** 1.23456 **";1.23456;"**-1.23456 **";-1.23456;"**"
1560 PRINT "** 91234.5 **";+91234.5;"**-91234.5 **";-91234.5;"**"
1570 PRINT "** 99999.9 **";99999.9;"**-99999.9 **";-99999.9;"**"
1580 PRINT
1590 PRINT "                                  END TEST."
1600 PRINT
1610 PRINT "      SECTION 4.1.8: NR2 CONSTANTS SEPARATED BY SEMICOLONS."
1620 PRINT
1630 PRINT "                                BEGIN TEST."
1640 PRINT
1650 PRINT "00000000011111111112222222222333333333344444444445";
1660 PRINT "5555555556666666666777"
1670 PRINT "12345678901234567890123456789012345678901234567890";
1680 PRINT "1234567890123456789012"
1690 PRINT .123456;-99999.9;91234.5
```

```
1700 PRINT -1.23456;89123.4;-2.34567
1710 PRINT
1720 PRINT "                                   END TEST."
1730 PRINT
1740 END
```

****
* SAMPLE OUTPUT *
****

```
PROGRAM FILE 4


SECTION 4.1: PRINT/CONSTANTS, WITHIN 6 SIGNIFICANT DIGITS.

         SECTION 4.1.1: CONSTANTS IN NR1 FORM (INTEGERS).

                                BEGIN TEST.

000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
STANDARD      TEST SYSTEM   STANDARD      TEST SYSTEM
 1             1            -1            -1
 12            12           -12           -12
 123           123          -123          -123
 1234          1234         -1234         -1234
 12345         12345        -12345        -12345
 123456        123456       -123456       -123456
 999999        999999       -999999       -999999

                                 END TEST.

        SECTION 4.1.2: NR1 CONSTANTS SEPARATED BY COMMAS.

                                BEGIN TEST.

000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
 1            -12            123
-1234          12345        -123456

                                 END TEST.

SEC. 4.1.3: SPACE ALLOTMENT FOR INTEGERS, IN COMPACT FORM.

                                BEGIN TEST.

000000000111111111122222222223333333333444444444455555555556666666666777
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
STANDARD  SYSTEM STANDARD  SYSTEM
* 1 ****** 1 *-1 ******-1 ******
* 12 ***** 12 *-12 *****-12 *****
* 123 **** 123 *-123 ****-123 ****
* 1234 *** 1234 *-1234 ***-1234 ***
* 12345 ** 12345 *-12345 **-12345 **
* 123456 * 123456 *-123456 *-123456 *
* 999999 * 999999 *-999999 *-999999 *
```

END TEST.

SECTION 4.1.4: NR1 CONSTANTS SEPARATED BY SEMICOLONS.

BEGIN TEST.

```
000000000011111111112222222222333333333344444444445555555555666666666677777
123456789012345678901234567890123456789012345678901234567890123456789012345678
 1 -12  123
-1234  12345 -123456
```

END TEST.

SECTION 4.1.5: CONSTANTS IN NR2 FORM (REALS).

BEGIN TEST.

```
000000000011111111112222222222333333333344444444445555555555666666666677777
123456789012345678901234567890123456789012345678901234567890123456789012345678
STANDARD      TEST SYSTEM   STANDARD      TEST SYSTEM
 .1            .1           -.1           -.1
 .12           .12          -.12          -.12
 .123          .123         -.123         -.123
 .1234         .1234        -.1234        -.1234
 .12345        .12345       -.12345       -.12345
 .123456       .123456      -.123456      -.123456
 .234567       .234567      -.234567      -.234567
 .345678       .345678      -.345678      -.345678
 .456789       .456789      -.456789      -.456789
 .56789        .56789       -.56789       -.56789
 .6789         .6789        -.6789        -.6789
 .789          .789         -.789         -.789
 .89           .89          -.89          -.89
 .9            .9           -.9           -.9
 1.23456       1.23456      -1.23456      -1.23456
 9.87654       9.87654      -9.87654      -9.87654
 12.3456       12.3456      -12.3456      -12.3456
 123.456       123.456      -123.456      -123.456
 1234.56       1234.56      -1234.56      -1234.56
 12345.6       12345.6      -12345.6      -12345.6
 23456.7       23456.7      -23456.7      -23456.7
 34567.8       34567.8      -34567.8      -34567.8
 45678.9       45678.9      -45678.9      -45678.9
 56789.1       56789.1      -56789.1      -56789.1
 67891.2       67891.2      -67891.2      -67891.2
```

```
 78912.3       78912.3      -78912.3      -78912.3
 89123.4       89123.4      -89123.4      -89123.4
 91234.5       91234.5      -91234.5      -91234.5
 99999.9       99999.9      -99999.9      -99999.9
```

END PART I.

PART II

```
 .022222       .022222      -.022222      -.022222
 .004444       .004444      -.004444      -.004444
 .000888       .000888      -.000888      -.000888
 .000044       .000044      -.000044      -.000044
 .000002       .000002      -.000002      -.000002
```

END TEST.

SECTION 4.1.6: NR2 CONSTANTS SEPARATED BY COMMAS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
 .123456      -99999.9       91234.5
-1.23456       89123.4      -2.34567
```

END TEST.

SECTION 4.1.7: SPACE ALLOTMENT FOR REALS, IN COMPACT FORM.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
STANDARD   SYSTEM  STANDARD   SYSTEM
** .234567 ** .234567 **-.234567 **-.234567 **
** 1.23456 ** 1.23456 **-1.23456 **-1.23456 **
** 91234.5 ** 91234.5 **-91234.5 **-91234.5 **
** 99999.9 ** 99999.9 **-99999.9 **-99999.9 **
```

END TEST.

SECTION 4.1.8: NR2 CONSTANTS SEPARATED BY SEMICOLONS.

BEGIN TEST.

```
000000000111111111122222222223333333333444444444455555555556666666666777
123456789012345678901234567890123456789012345678901234567890123456789012
 .123456 -99999.9  91234.5
-1.23456  89123.4 -2.34567
```

END TEST.

5.0 PRINTING FLOATING POINT CONSTANTS

As in the previous section the words "floating point" refer to the external character appearance, as for example, on input or output. It does not mean that internal representation is required to be floating point, although this is sometimes the case and software is sometimes used to perform the required conversions for input and output formatting. The user should refer to sections 5 and 12 of BSR X3.60 for the specifications appropriate to this section.

5.1 Constants in NR3 Form (E-Format)

The objective of this part is to print constants that require the use of an explicit print scaled representation form. There are two cases in which this should occur. First , if the number has a magnitude greater than (10^d)-(1/2) and secondly, if the number is less than 0.1-(0.5)*(10^(-d-1)) and cannot be expressed exactly with d decimal digits following a period. The minimal upper and maximal lower bounds for the absolute value of numbers are 1.E38 and 1.E-38, respectively. This is the range of the magnitude of numbers that must be recognized by all systems supporting Minimal Basic. Larger ranges containing this range are acceptable. Note that for NR3 representations on output, the significand, represented by x, must lie in the range 1<= x <10 which means that there must be one digit to the left of the decimal point.

The first part of this test subsection will print groups of large numbers. The optional exrad sign will be either absent or a plus, the significand will be either a plus (or optionally blank) or minus. The second part of the test section prints 2 groups of small numbers with negative exrad signs.

On output , if the sign of the significand of any of the numbers is positive then a space is printed. The positive signs to the significand are omitted. Although the sample output maintains trailing zeroes in the significand, the standard allows trailing zeroes to be omitted in the fractional part of the significand. Leading zeroes may be omitted in the exrad. There is to be no spacing allowed in the exrad. The standard does not specify the output sign of the exrad. It can be either a + or no sign, but as indicated above no spaces are allowed in the exrad since this would violate the rule that spaces are not allowed in numeric constants.

The output is displayed as follows: After the test title a standard and an optional form is given based upon d=6 and e=2. Below these initial numbers, two columns of numbers should appear. All the numbers should be equal to each other and be equal to one of the allowed forms. Of course in the case that d is greater than 6 digits, extra zeroes in the significand of the actual system output may appear or be deleted. The first group of columns will have a positive signed significand in both columns. The exrad will be signed in the second column in the source code. The numbers used in the source code are various forms of implcit and explicit print scaled versions of the same number.

In the second set, two columns of numbers as in the first set will be printed. These will be generated by numbers having an unsigned significand and either signed or unsigned exrads.

The group three numbers will be the same as the previous two groups except that a minus sign in front of the significand.

The fourth set will have only one standard form but there will be two columns of output below it. In this case there will be an unsigned significand and negative signed exrad.

Finally the fifth group will have one column of output and one standard form. Both significand and exrad will have negative signs.

In summary, then, for the positive significand and positive exrad there are the following sign combination input cases: signed significand, unsigned exrad; signed significand, signed exrad; unsigned significand, unsigned exrad; unsigned significand, signed exrad. For the positive significand and negative exrad there are only two cases: signed and unsigned significand. For the negative significand and negative exrad there is only one case.

5.2 Printing of Some General NR3 Constants

This test is a continuation of test 5.1, part II. The object here is simply to print a selection of NR3 numbers without attempting to consider the many possible configurations of the same number. Columns one and two are allowed optional graphical representation. Column three contains the values generated by the test system.

5.3 NR3 Constants Separated by Commas

The objective of this test is similar to that of test 1.2.6. In this test, however, the print-items in the print-list are numeric constants in NR3 form. On output there is a string of digits as column counters. This string of digits should be followed by two lines of numbers. The numbers in the first line should be 1.00000E30, -9.87000E-37, and 1.23456E32 in their respective print zones. The second line of numbers should be -1.23456E32, 1.786500E36 and 5.00000E-20 in their respective print zones.

```
******************
* PROGRAM FILE 5 *
******************
```

```
10 PRINT "PROGRAM FILE 5"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "          SECTION 5.1: CONSTANTS IN NR3 FORM (E-FORMAT)."
100 PRINT
110 PRINT "                              BEGIN TEST."
120 PRINT
130 PRINT
140 PRINT "       INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 1)"
150 PRINT
160 PRINT "   ********************************************************"
170 PRINT "   *  INPUT  *     SIGNIFICAND     *        EXRAD          *"
180 PRINT "   ********************************************************"
190 PRINT "   * COLUMNS * PLUS SIGN * NO SIGN * PLUS SIGN * NO SIGN *"
200 PRINT "   ********************************************************"
210 PRINT "   *    1    *    YES    *         *           *   YES    *"
220 PRINT "   *    2    *    YES    *         *    YES    *          *"
230 PRINT "   ********************************************************"
240 PRINT
250 PRINT
260 PRINT "           STANDARD OUTPUT FOR GROUP 1 WHERE (D=6, E=2),"
270 PRINT
280 PRINT "SHOULD BE:"," 1.23456E32 "
290 PRINT
300 PRINT "                                                OR OPTIONALLY,"
310 PRINT
320 PRINT "SHOULD BE:"," 1.23456E+32 "
330 PRINT
340 PRINT "     OUTPUT OF SYSTEM."
350 PRINT
360 PRINT +123456E27,+123456E+27
370 PRINT +123456.E27,+123456.E+27
380 PRINT +12345.6E28,+12345.6E+28
390 PRINT +1234.56E29,+1234.56E+29
400 PRINT +123.456E30,+123.456E+30
410 PRINT +12.3456E31,+12.3456E+31
420 PRINT +1.23456E32,+1.23456E+32
430 PRINT +.123456E33,+.123456E+33
440 PRINT +1.2345600000000E32,+1.2345600000000E+32
450 PRINT +.00000123456E38,+.00000123456E+38
460 PRINT +.00001234560000E37,+.00001234560000E+37
470 PRINT
480 PRINT
490 PRINT "       INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 2)"
500 PRINT
510 PRINT "   ********************************************************"
520 PRINT "   *  INPUT  *     SIGNIFICAND     *        EXRAD          *"
530 PRINT "   ********************************************************"
540 PRINT "   * COLUMNS * PLUS SIGN * NO SIGN * PLUS SIGN * NO SIGN *"
550 PRINT "   ********************************************************"
560 PRINT "   *    1    *           *   YES   *           *   YES    *"
570 PRINT "   *    2    *           *   YES   *    YES    *          *"
580 PRINT "   ********************************************************"
590 PRINT
600 PRINT
610 PRINT "           STANDARD OUTPUT FOR GROUP 2 WHERE (D=6, E=2),"
```

```
620 PRINT
630 PRINT "SHOULD BE:"," 1.23456E32 "
640 PRINT
650 PRINT "                                                OR OPTIONALLY,"
660 PRINT
670 PRINT "SHOULD BE:"," 1.23456E+32 "
680 PRINT
690 PRINT "     OUT OF SYSTEM."
700 PRINT
710 PRINT 123456E27,123456E+27
720 PRINT 123456.E27,123456.E+27
730 PRINT 12345.6E28,12345.6E+28
740 PRINT 1234.56E29,1234.56E+29
750 PRINT 123.456E30,123.456E+30
760 PRINT 12.3456E31,12.3456E+31
770 PRINT 1.23456E32,1.23456E+32
780 PRINT .123456E33,.123456E+33
790 PRINT 1.2345600000000E32,1.2345600000000E+32
800 PRINT .00000123456E38,.00000123456E+38
810 PRINT .00001234560000E37,.00001234560000E+37
820 PRINT
830 PRINT
840 PRINT "       INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 3)"
850 PRINT
860 PRINT "  ************************************************************"
870 PRINT "  *  INPUT  *      SIGNIFICAND     *        EXRAD          *"
880 PRINT "  ************************************************************"
890 PRINT "  * COLUMNS * MINUS SIGN * NO SIGN * PLUS SIGN * NO SIGN *"
900 PRINT "  ************************************************************"
910 PRINT "  *    1    *     YES    *         *            *   YES    *"
920 PRINT "  *    2    *     YES    *         *    YES     *          *"
930 PRINT "  ************************************************************"
940 PRINT
950 PRINT
960 PRINT "           STANDARD OUTPUT FOR GROUP 3 WHERE (D=6, E=2),"
970 PRINT
980 PRINT "SHOULD BE:","-1.23456E32 "
990 PRINT
1000 PRINT "                                                OR OPTIONALLY,"
1010 PRINT
1020 PRINT "SHOULD BE:","-1.23456E+32 "
1030 PRINT
1040 PRINT "     OUTPUT OF SYSTEM."
1050 PRINT
1060 PRINT -123456E27,-123456E+27
1070 PRINT -123456.E27,-123456.E+27
1080 PRINT 0-12345.6E28,-12345.6E+28
1090 PRINT -1234.56E29,-1234.56E+29
1100 PRINT -123.456E30,-123.456E+30
1110 PRINT -12.3456E31,-12.3456E+31
1120 PRINT -1.23456E32,-1.23456E+32
1130 PRINT -.123456E33,-.123456E+33
1140 PRINT -1.2345600000000E32,-1.2345600000000E+32
1150 PRINT -.00000123456E38,-.00000123456E+38
1160 PRINT -.00001234560000E37,-.00001234560000E+37
1170 PRINT
1180 PRINT
```

```
1190 PRINT "       INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 4)"
1200 PRINT "  ************************************************************"
1210 PRINT "  *  INPUT  *      SIGNIFICAND      *          EXRAD          *"
1220 PRINT "  ************************************************************"
1230 PRINT "  * COLUMNS * PLUS SIGN * NO SIGN * MINUS SIGN * NO SIGN *"
1240 PRINT "  ************************************************************"
1250 PRINT "  *    1    *    YES    *          *     YES     *          *"
1260 PRINT "  *    2    *           *   YES    *     YES     *          *"
1270 PRINT "  ************************************************************"
1280 PRINT
1290 PRINT
1300 PRINT "           STANDARD OUTPUT FOR GROUP 4 WHERE (D=6, E=2),"
1310 PRINT
1320 PRINT "SHOULD BE:"," 1.23456E-24 "
1330 PRINT
1340 PRINT "                                          AS FOR OPTIONALS,"
1350 PRINT "NONE."
1360 PRINT
1370 PRINT "     OUTPUT OF SYSTEM."
1380 PRINT
1390 PRINT +123456E-29,123456E-29
1400 PRINT +123456.E-29,123456.E-29
1410 PRINT +12345.6E-28,12345.6E-28
1420 PRINT +1234.56E-27,1234.56E-27
1430 PRINT +123.456E-26,123.456E-26
1440 PRINT +12.3456E-25,12.3456E-25
1450 PRINT +1.23456E-24,1.23456E-24
1460 PRINT +.123456E-23,.123456E-23
1470 PRINT +1.2345600000000E-24,1.2345600000000E-24
1480 PRINT +.00000123456E-18,.00000123456E-18
1490 PRINT +.00001234560000E-19,.00001234560000E-19
1500 PRINT
1510 PRINT
1520 PRINT "       INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 5)"
1530 PRINT
1540 PRINT " ************************************************************"
1550 PRINT " *  INPUT  *      SIGNIFICAND      *          EXRAD          *"
1560 PRINT " ************************************************************"
1570 PRINT " * COLUMNS * MINUS SIGN * NO SIGN * MUNIS SIGN * NO SIGN *"
1580 PRINT " ************************************************************"
1590 PRINT " *    1    *     YES    *          *     YES     *          *"
1600 PRINT " ************************************************************"
1610 PRINT
1620 PRINT
1630 PRINT "           STANDARD OUTPUT FOR GROUP 5 WHERE (D=6, E=2),"
1640 PRINT
1650 PRINT "SHOULD BE:","-1.23456E-24 "
1660 PRINT
1670 PRINT "                                          AS FOR OPTIONALS,"
1680 PRINT
1690 PRINT "NONE."
1700 PRINT
1710 PRINT "     OUTPUT OF SYSTEM."
1720 PRINT
1730 PRINT -123456E-29
1740 PRINT -123456.E-29
1750 PRINT -12345.6E-28
```

```
1760 PRINT -1234.56E-27
1770 PRINT -123.456E-26
1780 PRINT -12.3456E-25
1790 PRINT -1.23456E-24
1800 PRINT -.123456E-23
1810 PRINT -1.2345600000000E-24
1820 PRINT -.00000123456E-18
1830 PRINT -.00001234560000E-19
1840 PRINT
1850 PRINT
1860 PRINT "                                   END TEST."
1870 PRINT
1880 PRINT " SECTION 5.2: EXECUTION OF SOME GENERAL NR3 CONSTANT."
1890 PRINT
1900 PRINT "                                  BEGIN TEST."
1910 PRINT
1920 PRINT
1930 PRINT "STANDARD","OPTIONAL","SYSTEM"
1940 PRINT
1950 PRINT " 1.00000E30 "," 1.00000E+30 ",1E30
1960 PRINT " 5.00000E-20 ","NONE",5E-20
1970 PRINT " 4.25000E-25 ","NONE",+.425E-24
1980 PRINT "-3.60000E25 ","-3.60000E+25 ",-3.6E25
1990 PRINT "-7.89100E35 ","-7.89100E+35 ",-7.891E+35
2000 PRINT " 1.78650E36 "," 1.78650E+36 ",+1.7865E+36
2010 PRINT " 9.76543E37 "," 9.76543E+37 ",9.76543E+37
2020 PRINT " 9.00000E34 "," 9.00000E+37 ",+9E34
2030 PRINT "-9.87000E-37","NONE",-9.87E-37
2040 PRINT
2050 PRINT "                                   END TEST."
2060 PRINT
2070 PRINT "       SECTION 5.3: NR3 CONSTANTS SEPARATED BY COMMAS."
2080 PRINT
2090 PRINT "                                  BEGIN TEST."
2100 PRINT
2110 PRINT "00000000011111111112222222222333333333344444444445";
2120 PRINT "55555555566666666677"
2130 PRINT "12345678901234567890123456789012345678901234567890";
2140 PRINT "123456789012345678912"
2150 PRINT 1E30,-9.87E-37,1.23456E32
2160 PRINT -1.23456E32,+1.7865E36,5E-20
2170 PRINT
2180 PRINT "                                   END TEST."
2190 PRINT
2200 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

PROGRAM FILE 5

SECTION 5.1: CONSTANTS IN NR3 FORM (E-FORMAT).

BEGIN TEST.

INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 1)

INPUT	SIGNIFICAND		EXRAD	
COLUMNS	PLUS SIGN	NO SIGN	PLUS SIGN	NO SIGN
1	YES			YES
2	YES		YES	

STANDARD OUTPUT FOR GROUP 1 WHERE (D=6, E=2),

SHOULD BE: 1.23456E32

OR OPTIONALLY,

SHOULD BE: 1.23456E+32

OUTPUT OF SYSTEM.

```
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
```

INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 2)

INPUT	SIGNIFICAND		EXRAD	
COLUMNS	PLUS SIGN	NO SIGN	PLUS SIGN	NO SIGN
1		YES		YES
2		YES	YES	

STANDARD OUTPUT FOR GROUP 2 WHERE (D=6, E=2),

SHOULD BE: 1.23456E32

OR OPTIONALLY,

SHOULD BE: 1.23456E+32

OUT OF SYSTEM.

```
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
 1.23456E+32   1.23456E+32
```

INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 3)

INPUT	SIGNIFICAND		EXRAD	
COLUMNS	MINUS SIGN	NO SIGN	PLUS SIGN	NO SIGN
1	YES			YES
2	YES		YES	

STANDARD OUTPUT FOR GROUP 3 WHERE (D=6, E=2),

SHOULD BE: -1.23456E32

OR OPTIONALLY,

SHOULD BE: -1.23456E+32

OUTPUT OF SYSTEM.

```
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
-1.23456E+32  -1.23456E+32
```

INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 4)

INPUT	SIGNIFICAND		EXRAD	
COLUMNS	PLUS SIGN	NO SIGN	MINUS SIGN	NO SIGN
1	YES		YES	
2		YES	YES	

STANDARD OUTPUT FOR GROUP 4 WHERE (D=6, E=2),

SHOULD BE: 1.23456E-24

AS FOR OPTIONALS,

NONE.

OUTPUT OF SYSTEM.

```
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
 1.23456E-24   1.23456E-24
```

INPUT FORMS ON SIGNIFICAND AND EXRAD (GROUP 5)

INPUT	SIGNIFICAND		EXRAD	
COLUMNS	MINUS SIGN	NO SIGN	MUNIS SIGN	NO SIGN
1	YES		YES	

STANDARD OUTPUT FOR GROUP 5 WHERE (D=6, E=2),

SHOULD BE: -1.23456E-24

AS FOR OPTIONALS,

NONE.

OUTPUT OF SYSTEM.

```
-1.23456E-24
-1.23456E-24
-1.23456E-24
-1.23456E-24
```

```
-1.23456E-24
-1.23456E-24
-1.23456E-24
-1.23456E-24
-1.23456E-24
-1.23456E-24
-1.23456E-24
```

END TEST.

SECTION 5.2: EXECUTION OF SOME GENERAL NR3 CONSTANT.

BEGIN TEST.

STANDARD	OPTIONAL	SYSTEM
1.00000E30	1.00000E+30	1.00000E+30
5.00000E-20	NONE	5.00000E-20
4.25000E-25	NONE	4.25000E-25
-3.60000E25	-3.60000E+25	-3.60000E+25
-7.89100E35	-7.89100E+35	-7.89100E+35
1.78650E36	1.78650E+36	1.78650E+36
9.76543E37	9.76543E+37	9.76543E+37
9.00000E34	9.00000E+37	9.00000E+34
-9.87000E-37	NONE	-9.87000E-37

END TEST.

SECTION 5.3: NR3 CONSTANTS SEPARATED BY COMMAS.

BEGIN TEST.

```
0000000001111111111222222222233333333334444444444555555555566666666677712345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
 1.00000E+30  -9.87000E-37   1.23456E+32
-1.23456E+32   1.78650E+36   5.00000E-20
```

END TEST.

6.Ø PRINTING OF FLOATING POINT NUMBERS (CONTINUED) AND ASSIGNMENT OF INTEGER AND FIXED POINT VALUES

In this section we complete the task of evaluating the format of the NR3 constants separated by semicolon delimiters on output and begin the task of assigning the previous numeric constant types to simple variables and printing those variables. Again trailing zeroes in the significand are optional. The user is referred to section 5, 6, 9 and 12 of BSR X3.60 for specifications.

6.1 Printing NR3 Numbers with Semicolon Delimiters

6.1.1 Space Allotment (E-Format) in Compact Form

The objective of this test is similar to that of previous semicolon separation tests. The representation of NR3 numbers requires exactly d+e+5 characters. Therefore, several NR3 numbers will be given, bracketed by asterisks to delineate the number of characters allocated for an NR3 number. The characters allow space for at most two signs, a period, the "E", and a trailing space.

6.1.2 NR3 Constants Separated by Semicolons

The objective of this test is the same as that of test 4.1.4. However, for this test NR3 constants are being used as the print items. There should be output similar to that of test 4.1.4 except for the numbers.

6.2 Assignment of Numeric Constants to Simple Variables

Numeric variables are named by either a single letter or a letter and a digit. Numeric variables may be either simple variables, which refer to single values, or they may refer to elements of one or two dimensional arrays. Such references are called subscripted variables. As indicated above simple numeric variables consist of a letter followed by an optional digit. Arrays will not be considered in this set of tests. They will be deferred until after the Dimension statement has been introduced. Variables can also be assigned to other variables. This is the first primitive assignment of numeric expressions beyond assigning constants to variables. There are 286 possible simple numeric variables.

6.2.1 Assignment of NR1 Constants

The objective of this test is to verify that signed as well as unsigned NR1 constants can be assigned to simple variables. It also verifies that an assigned NR1 constant can be assigned from one simple variable to another.

6.2.1.1 Unsigned Constants

The objective of this test is to verify that the positive sign is optional when representing NR1 constants. Therefore, this test determines whether not having a plus sign with a number when it is assigned to a simple variable causes any discrepancy. On output there should be two columns of numbers. The first column of numbers represents the desired numbers, which can be compared with those outputted in the second column as the evaluated representations of the system.

6.2.1.2 Signed Constants

The objective of this test is the same as that of test 6.2.1.1 except that this test uses signed NR1 constants assigned to the simple variables. The test has an output format similar to that of test 6.2.1.1.

6.2.1.3 Cross Assignment

The objective of this test is to determine whether repeated assignments of an NR1 constant between several different simple variables will modify the value of that constant, thus verifying another phase of assigning numeric expressions to simple variables. The test uses 26 of the simple variables, A through Z. The simple variable A is initially assigned an NR1 constant and then the content of A is assigned to B, and then B to C, and then C to D; and this sequence is con tinued through Z. The output represents the value of Z which should be the value of the original NR1 constant assigned to A.

6.2.2 Assignment of NR2 Constants

The objective of this test is to verify that signed as well as unsigned NR2 constants can be assigned to simple variables. Also, it verifies whether an assigned NR2 can be assigned from one simple variable to another.

6.2.2.1 Unsigned Constants

The objective of this test is to verify that the positive sign is op tional when representing NR2 constants. Therefore, this test determines whether there will be any modification in the numerical representation if an unsigned NR2 constant is assigned to a simple variable. On output there should be two columns of numbers. The first column of numbers represents the desired numbers that can be compared with those outputted in the second column as the evaluated representations of the system.

6.2.2.2 Signed Constants

The objective of this test is the same as that of test 2.2.2.1 except that this test uses signed NR2 constants in the assignment made to the simple variables. The test has a similar output format.

6.2.2.3 Cross Assignment

The objective of this test is to determine whether repeated assignments of an NR2 constant between several different simple variables will modify the value of that constant. The test uses 26 of the simple variables, A0 through Z0. The simple variable A0 is initially assigned the NR2 constant and then the content of A0 is assigned to B0, and then B0 to C0, and then C0 to D0; and this sequence of assigning is continued through Z0. The output represents the value of Z0 which should be the value of the original NR2 constant assigned to A0.

```
******************
* PROGRAM FILE 6 *
******************
```

```
10 PRINT "PROGRAM FILE 6"
20 PRINT
30 PRINT
40 PRINT
90 PRINT " SECTION 6.1.1: SPACE ALLOTMENT (E-FORMAT), COMPACT FORM."
100 PRINT
110 PRINT "                              BEGIN TEST."
120 PRINT
130 PRINT "000000000111111111122222222223333333333444444444 5";
140 PRINT "55555555566666666677 7"
150 PRINT "123456789012345678901234567890123456789012345678 90";
160 PRINT "12345678901234567890 12"
170 PRINT
180 PRINT "  STANDARD";"        OPTIONAL";"        SYSTEM"
190 PRINT
200 PRINT "** 1.23456E32 *";"* 1.23456E+32 **";1.23456E32;"**"
210 PRINT "**-1.23456E32 *";"*-1.23456E+32 **";-1.23456E32;"**"
220 PRINT "**-1.23456E-24 *";"*-1.23456E-24 **";-1.23456E-24;"**"
230 PRINT "** 1.00000E30 *";"* 1.00000E+30 **";1E30;"**"
240 PRINT "**-9.87000E-37 *";"*-9.87000E-37 **";-9.87E-37;"**"
250 PRINT
260 PRINT "                                END TEST."
270 PRINT
280 PRINT "     SECTION 6.1.2: NR3 CONSTANTS SEPARATED BY SEMICOLONS."
290 PRINT
300 PRINT "                                 BEGIN TEST."
310 PRINT
320 PRINT "000000000111111111122222222223333333333444444444 5";
330 PRINT "55555555566666666677 7"
340 PRINT "123456789012345678901234567890123456789012345678 90";
350 PRINT "12345678901234567890 12"
360 PRINT 1E30;-9.87E-37;1.23456E32
370 PRINT -1.23456E32;+1.7895E36;5E-20
380 PRINT
390 PRINT "                                  END TEST."
400 PRINT
```

```
410 PRINT
420 PRINT
430 PRINT "                SECTION 6.2: ASSIGNED NUMERIC CONSTANTS."
440 PRINT
450 PRINT "              SECTION 6.2.1: ASSIGNMENT OF NR1 CONSTANTS."
460 PRINT
470 PRINT
480 PRINT "        SECTION 6.2.1.1: UNSIGNED CONSTANTS TO A VARIABLE."
490 PRINT
500 PRINT "                                  BEGIN TEST."
510 LET M1=1
520 LET J1=12345
530 LET K1=000
540 PRINT "NEEDED","SYSTEM"
550 PRINT "OUTPUT","OUTPUT"
560 PRINT
570 PRINT " 1 ",M1
580 PRINT " 12345 ",J1
590 PRINT " 0 ",K1
600 PRINT
610 PRINT "                                   END TEST."
620 PRINT
630 PRINT "         SECTION 6.2.1.2: SIGNED CONSTANTS TO A VARIABLE."
640 PRINT
650 PRINT "                                  BEGIN TEST."
660 PRINT
670 LET M2=+2
680 LET L2=-3
690 LET N2=-8765
700 LET O2=+6912
710 PRINT "NEEDED","SYSTEM"
720 PRINT "OUTPUT","OUTPUT"
730 PRINT
740 PRINT " 2 ",M2
750 PRINT "-3 ",L2
760 PRINT "-8765 ",N2
770 PRINT " 6912 ",O2
780 PRINT "                                   END TEST."
790 PRINT
800 PRINT " SECTION 6.2.1.3: TRANSITIVE ASSIGNMENT OF AN NR1 CONSTANT."
810 PRINT
820 PRINT "                                  BEGIN TEST."
830 PRINT
840 LET A=-99999
850 LET B=A
860 LET C=B
870 LET D=C
880 LET E=D
890 LET F=E
900 LET G=F
910 LET H=G
920 LET I=H
930 LET J=I
940 LET K=J
950 LET L=K
960 LET M=L
970 LET N=M
```

```
980 LET O=N
990 LET P=O
1000 LET Q=P
1010 LET R=Q
1020 LET S=R
1030 LET T=S
1040 LET U=T
1050 LET V=U
1060 LET W=V
1070 LET X=W
1080 LET Y=X
1090 LET Z=Y
1100 PRINT "NEEDED","SYSTEM"
1110 PRINT "OUTPUT","OUTPUT"
1120 PRINT
1130 PRINT "-99999 ",Z
1140 PRINT
1150 PRINT "                                    END TEST."
1160 PRINT
1170 PRINT "                SECTION 6.2.2: ASSIGNMENT OF NR2 CONSTANTS."
1180 PRINT
1190 PRINT "         SECTION 6.2.2.1: UNSIGNED CONSTANTS TO A VARIABLE."
1200 PRINT
1210 PRINT
1220 LET A3=1.05
1230 LET B3=358.672
1240 PRINT "NEEDED","SYSTEM"
1250 PRINT "OUTPUT","OUTPUT"
1260 PRINT
1270 PRINT " 1.05 ",A3
1280 PRINT " 358.672 ",B3
1290 PRINT
1300 PRINT "                                    END TEST."
1310 PRINT
1320 PRINT
1330 PRINT "          SECTION 6.2.2.2: SIGNED CONSTANTS TO A VARIABLE."
1340 PRINT
1350 PRINT "                                  BEGIN TEST."
1360 PRINT
1370 LET A4=-2.1
1380 LET B4=+3.1
1390 LET C4=-2714.25
1400 LET D4=+29.3054
1410 PRINT "NEEDED","SYSTEM"
1420 PRINT "OUTPUT","OUTPUT"
1430 PRINT
1440 PRINT "-2.1 ",A4
1450 PRINT " 3.1 ",B4
1460 PRINT "-2714.25 ",C4
1470 PRINT " 29.3054 ",D4
1480 PRINT
1490 PRINT "                                    END TEST."
1500 PRINT
1510 PRINT
1520 PRINT " SECTION 6.2.2.3: TRANSITIVE ASSIGNMENT OF AN NR2 CONSTANT."
1530 PRINT
1540 PRINT "                                  BEGIN TEST."
```

```
1550 LET A0=-9999.99
1560 LET B0=A0
1570 LET C0=B0
1580 LET D0=C0
1590 LET E0=D0
1600 LET F0=E0
1610 LET G0=F0
1620 LET H0=G0
1630 LET I0=H0
1640 LET J0=I0
1650 LET K0=J0
1660 LET L0=K0
1670 LET M0=L0
1680 LET N0=M0
1690 LET O0=N0
1700 LET P0=O0
1710 LET Q0=P0
1720 LET R0=Q0
1730 LET S0=R0
1740 LET T0=S0
1750 LET U0=T0
1760 LET V0=U0
1770 LET W0=V0
1780 LET X0=W0
1790 LET Y0=X0
1800 LET Z0=Y0
1810 PRINT "NEEDED","SYSTEM"
1820 PRINT "OUTPUT","OUTPUT"
1830 PRINT
1840 PRINT "-9999.99 ",Z0
1850 LET A5=1.5
1860 LET B5=-2.5
1870 LET C5=+3.5
1880 LET D5=4.5
1890 LET E5=+5.5
1900 LET F5=-6.5
1910 LET A1=A5
1920 LET B1=B5
1930 LET C1=C5
1940 LET D1=D5
1950 LET E1=E5
1960 LET F1=F5
1970 LET G1=A1
1980 LET H1=B1
1990 LET I1=C1
2000 LET J1=D1
2010 LET K1=E1
2020 LET L1=F1
2030 LET M1=G1
2040 LET N1=H1
2050 LET O1=I1
2060 LET P1=J1
2070 LET Q1=K1
2080 LET R1=L1
2090 LET S1=M1
2100 LET T1=N1
2110 LET U1=O1
```

```
2120 LET V1=P1
2130 LET W1=Q1
2140 LET X1=R1
2150 PRINT " 1.5 ",S1
2160 PRINT "-2.5 ",T1
2170 PRINT " 3.5 ",U1
2180 PRINT " 4.5 ",V1
2190 PRINT " 5.5 ",W1
2200 PRINT "-6.5 ",X1
2210 LET Y1=9.99999
2220 LET Z1=Y1
2230 PRINT " 9.99999 ",Z1
2240 PRINT
2250 PRINT "                                   END TEST."
2260 PRINT
2270 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 6

```
 SECTION 6.1.1: SPACE ALLOTMENT (E-FORMAT), COMPACT FORM.

                                BEGIN TEST.

00000000001111111111222222222233333333334444444444555555555566666666667777
12345678901234567890123456789012345678901234567890123456789012345678901234567890

  STANDARD         OPTIONAL         SYSTEM

** 1.23456E32 ** 1.23456E+32 ** 1.23456E+32 **
**-1.23456E32 **-1.23456E+32 **-1.23456E+32 **
**-1.23456E-24 **-1.23456E-24 **-1.23456E-24 **
** 1.00000E30 ** 1.00000E+30 ** 1.00000E+30 **
**-9.87000E-37 **-9.87000E-37 **-9.87000E-37 **

                                END TEST.

      SECTION 6.1.2: NR3 CONSTANTS SEPARATED BY SEMICOLONS.

                                BEGIN TEST.

00000000001111111111222222222233333333334444444444555555555566666666667777
12345678901234567890123456789012345678901234567890123456789012345678901234567890
 1.00000E+30 -9.87000E-37  1.23456E+32
```

```
-1.23456E+32  1.78950E+36  5.00000E-20
```

END TEST.

SECTION 6.2: ASSIGNED NUMERIC CONSTANTS.

SECTION 6.2.1: ASSIGNMENT OF NR1 CONSTANTS.

SECTION 6.2.1.1: UNSIGNED CONSTANTS TO A VARIABLE.

BEGIN TEST.

```
NEEDED          SYSTEM
OUTPUT          OUTPUT

 1               1
 12345           12345
 0               0
```

END TEST.

SECTION 6.2.1.2: SIGNED CONSTANTS TO A VARIABLE.

BEGIN TEST.

```
NEEDED          SYSTEM
OUTPUT          OUTPUT

 2               2
-3              -3
-8765           -8765
 6912            6912
```

END TEST.

SECTION 6.2.1.3: TRANSITIVE ASSIGNMENT OF AN NR1 CONSTANT.

BEGIN TEST.

```
NEEDED          SYSTEM
OUTPUT          OUTPUT

-99999          -99999
```

END TEST.

SECTION 6.2.2: ASSIGNMENT OF NR2 CONSTANTS.

SECTION 6.2.2.1: UNSIGNED CONSTANTS TO A VARIABLE.

```
NEEDED          SYSTEM
OUTPUT          OUTPUT

 1.05            1.05
 358.672         358.672
```

END TEST.

SECTION 6.2.2.2: SIGNED CONSTANTS TO A VARIABLE.

BEGIN TEST.

NEEDED OUTPUT	SYSTEM OUTPUT
-2.1	-2.1
3.1	3.1
-2714.25	-2714.25
29.3054	29.3054

END TEST.

SECTION 6.2.2.3: TRANSITIVE ASSIGNMENT OF AN NR2 CONSTANT.

BEGIN TEST.

NEEDED OUTPUT	SYSTEM OUTPUT
-9999.99	-9999.99
1.5	1.5
-2.5	-2.5
3.5	3.5
4.5	4.5
5.5	5.5
-6.5	-6.5
9.99999	9.99999

END TEST.

7.0 ASSIGNMENT OF FLOATING POINT CONSTANTS

This test verifies that signed as well as unsigned NR3 constants can be assigned to simple variables. It also verifies that the assigned value of an NR3 constant can be cross assigned from one simple variable to another. The user should refer to section 5, 6, 9, and 12 of BSR X3.60 and observe that trailing zeroes in the significand are optional.

7.1 Using NR3 Form to Assign NR1 and NR2 Constants, both Signed and Unsigned

This test verifies that the NR3 form of constants can be used to represent numerical constants used in Minimal BASIC program. That is, all numerical constants should be acceptable in NR3 form as program constants but on output should be printed in their appropriate form as either NR1 or NR2 constants. The test also verifies that a positive number in NR3 form with or without a preceding plus sign can be assigned to a simple variable without modifying the value of that constant. On output there should be two columns of numbers. The first column of numbers represents the desired numbers which can be compared with those outputted in the second column as the evaluated representation of the system.

7.2 Using NR3 Form to Assign NR3 Constants, both Signed and Unsigned

The objective of this test is to verify that the positive sign is optional when representing NR3 constants which cannot be represented in any other form except NR3. Therefore, this section tests whether there will be any modification in the numerical representation regardless of whether a plus sign is in front of the assigned NR3 number. On output there should be two columns of numbers. The first column of numbers represents the desired numbers which can be compared with those outputted in the second column as the evaluated representation of the system.

7.3 Cross Assignment of an NR3 Constant

The objective of this test is to sample a number of the allowed representations for a simple variable to determine that cross assignments of an NR3 constant will not modify the value of that constant. The test uses 33 of the simple variables: A2, B2, C2, D2, C3, D3, A6, B6, C6, D6, E2, E3, E4, E6, F2, F3, F4, F6, G2, G3, G4, G5, G6, H2, H3, H4, H5, H6, I2, I3, I4, I5, and I6. The simple variable A2 is initially assigned the NR3 constant and then the content of A2 is assigned to B2, and then B2 to C2, and then C2 to D2; and this form assigning continues sequentially through the list of variables above through I6. The final printed output represents the value of I6 which should be the value of the original NR3 constant assigned to A2.

****

```
                    * PROGRAM FILE 7 *
                    ******************
```

```
10 PRINT "PROGRAM FILE 7"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "              SECTION 7.0: ASSIGNMENT OF NR3 CONSTANTS."
100 PRINT
110 PRINT
120 PRINT "                          SECTION 7.1:"
130 PRINT
140 PRINT "          USING NR3 FORM TO ASSIGN NR1 AND NR2 CONSTANTS,"
150 PRINT "                    BOTH SIGNED AND UNSIGNED."
160 PRINT
170 PRINT
180 PRINT "                               BEGIN TEST."
190 PRINT
200 LET A7=1.05E02
210 LET B7=-7.6E1
220 LET C7=+332.4E0
230 LET D7=51.32E-1
240 LET E7=+5.34E-3
250 LET F7=-14.19E-2
260 LET G7=-9.9E+2
270 LET H7=+10.5210E+3
280 LET I7=4.56E+1
290 PRINT "NEEDED","SYSTEM"
300 PRINT "OUTPUT","OUTPUT"
310 PRINT
320 PRINT " 105 ",A7
330 PRINT "-76 ",B7
340 PRINT " 332.4 ",C7
350 PRINT " 5.132 ",D7
360 PRINT " .00534 ",E7
370 PRINT "-.1419 ",F7
380 PRINT "-990 ",G7
390 PRINT " 10521 ",H7
400 PRINT " 45.6 ",I7
410 PRINT
420 PRINT "                               END TEST."
430 PRINT
440 PRINT
450 PRINT "                          SECTION 7.2:"
460 PRINT
470 PRINT "          USING NR3 FORM TO ASSIGN NR3 CONSTANTS ONLY,"
480 PRINT "                    BOTH SIGNED AND UNSIGNED."
490 PRINT
500 PRINT
510 PRINT "                               BEGIN TEST."
520 PRINT
530 LET A8=1.E30
540 LET B8=+123.E20
```

```
550 LET C8=-11.E30
560 LET D8=144.E-21
570 LET E8=-12.E-22
580 LET F8=+3645.E-23
590 LET G8=1.E+34
600 LET H8=-200.E+21
610 LET I8=+99.E+32
620 LET A9=.234E20
630 LET B9=-.3E22
640 LET C9=+.44E17
650 LET D9=.36E-33
660 LET E9=+.9E-24
670 LET F9=-.10E-25
680 LET G9=.777E+18
690 LET H9=-.29E+31
700 LET I9=+.04E+26
710 LET J1=709E35
720 LET J2=+81E36
730 LET J3=-9E15
740 LET J4=627E+27
750 LET J5=+53E+19
760 LET J6=-4E+28
770 LET J7=1463E-29
780 LET J8=+2E-37
790 LET J9=-355E-19
800 PRINT "STANDARD","OPTIONAL","SYSTEM"
810 PRINT
820 PRINT " 1.00000E30 "," 1.00000E+30 ",A8
830 PRINT " 1.23000E22 "," 1.23000E+22 ",B8
840 PRINT "-1.10000E31 ","-1.10000E+31 ",C8
850 PRINT " 1.44000E-19 ","NONE",D8
860 PRINT "-1.20000E-21 ","NONE",E8
870 PRINT " 3.64500E=20 ","NONE",F8
880 PRINT " 1.00000E34 "," 1.00000E+34 ",G8
890 PRINT "-2.00000E23 ","-2.00000E+23 ",H8
900 PRINT " 9.90000E33 "," 9.90000E+33 ",I8
910 PRINT " 2.34000E19 "," 2.34000E+19 ",A9
920 PRINT "-3.00000E21 ","-3.00000E+21 ",B9
930 PRINT " 4.40000E16 "," 4.40000E+16 ",C9
940 PRINT " 3.60000E-34 ","NONE",D9
950 PRINT " 9.00000E-25 ","NONE",E9
960 PRINT "-1.00000E-26 ","NONE",F9
970 PRINT " 7.77000E17 "," 7.77000E+17 ",G9
980 PRINT "-2.90000E30 ","-2.90000E+30 ",H9
990 PRINT " 4.00000E24 "," 4.00000E+24 ",I9
1000 PRINT " 7.09000E37 "," 7.09000E+37 ",J1
1010 PRINT " 8.10000E7 "," 8.10000E+37 ",J2
1020 PRINT "-9.00000E15 ","-9.00000E+15 ",J3
1030 PRINT " 6.27000E29 "," 6.27000E+29 ",J4
1040 PRINT " 5.30000E20 "," 5.30000E+20 ",J5
1050 PRINT "-4.00000E28 ","-4.00000E+28 ",J6
1060 PRINT " 1.46300E-26 ","NONE",J7
1070 PRINT " 2.00000E-37 ","NONE",J8
1080 PRINT "-3.55000E-17 ","NONE",J9
1090 PRINT
1100 PRINT "                                   END TEST."
1110 PRINT
```

```
1120 PRINT
1130 PRINT "                              SECTION 7.3:"
1140 PRINT
1150 PRINT "                TRANSITIVE ASSIGNMENT OF AN NR3 CONSTANT."
1160 PRINT
1170 PRINT
1180 PRINT "                                   BEGIN TEST."
1190 PRINT
1200 LET A2=1.E30
1210 LET B2=A2
1220 LET C2=B2
1230 LET D2=C2
1240 LET C3=D2
1250 LET D3=C3
1260 LET A6=D3
1270 LET B6=A6
1280 LET C6=B6
1290 LET D6=C6
1300 LET E2=D6
1310 LET E3=E2
1320 LET E4=E3
1330 LET E6=E4
1340 LET F2=E6
1350 LET F3=F2
1360 LET F4=F3
1370 LET F6=F4
1380 LET G2=F6
1390 LET G3=G2
1400 LET G4=G3
1410 LET G5=G4
1420 LET G6=G5
1430 LET H2=G6
1440 LET H3=H2
1450 LET H4=H3
1460 LET H5=H4
1470 LET H6=H5
1480 LET I2=H6
1490 LET I3=I2
1500 LET I4=I3
1510 LET I5=I4
1520 LET I6=I5
1530 PRINT "STANDARD","OPTIONAL","SYSTEM"
1540 PRINT
1550 PRINT " 1.00000E30 "," 1.00000E+30 ",I6
1560 PRINT
1570 PRINT "                                     END TEST."
1580 PRINT
1590 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 7

SECTION 7.0: ASSIGNMENT OF NR3 CONSTANTS.

SECTION 7.1:

USING NR3 FORM TO ASSIGN NR1 AND NR2 CONSTANTS,
BOTH SIGNED AND UNSIGNED.

BEGIN TEST.

NEEDED OUTPUT	SYSTEM OUTPUT
105	105
-76	-76
332.4	332.4
5.132	5.132
.00534	.00534
-.1419	-.1419
-990	-990
10521	10521
45.6	45.6

END TEST.

SECTION 7.2:

USING NR3 FORM TO ASSIGN NR3 CONSTANTS ONLY,
BOTH SIGNED AND UNSIGNED.

BEGIN TEST.

STANDARD	OPTIONAL	SYSTEM
1.00000E30	1.00000E+30	1.00000E+30
1.23000E22	1.23000E+22	1.23000E+22
-1.10000E31	-1.10000E+31	-1.10000E+31
1.44000E-19	NONE	1.44000E-19
-1.20000E-21	NONE	-1.20000E-21
3.64500E=20	NONE	3.64500E-20
1.00000E34	1.00000E+34	1.00000E+34
-2.00000E23	-2.00000E+23	-2.00000E+23
9.90000E33	9.90000E+33	9.90000E+33
2.34000E19	2.34000E+19	2.34000E+19
-3.00000E21	-3.00000E+21	-3.00000E+21
4.40000E16	4.40000E+16	4.40000E+16
3.60000E-34	NONE	3.60000E-34
9.00000E-25	NONE	9.00000E-25
-1.00000E-26	NONE	-1.00000E-26
7.77000E17	7.77000E+17	7.77000E+17

```
-2.90000E30   -2.90000E+30  -2.90000E+30
 4.00000E24    4.00000E+24   4.00000E+24
 7.09000E37    7.09000E+37   7.09000E+37
 8.10000E7     8.10000E+37   8.10000E+37
-9.00000E15   -9.00000E+15  -9.00000E+15
 6.27000E29    6.27000E+29   6.27000E+29
 5.30000E20    5.30000E+20   5.30000E+20
-4.00000E28   -4.00000E+28  -4.00000E+28
 1.46300E-26  NONE           1.46300E-26
 2.00000E-37  NONE           2.00000E-37
-3.55000E-17  NONE          -3.55000E-17
```

END TEST.

SECTION 7.3:

TRANSITIVE ASSIGNMENT OF AN NR3 CONSTANT.

BEGIN TEST.

```
STANDARD      OPTIONAL      SYSTEM

 1.00000E30    1.00000E+30   1.00000E+30
```

END TEST.

8.0 TESTING THE MINIMAL LIMITS IN MAGNITUDE OF NUMERICAL CONSTANTS

The objective of this test is to determine whether nonzero numbers of six significant digits with absolute values near the endpoints of the interval 1E-38 to 1E38, can be used as print items in the form of numerical constants or be assigned to simple variables. The standard specifies that although constants can have an arbitrary number of digits in the exrad, the implementation-defined range shall be at least 1E-38 to 1E+38 (see section 5.4 of BSR X3.60). Nonzero constants, whose magnitudes are outside of the implementation-defined range, are to be diagnosed as errors. There will be specific error diagnostic tests in later sections. This section concentrates on keeping within the range 1E-38 to 1E38.

8.1 Printing NR3 Constants of Six Significant Digits That Are Near the Endpoints of: 1E38 to 1E-38

The objective of this subsection is to verify that NR3 constants with absolute values near the endpoints 1E-38 or 1E38 can be used as print-items.

8.1.1 Positive Exponent

The objective of this test is to print constants that, in absolute value, are near the endpoint 1E38. The test uses both positively and negatively signed NR3 constants. On output there should be three columns of numbers. The first column should be headed "Standard Form". This heads a list of numbers that should also be printed in the third column, which is headed "Form of System". Column two is a list of optional output forms for the NR3 constants.

8.1.2 Negative Exponent

The objective of this test is to execute the printing of NR3 constants that, in absolute value, are near the endpoint 1E-38. The output of this test is in the same format as that of test 8.1.1.

8.2 Assigning NR3 Constants of Six Significant Digits That Are Near the Endpoints of: 1E38 to 1E-38

The objective of this test is to verify that NR3 constants, whose absolute values are close to the magnitude of the endpoints 1E-38 and 1E38, can be assigned to simple variables. These simple variables are then used as print-items.

8.2.1 Positive Exponent

The objective of this test is the same as that of test 8.1.1, except that for this test the NR3 constants are assigned to simple variables. The structures of the two tests are basically the same, except for the simple variables used. On output, the format should be the same for this test as

for test 8.1.1.

8.2.2 Negative Exponent

The objective of this test is the same as that of test 8.1.2, except that for this test the NR3 constants are assigned to simple variables which are the print-items used. The structure and output of this test are the same as those of test 8.2.1.

8.3 Printing and Assigning the Extreme Values 1E+38 and 1E-38

8.3.1 As a Print-Item in Numerical Form

The objective of this test is to execute as print-items, in NR3 constant form, both the positive and negative cases of 1E-38 and 1E38. The test uses a three column formatted output. The first column, titled "Standard Form", illustrates the expected standard conforming E-formatted output. The second column, titled "Optional Form", gives another acceptable standard conforming output for E-formatted constants. Column three, titled "Form of System", gives the output form used by the system being tested.

8.3.2 As Print-Items Which Have Been Numerically Assigned

The objective of this test is the same as that of test 8.3.1, except that for this test the positive and negative cases of the endpoints, 1E-38 and 1E38, are first assigned to simple numeric variables, and then those simple variables are used as the print-items. The structure of the test is the same as that of test 8.3.1. The output format for this test is the same as the output format for test 8.3.1.

```
**********
* PROGRAM FILE 8 *
**********
```

```
10 PRINT "PROGRAM FILE 8"
60 PRINT
70 PRINT
80 PRINT
90 PRINT " SECTION 8.0: TESTING THE MAGNITUDE OF NUMERICAL CONSTANTS"
100 PRINT "              WITHIN MINIMAL LIMITS OF 1E-38 AND 1E38."
110 PRINT
120 PRINT "SECTION 8.1: PRINTING NR3 CONSTANTS OF 6 SIGNIFICANT DI-"
130 PRINT "                  GITS WHICH ARE NEAR THE MAGNITUDE OF:"
140 PRINT
150 PRINT "                                  1E38"
```

```
160 PRINT "                                   OR"
170 PRINT "                                  1E-38."
180 PRINT
190 PRINT
200 PRINT
210 PRINT "                               SECTION 8.1.1"
220 PRINT
230 PRINT "                              (POSITIVE EXPONENT.)"
240 PRINT
250 PRINT
260 PRINT "                                 BEGIN TEST."
270 PRINT
280 PRINT "        ","        "," FORM "
290 PRINT "STANDARD","OPTIONAL","  OF  "
300 PRINT "  FORM  ","  FOR  ","SYSTEM"
310 PRINT
320 PRINT "-9.99999E30 ","-9.99999E+30 ",-9.99999E30
330 PRINT " 9.99999E31 "," 9.99999E+31 ",+9.99999E31
340 PRINT "-9.99999E32 ","-9.99999E+32 ",-9.99999E+32
350 PRINT " 9.99999E33 "," 9.99999E+33 ",9.99999E33
360 PRINT "-9.99999E34 ","-9.99999E+34 ",-9.99999E34
370 PRINT " 9.99999E35 "," 9.99999E+35 ",+9.99999E+35
380 PRINT "-9.99999E36 ","-9.99999E+36 ",-9.99999E+36
390 PRINT " 9.99999E37 "," 9.99999E+37 ",9.99999E37
400 PRINT
410 PRINT "                                   END TEST."
420 PRINT
430 PRINT
440 PRINT
450 PRINT "                               SECTION 8.1.2"
460 PRINT
470 PRINT "                              (NEGATIVE EXPONENT.)"
480 PRINT
490 PRINT
500 PRINT "                                 BEGIN TEST."
510 PRINT
520 PRINT "        ","        "," FORM "
530 PRINT "STANDARD","OPTIONAL","  OF  "
540 PRINT "  FORM  ","  FORM  ","SYSTEM"
550 PRINT
560 PRINT "-9.99999E-30 ","NONE",-9.99999E-30
570 PRINT " 9.99999E-31 ","NONE",+9.99999E-31
580 PRINT "-9.99999E-32 ","NONE",-9.99999E-32
590 PRINT " 9.99999E-33 ","NONE",9.99999E-33
600 PRINT "-9.99999E-34 ","NONE",-9.99999E-34
610 PRINT " 9.99999E-35 ","NONE",+9.99999E-35
620 PRINT "-9.99999E-36 ","NONE",-9.99999E-36
630 PRINT " 9.99999E-37 ","NONE",9.99999E-37
640 PRINT
650 PRINT "                                   END TEST."
660 PRINT
670 PRINT
680 PRINT
690 PRINT "SECTION 8.2: ASSIGNING NR3 CONSTANTS OF 6 SIGNIFICANT DI-"
700 PRINT "                  GITS WHICH ARE NEAR THE MAGNITUDE OF:"
710 PRINT
720 PRINT "                                     1E38"
```

```
730 PRINT "                                        OR"
740 PRINT "                                       1E-38."
750 PRINT
760 PRINT
770 PRINT
780 PRINT "                                  SECTION 8.2.1"
790 PRINT
800 PRINT "                                (POSITIVE EXPONENT.)"
810 PRINT
820 PRINT
830 PRINT "                                    BEGIN TEST."
840 PRINT
850 PRINT "        ","        "," FORM "
860 PRINT "STANDARD","OPTIONAL","  OF  "
870 PRINT "  FORM  ","  FORM  ","SYSTEM"
880 PRINT
890 LET A=-9.99999E30
900 LET B=+9.99999E31
910 LET C=-9.99999E+32
920 LET D=9.99999E33
930 LET E=-9.99999E34
940 LET F=+9.99999E+35
950 LET G=-9.99999E+36
960 LET H=9.99999E37
970 PRINT "-9.99999E30 ","-9.99999E+30 ",A
980 PRINT " 9.99999E31 "," 9.99999E+31 ",B
990 PRINT "-9.99999E32 ","-9.99999E+32 ",C
1000 PRINT " 9.99999E33 "," 9.99999E+33 ",D
1010 PRINT "-9.99999E34 ","-9.99999E+34",E
1020 PRINT " 9.99999E35 "," 9.99999E+35 ",F
1030 PRINT "-9.99999E36 ","-9.99999E+36 ",G
1040 PRINT " 9.99999E37 "," 9.99999E+37 ",H
1050 PRINT
1060 PRINT "                                      END TEST."
1070 PRINT
1080 PRINT
1090 PRINT
1100 PRINT "                                  SECTION 8.2.2"
1110 PRINT
1120 PRINT "                                (NEGATIVE EXPONENT.)"
1130 PRINT
1140 PRINT
1150 PRINT "        ","        "," FORM "
1160 PRINT "STANDARD","OPTIONAL","  OF  "
1170 PRINT "  FORM  ","  FORM  ","SYSTEM"
1180 PRINT
1190 LET A=-9.99999E-30
1200 LET B=+9.99999E-31
1210 LET C=-9.99999E-32
1220 LET D=9.99999E-33
1230 LET E=-9.99999E-34
1240 LET F=+9.99999E-35
1250 LET G=-9.99999E-36
1260 LET H=9.99999E-37
1270 PRINT "-9.99999E-30 ","NONE",A
1280 PRINT " 9.99999E-31 ","NONE",B
1290 PRINT "-9.99999E-32 ","NONE",C
```

```
1300 PRINT " 9.99999E-33 ","NONE",D
1310 PRINT "-9.99999E-34 ","NONE",E
1320 PRINT " 9.99999E-35 ","NONE",F
1330 PRINT "-9.99999E-36 ","NONE",G
1340 PRINT " 9.99999E-37 ","NONE",H
1350 PRINT
1360 PRINT "                                   END TEST."
1370 PRINT
1380 PRINT
1390 PRINT
1400 PRINT "SECTION 8.3: PRINTING/ASSIGNING THE EXTREMES OF THE NR3"
1410 PRINT "                              (FORM) CONSTANTS."
1420 PRINT
1430 PRINT
1440 PRINT
1450 PRINT "                                 SECTION 8.3.1"
1460 PRINT
1470 PRINT "           (AS A PRINT ITEM IN CONSTANT FORM, NUMERICALLY.)"
1480 PRINT
1490 PRINT
1500 PRINT "                                   BEGIN TEST."
1510 PRINT
1520 PRINT "         ","         "," FORM "
1530 PRINT "STANDARD","OPTIONAL","  OF   "
1540 PRINT "  FORM  ","  FORM  ","SYSTEM"
1550 PRINT
1560 PRINT " 1.00000E38 "," 1.00000E+38 ",1.00000E38
1570 PRINT " 1.00000E-38 ","NONE",1.00000E-38
1580 PRINT "-1.00000E38 ","-1.00000E+38 ",-1.00000E+38
1590 PRINT "-1.00000E-38 ","NONE",-1.00000E-38
1600 PRINT
1610 PRINT "                                    END TEST."
1620 PRINT
1630 PRINT
1640 PRINT
1650 PRINT "                                 SECTION 8.3.2"
1660 PRINT
1670 PRINT "      (AS A PRINT ITEM WHICH HAS BEEN NUMERICALLY ASSIGNED.)"
1680 PRINT
1690 PRINT
1700 PRINT "                                   BEGIN TEST."
1710 PRINT
1720 PRINT "         ","         "," FORM "
1730 PRINT "STANDARD","OPTIONAL","  OF   "
1740 PRINT "  FORM  ","  FORM  ","SYSTEM"
1750 PRINT
1760 LET A=1.00000E38
1770 LET B=1.00000E-38
1780 LET C=-1.00000E+38
1790 LET D=-1.00000E-38
1800 PRINT " 1.00000E38 "," 1.00000E+38 ",A
1810 PRINT " 1.00000E-38 ","NONE",B
1820 PRINT "-1.00000E38 ","-1.00000E+38 ",C
1830 PRINT "-1.00000E-38 ","NONE",D
1840 PRINT
1850 PRINT "                                   END TEST."
1860 PRINT
```

```
1870 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 8

SECTION 8.0: TESTING THE MAGNITUDE OF NUMERICAL CONSTANTS
WITHIN MINIMAL LIMITS OF 1E-38 AND 1E38.

SECTION 8.1: PRINTING NR3 CONSTANTS OF 6 SIGNIFICANT DIGITS WHICH ARE NEAR THE MAGNITUDE OF:

1E38
OR
1E-38.

SECTION 8.1.1

(POSITIVE EXPONENT.)

BEGIN TEST.

STANDARD FORM	OPTIONAL FOR	FORM OF SYSTEM
-9.99999E30	-9.99999E+30	-9.99999E+30
9.99999E31	9.99999E+31	9.99999E+31
-9.99999E32	-9.99999E+32	-9.99999E+32
9.99999E33	9.99999E+33	9.99999E+33
-9.99999E34	-9.99999E+34	-9.99999E+34
9.99999E35	9.99999E+35	9.99999E+35
-9.99999E36	-9.99999E+36	-9.99999E+36
9.99999E37	9.99999E+37	9.99999E+37

END TEST.

SECTION 8.1.2

(NEGATIVE EXPONENT.)

BEGIN TEST.

STANDARD FORM	OPTIONAL FORM	FORM OF SYSTEM
-9.99999E-30	NONE	-9.99999E-30
9.99999E-31	NONE	9.99999E-31
-9.99999E-32	NONE	-9.99999E-32
9.99999E-33	NONE	9.99999E-33
-9.99999E-34	NONE	-9.99999E-34
9.99999E-35	NONE	9.99999E-35
-9.99999E-36	NONE	-9.99999E-36
9.99999E-37	NONE	9.99999E-37

END TEST.

SECTION 8.2: ASSIGNING NR3 CONSTANTS OF 6 SIGNIFICANT DIGITS WHICH ARE NEAR THE MAGNITUDE OF:

1E38
OR
1E-38.

SECTION 8.2.1

(POSITIVE EXPONENT.)

BEGIN TEST.

STANDARD FORM	OPTIONAL FORM	FORM OF SYSTEM
-9.99999E30	-9.99999E+30	-9.99999E+30
9.99999E31	9.99999E+31	9.99999E+31
-9.99999E32	-9.99999E+32	-9.99999E+32
9.99999E33	9.99999E+33	9.99999E+33
-9.99999E34	-9.99999E+34	-9.99999E+34
9.99999E35	9.99999E+35	9.99999E+35
-9.99999E36	-9.99999E+36	-9.99999E+36
9.99999E37	9.99999E+37	9.99999E+37

END TEST.

SECTION 8.2.2

(NEGATIVE EXPONENT.)

STANDARD FORM	OPTIONAL FORM	FORM OF SYSTEM
-9.99999E-30	NONE	-9.99999E-30
9.99999E-31	NONE	9.99999E-31
-9.99999E-32	NONE	-9.99999E-32
9.99999E-33	NONE	9.99999E-33
-9.99999E-34	NONE	-9.99999E-34
9.99999E-35	NONE	9.99999E-35
-9.99999E-36	NONE	-9.99999E-36
9.99999E-37	NONE	9.99999E-37

END TEST.

SECTION 8.3: PRINTING/ASSIGNING THE EXTREMES OF THE NR3 (FORM) CONSTANTS.

SECTION 8.3.1

(AS A PRINT ITEM IN CONSTANT FORM, NUMERICALLY.)

BEGIN TEST.

STANDARD FORM	OPTIONAL FORM	FORM OF SYSTEM
1.00000E38	1.00000E+38	1.00000E+38
1.00000E-38	NONE	1.00000E-38
-1.00000E38	-1.00000E+38	-1.00000E+38
-1.00000E-38	NONE	-1.00000E-38

END TEST.

SECTION 8.3.2

(AS A PRINT ITEM WHICH HAS BEEN NUMERICALLY ASSIGNED.)

BEGIN TEST.

STANDARD FORM	OPTIONAL FORM	FORM OF SYSTEM
1.00000E38	1.00000E+38	1.00000E+38
1.00000E-38	NONE	1.00000E-38
-1.00000E38	-1.00000E+38	-1.00000E+38
-1.00000E-38	NONE	-1.00000E-38

END TEST.

9.0 THE REM AND GOTO STATEMENTS

The objectives of this test are two-fold in that they test the REM and GOTO-statements both separately and together. The REM statement is tested for the requirement that it allow the use of any characters in the ASCII character set for BASIC in its REM-string . The GOTO-statement is tested for its unconditional transferability. This section begins the evaluation of the host system's control structure with the most primitive BASIC control statement. The user should refer here to sections 10 and 18 and the syntax of a remark-string in section 3.2 of BSR X3.60.

9.1 REM-Statement as a Program Annotator

The objective of this test is to use all of the ASCII allowed Minimal BASIC string-characters within a remark-string. This character set consists of the set of upper-case Roman letters referenced by the ASCII character set in positions 4/1 through 5/10, the set of Arabic digits referenced by the ASCII character set in positions 3/0 through 3/9 and the remaining string-characters. The remaining string-characters consist of the following characters: space, exclamation-point, number-sign, dollar, percent, ampersand, quote, apostrophe, open-parenthesis, close-parenthesis, asterisk, plus, comma, minus, period, slant, colon, semicolon, less-than, equals, greater-than, question-mark, circumflex, and underline. The test uses each of the characters described above in REM strings. On output, there must be only one printed statement between the beginning of the test and its end with two blank lines above it and one below it.

9.2 GOTO Statement with a Single Space Between GO and TO

The objective of this test is to test the transfer capability of the unconditional GOTO-statement. This transfer capability is tested for both forward and backward transferring. In testing the forward and backward transferring capabilities of the GOTO-statements, the transfers are made to both adjacent and non-adjacent line numbers. Along with these transfers there is also a test of the capability of the GOTO-statement to carry with its transfers any numeric values that may be requested , i.e. there is a test to verify that assigned variables are global. In this test section, only a single space is used between the GO and TO in the keyword. On output there should be numerically ordered printing of the digits 1, 2, ..., 8 for each successful transfer by the GOTO-statements. There are also error traps in the test. Should a transfer be unsuccessful, with no diagnostic and execution continuing at the next statement, then a printout of the attempted line transfer, just prior to the halting of execution, will appear.

9.3 GOTO Statement with Many Spaces Between GO and TO

The objective here is to test the use of an arbitrary number of spaces within the keyword between the GO and the TO of the GOTO-statement. The test also verifies that although the REM-statement has no effect when executed, it can serve as a statement to which control is transferred.

The output for this test should be the same as for test 9.2. Error traps are also a part of this test for any faulty transfer attempts. The printout for any errors will be the same as specified for test 9.2.

9.4 GOTO Statement with a Transfer to a GOTO

The objective of this test is to verify further the unconditional transfer capability of the GOTO-statement, which in this test transfers to another GOTO-statement. The test has error messages printed if there are any incorrect transfers. These messages will indicate which transfers were requested, but failed. On output there should be a statement to the user on the status of the test.

```
******************
* PROGRAM FILE 9 *
******************
```

```
1 PRINT "PROGRAM FILE 9"
2 PRINT
3 PRINT
4 PRINT
30 PRINT "              SECTION 9.1: REM-STATEMENT, AN ANNOTATOR."
40 PRINT
50 PRINT "                                   BEGIN TEST."
60 PRINT
70 REM    THIS TEST SECTION TESTS THE REM STATEMENT.
80 REM ALL THE MINIMAL BASIC CHARACTER SET OF THE ASCII CHARACTER
90 REM    SET ARE GIVEN BELOW.
100 REM
110 REM    ABCDEFGHIJKLMNOPQRSTUVWXYZ*()%_
120 REM  ?0123456789"!#$&'+,-./:;<=>^
130 REM
140 REM THE PROGRAM FLOW SHOULD NOT BE ALTERED BY REM STATEMENTS.
150 REM
160 PRINT
170 PRINT "               IF THIS LINE ONLY PRINT, REM TEST PASSES."
180 PRINT
190 PRINT "                                   END TEST."
200 PRINT
210 PRINT "              SECTION 9.2: GOTO-STATEMENT, SINGLE SPACED."
220 PRINT
230 PRINT "                                   BEGIN TEST."
240 PRINT
250 PRINT "    IF A ONE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
260 GO TO 430
270 PRINT "           ERROR, TRANSFER FROM LINE 260 TO 430 IMPROPER."
280 LET M=3
290 PRINT TAB(30);M
```

```
300 PRINT "    IF A FOUR FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
310 GO TO 580
320 PRINT "            ERROR, TRANSFER FROM LINE 310 TO 580 IMPROPER."
330 LET M=7
340 PRINT TAB(30);M
350 PRINT " IF AN EIGHT FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
360 GO TO 480
370 PRINT "            ERROR, TRANSFER FROM LUNE 360 TO 480 IMPROPER."
380 LET M=2
390 PRINT TAB(30);M
400 PRINT " IF A THREE FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
410 GO TO 280
420 PRINT "            ERROR, TRANSFER FROM LINE 410 TO 280 IMPROPER."
430 LET M=1
440 PRINT TAB(30);M
450 PRINT "    IF A TWO FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
460 GO TO 380
470 PRINT "            ERROR, TRANSFER FROM LINE 460 TO 380 IMPROPER."
480 LET M=8
490 PRINT TAB(30);M
500 PRINT "      IF A MESSAGE ON OUTPUT SHOULD FOLLOW THIS PARAGRAPH,"
505 PRINT "THEN FORWARD TRANSFER PERFORMED."
510 GO TO 680
520 PRINT "            ERROR, TRANSFER FROM LINE 510 TO 680 IMPROPER."
530 LET M=6
540 PRINT TAB(30);M
550 PRINT " IF A SEVEN FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
560 GO TO 330
570 PRINT "            ERROR, TRANSFER FROM LINE 560 TO 330 IMPROPER."
580 LET M=4
590 PRINT TAB(30);M
600 PRINT "    IF A FIVE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
610 GO TO 630
620 PRINT "            ERROR, TRANSFER FROM LINE 610 TO 630 IMPROPER."
630 LET M=5
640 PRINT TAB(30);M
650 PRINT "    IF A SIX FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
660 GO TO 530
670 PRINT "            ERROR, TRANSFER FROM LINE 660 TO 530 IMPROPER."
680 PRINT "       THE ABOVE PRINTOUT SHOULD BE IN THE ORDER 1,2,...,8."
690 PRINT
700 PRINT "                                  END TEST."
710 PRINT
720 PRINT "          SECTION 9.3: GOTO-STATEMENT, ANY SPACE/REM."
730 PRINT
740 PRINT "                                BEGIN TEST."
750 PRINT
760 PRINT "    IF A ONE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
770 GOTO 970
780 PRINT "            ERROR, TRANSFER FROM LINE 770 TO 970 IMPROPER."
790 REM CONTROL TRANSFERED BACKWARDS TO THIS STATEMENT.
800 LET M=3
810 PRINT TAB(30);M
820 PRINT "    IF A FOUR FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
830 GO      TO 1150
840 PRINT "             ERROR, TRANSFER FROM LINE  830 TO 1150 IMPROPER.
850 REM CONTROL TRANSFERED BACKWARDS TO THIS STATEMENT.
```

```
860 LET M=7
870 PRINT TAB(30);M
880 PRINT " IF AN EIGHT FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
890 GO              TO 1030
900 PRINT "           ERROR, TRANSFER FROM LINE 890 TO 1030 IMPROPER."
910 REM CONTROL TRANSFERED BACKWARDS TO THIS STATEMENT."
920 LET M=2
930 PRINT TAB(30);M
940 PRINT " IF A THREE FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
950 GO    TO 790
960 PRINT "           ERROR, TRANSFER FROM LINE 950 TO 790 IMPROPER."
970 REM CONTROL TRANSFERED FORWARD TO THIS STATEMENT.
980 LET M=1
990 PRINT TAB(30);M
1000 PRINT "   IF A TWO FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
1010 GO  TO 910
1020 PRINT "           ERROR, TRANSFER FROM LINE 1010 TO 910 IMPROPER."
1030 REM CONTROL TRANSFERED FORWARD TO THIS STATEMENT.
1040 LET M=8
1050 PRINT TAB(30);M
1060 PRINT "     IF A MESSAGE ON OUTPUT SHOULD FOLLOW THIS PARAGRAPH,"
1065 PRINT "THEN FORWARD TRANSFER PERFORMED."
1070 GO               TO 1270
1080 PRINT "          ERROR, TRANSFER FROM LINE 1070 TO 1270 IMPROPER."
1090 REM CONTROL TRANSFERED BACKWARDS TO THIS STATEMENT.
1100 LET M=6
1110 PRINT TAB(30);M
1120 PRINT " IF A SEVEN FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
1130 GO            TO 850
1140 PRINT "          ERROR, TRANSFER FROM LINE 1130 TO 850 IMPROPER."
1150 REM CONTROL TRANSFERED FORWARD TO THIS STATEMENT.
1160 LET M=4
1170 PRINT TAB(30);M
1180 PRINT "   IF A FIVE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED."
1190 GO        TO 1210
1200 PRINT "          ERROR, TRANSFER FROM LINE 1190 TO 1210 IMPROPER."
1210 REM CONTROL TRANSFERED FORWARD TO THIS STATEMENT.
1220 LET M=5
1230 PRINT TAB(30);M
1240 PRINT "   IF A SIX FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED."
1250 GO          TO 1090
1260 PRINT "          ERROR, TRANSFER FROM LINE 1250 TO 1090 IMPROPER."
1270 REM CONTROL TRANSFERED FORWARD TO THIS STATEMENT.
1280 PRINT "       THE ABOVE PRINTOUT SHOULD BE IN THE ORDER 1,2,...,8."
1290 PRINT
1300 PRINT "                                  END TEST."
1310 PRINT
1320 PRINT
1330 PRINT "          SECTION 9.4: GOTO-STATEMENT, TRANSFER TO A GOTO."
1340 PRINT
1350 PRINT "                                BEGIN TEST."
1360 PRINT
1370 GO TO 1440
1380 PRINT "          ERROR, TRANSFER FROM LINE 1370 TO 1440 IMPROPER."
1390 PRINT "                  IF NO ERROR MESSAGES, TEST PASSED."
1400 GO TO 1460
1410 PRINT "          ERROR, TRANSFER FROM LINE 1400 TO 1460 IMPROPER."
```

```
1420 GO TO 1390
1430 PRINT "           ERROR, TRANSFER FROM LINE 1420 TO 1390 IMPROPER."
1440 GO TO 1420
1450 PRINT "           ERROR, TRANSFER FROM LINE 1440 TO 1420 IMPROPER."
1460 PRINT
1470 PRINT "                                  END TEST."
1480 PRINT
1500 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

```
PROGRAM FILE 9

              SECTION 9.1: REM-STATEMENT, AN ANNOTATOR.

                           BEGIN TEST.

              IF THIS LINE ONLY PRINT, REM TEST PASSES.

                            END TEST.

              SECTION 9.2: GOTO-STATEMENT, SINGLE SPACED.

                           BEGIN TEST.

    IF A ONE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
                                1
   IF A TWO FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
                                2
 IF A THREE FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
                                3
   IF A FOUR FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
                                4
   IF A FIVE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
                                5
   IF A SIX FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
                                6
 IF A SEVEN FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
                                7
 IF AN EIGHT FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
                                8
     IF A MESSAGE ON OUTPUT SHOULD FOLLOW THIS PARAGRAPH,
THEN FORWARD TRANSFER PERFORMED.
       THE ABOVE PRINTOUT SHOULD BE IN THE ORDER 1,2,...,8.
```

END TEST.

SECTION 9.3: GOTO-STATEMENT, ANY SPACE/REM.

BEGIN TEST.

IF A ONE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
1
IF A TWO FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
2
IF A THREE FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
3
IF A FOUR FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
4
IF A FIVE FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
5
IF A SIX FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
6
IF A SEVEN FOLLOWS THIS LINE, BACKWARD TRANSFER PERFORMED.
7
IF AN EIGHT FOLLOWS THIS LINE, FORWARD TRANSFER PERFORMED.
8
IF A MESSAGE ON OUTPUT SHOULD FOLLOW THIS PARAGRAPH,
THEN FORWARD TRANSFER PERFORMED.
THE ABOVE PRINTOUT SHOULD BE IN THE ORDER 1,2,...,8.

END TEST.

SECTION 9.4: GOTO-STATEMENT, TRANSFER TO A GOTO.

BEGIN TEST.

IF NO ERROR MESSAGES, TEST PASSED.

END TEST.

10.0 TEST FOR GOTO WITH ILLEGAL STATEMENT LABEL

The objective of this test is to verify that the execution of a GOTO-statement, which refers to a non-existent line, will cause program execution to be suspended by the system. As specified in section 10.4 of BSR X3.60 all line-numbers in control statements must refer to lines in the program. In order to continue, this program will require some user-oriented restart procedure but this would be implementation dependent. The structure of this test is specifically constructed with a GOTO-transfer to a non-existent line number in order to determine whether the system implementation detects and processes this error. There should be some form of implementation defined diagnostic printed upon execution attempt of this program.

```
********************
* PROGRAM FILE 10 *
********************
```

```
10 PRINT "PROGRAM FILE 10"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "          SECTION 10.0: FATAL ERROR CHECK ON GOTO-STATEMENT."
100 PRINT
110 PRINT
120 PRINT
130 PRINT "     THIS SECTION HAS AS ITS OBJECTIVE TO USE A GOTO-STATE-"
140 PRINT "MENT WHICH REFERS TO A NON-EXISTENT LINE-NUMBER.  IF THE"
150 PRINT "SYSTEM RECOGNIZES THIS AS A FATAL ERROR (THAT IS, SUSPEND-"
160 PRINT "ING PROGRAM EXECUTION SUCH THAT USER-DIRECTED RESTART PRO-"
170 PRINT "CEDURES ARE REQUIRED), THEN THE TEST WILL HAVE PASSED."
180 PRINT
190 PRINT
200 PRINT
210 PRINT "                                   BEGIN TEST."
220 PRINT
230 GOTO 89
240 PRINT "SYSTEM FAILED TEST."
250 PRINT
260 PRINT "                                    END TEST."
270 PRINT
280 END
```

****
* SAMPLE OUTPUT *
****

This program requires a diagnostic. Since these are implementation dependent no universal diagnostic statement can be illustrated. However the following message illustrates the problem in the program and is given here only as an example:

```
? UNDEFINED LINE NUMBER 89 IN LINE 230
```

11.0 THE IF-THEN STATEMENT USED TO COMPARE POSITIVE NUMERICAL CONSTANTS

This test not only performs standard conforming executions of the IF-THEN statement, but also tests the relation operations. The IF-THEN statement allows interruption of the normal sequence of execution of statements. It causes execution to continue at a specified line, rather than at the one with the next higher line number. The syntax of the IF-THEN statement has the general form: IF expl rel exp2 THEN line-number. Expl and exp2 are numeric expressions or string expressions. Rel is any of the following relational operators: <, <=, =, <>, >=, or >. Equality in this test means strict equality rather than "approximate equality." Each test of the relation between expl and exp2 is designed to test both the true and false case of the IF- THEN statement. This is done by: (1) testing the IF-THEN statement against known true cases, and (2) against known false cases. Since there is no collating sequence specified by the standard, string comparisons are made only for = and <>.

All output for the IF-THEN test is formatted as a truth table. There are three headings for the output of each test. The first heading, "Assigned Values", labels the column of numbers that are being assigned to variables for comparison. The second heading, "Comparison of", labels the column of illustrative comparisons used in the IF-THEN statements. The third heading, "Truth Table", labels the symbols =, <, >, <>, >=, and <= used in the IF-THEN comparisons. Each of these symbols forms a column of comparative results in the truth table by the system's evaluations of the IF-THEN comparisons. The entries in the truth table are either T, F, or X, where X means that a comparison failed to yield the proper truth result. The truth table output is used in the next several tests. The user should refer to section 10 of BSR X3.60-**** for the proper specifications.

11.1 GO TO Transfer to an IF-THEN

The objective of this test is to extend the capability of the GOTO-statement. In fact the test requires the processor to perform a transfer to an IF-THEN statement by a GOTO-statement. Only a fail message can be printed. No other output will be generated if the transfer is performed.

11.2 IF-THEN Comparison of a Positive Numerical Constant with a Positive Numerical Constant

The objectives of the next two tests are simply to compare the values of positive numbers.

11.2.1 Equal Case

This test shows that the comparison of a positive numerical value with itself will be properly assigned a truth value by the IF- THEN statement for each of the relation symbols =, <, >, <>, >=, and <=. This test prints three possible outputs in the truth table. An "F" or a "T" indicates that the IF-THEN statement has made a correct evaluation where appropriate and performed the proper transfer. An "X" indicates that an incorrect evaluation

has been made and program flow continued at the next higher line number. For this test the program supplies the truth values in the first row of the truth table in the sample output for PROGRAM FILE 11.

11.2.2 Unequal Case

The objective of this test is to show that the comparison of a positive numerical value with another positive value, not of equal magnitude, will be properly made by the IF-THEN statement for each of the relation symbols: =, <, >, <>, >=, and <= . This test supplies the information needed in the second row of the table in the sample output.

11.3 IF-THEN Comparison of a Positive Number with a Negative Number

The objective of this test is to show that the comparison of a positive numerical value with a negative value, of the same absolute value, will be properly made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The output for this test is in the third row of the truth table in the sample output.

11.4 IF-THEN Comparison of a Negative Number with a Positive Number

The objective of this test is to show that the comparison of a negative numerical value with a positive value, not of the same absolute value, will be properly made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . This test prints output in the fourth row of the truth table.

11.5 IF-THEN Comparison of a Positive Number with Zero

The objective of this test is to show that the comparison of a positive numerical value with zero will be properly made by each of the relation symbols =, <, >, <>, >=, and <= . This test prints its results in the fifth row of the truth table.

11.6 IF-THEN Comparison of Zero with a Positive Number

The objective of this test is to show that the comparison of zero when compared with a positive value will be properly made for each of the relation symbols =, <, >, <>, >=, and <= . The test output lies in the sixth row of the truth table.

11.7 IF-THEN Comparison of Zero with Zero

The objective of this test is to show that the comparison of zero itself with zero will be properly made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output lies in the seventh row of the truth table.

```
                    ********************
                    * PROGRAM FILE 11 *
                    ********************
```

```
10 PRINT "PROGRAM FILE 11"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                                   BEGIN TEST."
100 PRINT
110 LET A$="="
120 LET B$="<"
130 LET C$=">"
140 LET D$="<>"
150 LET E$=">="
160 LET F$="<="
170 LET A=1
180 LET B=2
190 LET C=3
200 LET L=-1
210 LET X=-2
220 LET Y=3
230 LET Z=0
240 PRINT "ASSIGNED";TAB(20);"COMPARISON OF";TAB(54);"TRUTH TABLE"
250 PRINT " VALUES ";TAB(44);A$;TAB(49);B$;TAB(54);C$;TAB(59);D$;
260 PRINT TAB(65);E$;TAB(70);F$
270 PRINT
280 REM
290 REM ***     THE FIRST SECTION, 11.1, IS AN INTIAL TEST OF A ***
300 REM ***GOTO TRANSFER TO AN IF-THEN STATEMENT WHICH IS VITAL***
310 REM ***TO ERROR MESSAGES IN THE TESTS OF THE IF-THEN.      ***
320 REM
330 REM                SECTION 11.1: GOTO/IF-THEN.
340 GOTO 420
350 PRINT "                    GOTO TRANSFER TO IF-THEN, FAILED."
360 PRINT "                                      PLUS"
370 PRINT "CONTINUED TEST OF IF-THEN, MAY HAVE INVALID ERROR MESSAGES."
380 REM
390 REM             SECTION 11.2: IF-THEN/POS. VS. POS.
400 REM
410 REM                    SECTION 11.2.1: EQUAL CASE.
420 IF 4=4 THEN 510
430 LET A$="X"
440 GOTO 520
450 LET B$="X"
460 GOTO 580
470 LET C$="X"
480 GOTO 600
490 LET D$="X"
500 GOTO 670
510 LET A$="T"
520 IF 4<=4 THEN 630
530 LET F$="X"
```

```
540 GOTO 640
550 LET E$="T"
560 IF 4<4 THEN 450
570 LET B$="F"
580 IF 4>4 THEN 470
590 LET C$="F"
600 IF 4<>4 THEN 490
610 LET D$="F"
620 GOTO 670
630 LET F$="T"
640 IF 4>=4 THEN 550
650 LET E$="X"
660 GOTO 560
670 PRINT "  A=1   ";TAB(23);"4 TO 4";TAB(44);A$;TAB(49);B$;TAB(54);
680 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
690 REM
700 REM                SECTION 11.2.2: NOT EQUAL CASE.
710 IF A<B THEN 800
720 LET B$="X"
730 GOTO 810
740 LET A$="X"
750 GOTO 870
760 LET C$="X"
770 GOTO 890
780 LET E$="X"
790 GOTO 960
800 LET B$="T"
810 IF A<>B THEN 920
820 LET D$="X"
830 GOTO 930
840 LET F$="T"
850 IF A=B THEN 740
860 LET A$="F"
870 IF A>B THEN 760
880 LET C$="F"
890 IF A>=B THEN 780
900 LET E$="F"
910 GOTO 960
920 LET D$="T"
930 IF A<=B THEN 840
940 LET F$="X"
950 GOTO 850
960 PRINT "  B=2   ";TAB(23);"A TO B";TAB(44);A$;TAB(49);B$;TAB(54);
970 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
980 REM
990 REM             SECTION 11.3: IF-THEN/POS. VS. NEG.
1000 IF C>N THEN 1090
1010 LET C$="X"
1020 GOTO 1100
1030 LET A$="X"
1040 GOTO 1160
1050 LET B$="X"
1060 GOTO 1180
1070 F$="X"
1080 GOTO 1250
1090 LET C$="T"
1100 IF C>=N THEN 1210
```

```
1110 LET E$="X"
1120 GOTO 1220
1130 LET D$="T"
1140 IF C=N THEN 1030
1150 LET A$="F"
1160 IF C<N THEN 1050
1170 LET B$="F"
1180 IF C<=N THEN 1070
1190 LET F$="F"
1200 GOTO 1250
1210 LET E$="T"
1220 IF C<>N THEN 1130
1230 LET D$="X"
1240 GOTO 1140
1250 PRINT "  C=3   ";TAB(23);"C TO N";TAB(44);A$;TAB(49);B$;TAB(54);
1260 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1270 REM
1280 REM             SECTION 11.4: IF-THEN/NEG. VS. POS.
1290 IF X<>Y THEN 1380
1300 LET D$="X"
1310 GOTO 1390
1320 LET A$="X"
1330 GOTO 1450
1340 LET C$="X"
1350 GOTO 1470
1360 LET E$="X"
1370 GOTO 1540
1380 LET D$="T"
1390 IF X<Y THEN 1500
1400 LET B$="X"
1410 GOTO 1510
1420 LET F$="T"
1430 IF X=Y THEN 1320
1440 LET A$="F"
1450 IF X>Y THEN 1340
1460 LET C$="F"
1470 IF X>=Y THEN 1360
1480 LET E$="F"
1490 GOTO 1540
1500 LET B$="T"
1510 IF X<=Y THEN 1420
1520 LET F$="X"
1530 GOTO 1430
1540 PRINT "  N=-3  ";TAB(23);"X TO Y";TAB(44);A$;TAB(49);B$;TAB(54);
1550 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1560 REM
1570 REM            SECTION 11.5: IF-THEN/POS. VS. ZERO.
1580 IF 2>=Z THEN 1670
1590 LET E$="X"
1600 GOTO 1680
1610 LET A$="X"
1620 GOTO 1740
1630 LET B$="X"
1640 GOTO 1760
1650 LET F$="X"
1660 GOTO 1830
1670 LET E$="T"
```

```
1680 IF 2>Z THEN 1790
1690 LET C$="X"
1700 GOTO 1800
1710 LET D$="T"
1720 IF 2=Z THEN 1610
1730 LET A$="F"
1740 IF 2<Z THEN 1630
1750 LET B$="F"
1760 IF 2<=Z THEN 1650
1770 LET F$="F"
1780 GOTO 1830
1790 LET C$="T"
1800 IF 2<>Z THEN 1710
1810 LET D$="X"
1820 GOTO 1720
1830 PRINT "  X=-2  ";TAB(23);"2 TO Z";TAB(44);A$;TAB(49);B$;TAB(54);
1840 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1850 REM
1860 REM          SECTION 11.6: IF-THEN/ZERO VS. POS.
1870 IF 0<=A THEN 1960
1880 LET F$="X"
1890 GOTO 1970
1900 LET A$="X"
1910 GOTO 2030
1920 LET C$="X"
1930 GOTO 2050
1940 LET E$="X"
1950 GOTO 2120
1960 LET F$="T"
1970 IF 0<A THEN 2080
1980 LET B$="X"
1990 GOTO 2090
2000 LET D$="T"
2010 IF 0=A THEN 1900
2020 LET A$="F"
2030 IF 0>A THEN 1920
2040 LET C$="F"
2050 IF 0>=A THEN 1940
2060 LET E$="F"
2070 GOTO 2120
2080 LET B$="T"
2090 IF 0<>A THEN 2000
2100 LET D$="X"
2110 GOTO 2010
2120 PRINT "  Y=3   ";TAB(23);"0 TO A";TAB(44);A$;TAB(49);B$;TAB(54);
2130 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
2140 REM
2150 REM          SECTION 11.7; IF-THEN/ZERO VS. ZERO.
2160 IF Z=0 THEN 2250
2170 LET A$="X"
2180 GOTO 2260
2190 LET B$="X"
2200 GOTO 2320
2210 LET C$="X"
2220 GOTO 2340
2230 LET D$="X"
2240 GOTO 2410
```

```
2250 LET A$="T"
2260 IF Z>=0 THEN 2370
2270 LET E$="X"
2280 GOTO 2380
2290 LET F$="T"
2300 IF Z>0 THEN 2190
2310 LET B$="F"
2320 IF Z<0 THEN 2210
2330 LET C$="F"
2340 IF Z<>0 THEN 2230
2350 LET D$="F"
2360 GOTO 2410
2370 LET E$="T"
2380 IF Z<=0 THEN 2290
2390 LET F$="X"
2400 GOTO 2300
2410 PRINT "  Z=0   ";TAB(23);"Z TO 0";TAB(44);A$;TAB(49);B$;TAB(54);
2420 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
2430 PRINT
2440 PRINT "     IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN"
2450 PRINT "THE SYSTEM PASSED THE TEST."
2460 PRINT
2470 PRINT "                                    END TEST."
2480 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 11

BEGIN TEST.

ASSIGNED VALUES	COMPARISON OF	TRUTH TABLE =	<	>	<>	>=	<=
A=1	4 TO 4	T	F	F	F	T	T
B=2	A TO B	F	T	F	T	F	T
C=3	C TO N	F	F	T	T	T	F
N=-3	X TO Y	F	T	F	T	F	T
X=-2	2 TO Z	F	F	T	T	T	F
Y=3	0 TO A	F	T	F	T	F	T
Z=0	Z TO 0	T	F	F	F	T	T

IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN
THE SYSTEM PASSED THE TEST.

END TEST.

12.0 THE IF-THEN STATEMENT USED TO COMPARE NEGATIVE NUMERICAL CONSTANTS

12.1 IF-THEN Comparison of a Negative Number with a Negative Number

The objective of this subsection is to compare two negative numerical values by the IF-THEN statement (see section 10 of BSR X3.60).

12.1.1 Equal Case

The objective of this test is to show that the proper comparison of a negative numerical value with itself will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output is in the first row of the truth table in the sample output for PROGRAM FILE 12.

12.1.2 Unequal Case

The objective of this test is to show that the proper comparison of a negative numerical value with another negative value, not of the same magnitude, will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <=. The test output is in the second row of the truth table.

12.2 IF-THEN Comparison of Zero with a Negative Number

The objective of this test is to show that the proper comparison of zero with a negative value will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output is in the third row of the truth table.

12.3 IF-THEN Comparison of a Negative Number with Zero

The objective of this test is to show that the proper comparison of a negative numerical value with zero will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output is in the fourth row of the truth table.

12.4 IF-THEN Comparison of Positive with the Negation of a Negative, Using Parentheses

The objective of this test is to show that the proper comparison of a positive numerical value with a numerical value that is enclosed in parentheses will be made. The constant enclosed in the parentheses represents the negative of a negative numerical value of equal magnitude to the original positive value. The comparison will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output is in the fifth row of the truth table.

12.5 IF-THEN Comparison of Negative with a

Negation of a Positive, Using Parentheses

The objective of this test is to show that a proper comparison of a negative numerical value with a numerical value enclosed in parentheses will be made. The value in parentheses represents the negative of a positive numerical value of equal magnitude to the original negative value. The comparison will be made by an IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output is in the sixth row in the truth table.

```
********************
* PROGRAM FILE 12 *
********************
```

```
10 PRINT "PROGRAM FILE 12"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                                   BEGIN TEST."
100 PRINT
110 PRINT
120 LET A$="="
130 LET B$="<"
140 LET C$=">"
150 LET D$="<>"
160 LET E$=">="
170 LET F$="<="
180 LET A=1
190 LET B=2
200 LET C=3
210 LET L=-1
220 LET M=-2
230 LET N=-3
240 LET X=-2
250 LET Y=3
260 LET Z=0
270 PRINT "ASSIGNED";TAB(20);"COMPARSION OF";TAB(54);"TRUTH TABLE"
280 PRINT " VALUES ";TAB(44);A$;TAB(49);B$;TAB(54);C$;TAB(59);D$;
290 PRINT TAB(65);E$;TAB(70);F$
300 PRINT
310 REM
320 REM             SECTION 12.1: IF-THEN/NEG. VS. NEG.
330 REM
340 REM                 SECTION 12.1.1: EQUAL CASE.
350 IF M=X THEN 440
360 LET A$="X"
370 GOTO 450
380 LET B$="X"
```

```
390 GOTO 510
400 LET C$="X"
410 GOTO 530
420 LET D$="X"
430 GOTO 600
440 LET A$="T"
450 IF M>=X THEN 560
460 LET E$="X"
470 GOTO 570
480 LET F$="T"
490 IF M<X THEN 380
500 LET B$="F"
510 IF M>X THEN 400
520 LET C$="F"
530 IF M<>X THEN 420
540 LET D$="F"
550 GOTO 600
560 LET E$="T"
570 IF M<=X THEN 480
580 LET F$="F"
590 GOTO 490
600 PRINT "  B=2   ";TAB(23);"M TO X";TAB(44);A$;TAB(49);B$;TAB(54);
610 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
620 REM
630 REM               SECTION 12.1.2: NOT EQUAL CASE.
640 IF -3<L THEN 730
650 LET B$="X"
660 GOTO 740
670 LET A$="X"
680 GOTO 800
690 LET C$="X"
700 GOTO 820
710 LET E$="X"
720 GOTO 890
730 LET B$="T"
740 IF -3<>L THEN 850
750 LET D$="X"
760 GOTO 860
770 LET F$="T"
780 IF -3=L THEN 670
790 LET A$="F"
800 IF -3>L THEN 690
810 LET C$="F"
820 IF -3>=L THEN 710
830 LET E$="F"
840 GOTO 890
850 LET D$="T"
860 IF -3<=L THEN 770
870 LET F$="X"
880 GOTO 780
890 PRINT "  L=-1  ";TAB(22);"-3 TO L";TAB(44);A$;TAB(49);B$;TAB(54);
900 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
910 REM
920 REM             SECTION 12.2: IF-THEN/ZERO VS. NEG.
930 IF Z>X THEN 1020
940 LET C$="X"
950 GOTO 1030
```

```
960 LET A$="X"
970 GOTO 1090
980 LET B$="X"
990 GOTO 1110
1000 LET F$="X"
1010 GOTO 1180
1020 LET C$="T"
1030 IF Z<>X THEN 1140
1040 LET D$="X"
1050 GOTO 1150
1060 LET E$="T"
1070 IF Z=X THEN 960
1080 LET A$="F"
1090 IF Z<X THEN 980
1100 LET B$="F"
1110 IF Z<=X THEN 1000
1120 LET F$="F"
1130 GOTO 1180
1140 LET D$="T"
1150 IF Z>=X THEN 1060
1160 LET E$="X"
1170 GOTO 1090
1180 PRINT "  M=-2  ";TAB(23);"Z TO X";TAB(44);A$;TAB(49);B$;TAB(54);
1190 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1200 REM
1210 REM           SECTION 12.3: IF-THEN/NEG. VS. ZERO.
1220 IF M<>Z THEN 1310
1230 LET D$="X"
1240 GOTO 1320
1250 LET A$="X"
1260 GOTO 1380
1270 LET C$="X"
1280 GOTO 1400
1290 LET E$="X"
1300 GOTO 1470
1310 LET D$="T"
1320 IF M<Z THEN 1430
1330 LET B$="X"
1340 GOTO 1440
1350 LET F$="T"
1360 IF M=Z THEN 1250
1370 LET A$="F"
1380 IF M>Z THEN 1270
1390 LET C$="F"
1400 IF M>=Z THEN 1290
1410 LET E$="F"
1420 GOTO 1470
1430 LET B$="T"
1440 IF M<=Z THEN 1350
1450 LET F$="X"
1460 GOTO 1360
1470 PRINT "  N=-3  ";TAB(23);"M TO Z";TAB(44);A$;TAB(49);B$;TAB(54);
1480 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1490 REM
1500 REM       SECTION 12.4: IF-THEN/POS. VS. (NEG.OF NEG.).
1510 IF 3=(-N) THEN 1600
1520 LET A$="X"
```

```
1530 GOTO 1610
1540 LET B$="X"
1550 GOTO 1670
1560 LET C$="X"
1570 GOTO 1690
1580 LET D$="X"
1590 GOTO 1750
1600 LET A$="T"
1610 IF 3>=(-N) THEN 1720
1620 LET E$="X"
1630 GOTO 1730
1640 LET F$="T"
1650 IF 3<(-N) THEN 1540
1660 LET B$="F"
1670 IF 3>(-N) THEN 1560
1680 LET C$="F"
1690 IF 3<>(-N) THEN 1580
1700 LET D$="F"
1710 GOTO 1750
1720 LET E$="T"
1730 IF 3<=(-N) THEN 1640
1740 LET F$="X"
1750 PRINT "  X=-2  ";TAB(23);"3 TO (-N)";TAB(44);A$;TAB(49);B$;TAB(54);
1760 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1770 REM
1780 REM       SECTION 12.5: IF-THEN/NEG. VS. (NEG. OF POS.).
1790 IF M=(-B) THEN 1880
1800 LET A$="X"
1810 GOTO 1890
1820 LET B$="X"
1830 GOTO 1950
1840 LET C$="X"
1850 GOTO 1970
1860 LET D$="X"
1870 GOTO 2040
1880 LET A$="T"
1890 IF M>=(-B) THEN 2000
1900 LET E$="X"
1910 GOTO 2010
1920 LET F$="T"
1930 IF M<(-B) THEN 1820
1940 LET B$="F"
1950 IF M>(-B) THEN 1840
1960 LET C$="F"
1970 IF M<>(-B) THEN 1860
1980 LET D$="F"
1990 GOTO 2040
2000 LET E$="T"
2010 IF M<=(-B) THEN 1920
2020 LET F$="X"
2030 GOTO 1930
2040 PRINT "  Z=0   ";TAB(23);"M TO (-B)";TAB(44);A$;TAB(49);B$;TAB(54);
2050 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
2060 PRINT
2070 PRINT "     IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN"
2080 PRINT "THE SYSTEM PASSED THE TEST."
2090 PRINT
```

```
2100 PRINT
2110 PRINT "                                   END TEST."
2120 PRINT
2130 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 12

BEGIN TEST.

ASSIGNED VALUES	COMPARSION OF	TRUTH TABLE =	<	>	<>	>=	<=
B=2	M TO X	T	F	F	F	T	T
L=-1	-3 TO L	F	T	F	T	F	T
M=-2	Z TO X	F	F	T	T	T	F
N=-3	M TO Z	F	T	F	T	F	T
X=-2	3 TO (-N)	T	F	F	F	T	T
Z=0	M TO (-B)	T	F	F	F	T	T

IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN
THE SYSTEM PASSED THE TEST.

END TEST.

13.0 IF-THEN COMPARISON OF NEGATIVE CONSTANTS (CONT) AND VARIABLES AVOIDING PARENTHESES

This section continues a case-by-case analysis of the IF-THEN control statement that began in section 11. The emphasis in this section will be on comparing assigned numeric variables. The user is referred to section 6 and 10 in BSR X3.60.

13.1 IF-THEN Comparison of an Assigned Negative with a Negative Constant, Without Parentheses

The objective of this test is to show that the proper comparison of a negative numerical value assigned to a simple variable against another negative value not assigned to a simple variable or enclosed in parentheses will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output falls in the first row of the truth table output for PROGRAM FILE 13.

13.2 IF-THEN Comparison of a Negative Variable with a Negative Constant, Without Parentheses

The objective of this test is to show that the proper comparison of the negative of a variable, assigned a positive numerical value, with a negative value not assigned to a simple variable or enclosed in parentheses, will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output falls in the second row of the truth table.

13.3 IF-THEN Comparison of Positive Variable with Negative Variable, Without Parentheses

The objective of this test is to show that the proper comparison of a variable, assigned a positive value, with the negative of a variable assigned a value of the same magnitude as the positive value, will be made by the IF-THEN statement for each of the relation symbols =, <, >, <>, >=, and <= . The test output falls in the third row of the truth table.

```
********************
* PROGRAM FILE 13 *
********************
```

```
1 PRINT "PROGRAM FILE 13"
20 PRINT
30 PRINT
```

```
40 PRINT
60 LET A$="="
70 LET B$="<"
80 LET C$=">"
90 LET D$="<>"
100 LET E$=">="
110 LET F$="<="
120 LET A=1
130 LET B=2
140 LET N=-3
150 LET X=-2
160 PRINT "ASSIGNED";TAB(20);"COMPARISON OF";TAB(54);"TRUTH TABLE"
170 PRINT " VALUES ";TAB(44);A$;TAB(49);B$;TAB(54);C$;TAB(59);D$;
180 PRINT TAB(65);E$;TAB(70);F$
190 PRINT
200 REM
210 REM SECTION 13.1: IF-THEN/ASSIGNED NEG. VS. NEG. CONSTANT, ABSENT
220 REM               PARENTHESES.
230 IF N=-3 THEN 320
240 LET A$="X"
250 GOTO 330
260 LET  B$="X"
270 GOTO 390
280 LET  C$="X"
290 GOTO 410
300 LET  D$="X"
310 GOTO 480
320 LET  A$="T"
330 IF N>=-3 THEN 440
340 LET  E$="X"
350 GOTO 450
360 LET  F$="T"
370 IF N<-3 THEN 260
380 LET  B$="F"
390 IF N>-3 THEN 280
400 LET  C$="F"
410 IF N<>-3 THEN 300
420 LET  D$="F"
430 GOTO 480
440 LET  E$="T"
450 IF N<=-3 THEN 360
460 LET  F$="X"
470 GOTO 370
480 PRINT "  A=1   ";TAB(23);"N TO -3";TAB(44);A$;TAB(49);B$;TAB(54);
490 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
500 REM
510 REM SECTION 13.2: IF-THEN/NEG. VARIABLE VS. NEG. CONSTANT, ABSENT
520 REM               PARENTHESES.
530 IF -A=-1 THEN 620
540 LET  A$="X"
550 GOTO 630
560 LET  B$="X"
570 GOTO 690
580 LET  C$="X"
590 GOTO 710
600 LET  D$="X"
610 GOTO 780
```

```
620 LET  A$="T"
630 IF -A>=-1 THEN 740
640 LET  E$="X"
650 GOTO 750
660 LET  F$="T"
670 IF -A<-1 THEN 560
680 LET  B$="F"
690 IF -A>-1 THEN 580
700 LET  C$="F"
710 IF -A<>-1 THEN 600
720 LET  D$="F"
730 GOTO 780
740 LET  E$="T"
750 IF -A<=-1 THEN 660
760 LET  F$="X"
770 GOTO 670
780 PRINT "  B=2   ";TAB(22);"-A TO -1";TAB(44);A$;TAB(49);B$;TAB(54);
790 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
800 REM
810 REM SECTION 13.3: IF-THEN/POS. VARIABLE VS. NEG. VARIABLE, ABSENT
820 REM                PARENTHESES.
830 IF B=-X THEN 920
840 LET  A$="X"
850 GOTO 930
860 LET  B$="X"
870 GOTO 990
380 LET  C$="X"
890 GOTO 1010
900 LET  D$="X"
910 GOTO 1080
920 LET  A$="T"
930 IF B>=-X THEN 1040
940 LET  E$="X"
950 GOTO 1050
960 LET  F$="T"
970 IF B<-X THEN 860
980 LET  B$="F"
990 IF B>-X THEN 880
1000 LET  C$="F"
1010 IF B<>-X THEN 900
1020 LET  D$="F"
1030 GOTO 1080
1040 LET  E$="T"
1050 IF B<=-X THEN 960
1060 LET  F$="X"
1070 GOTO 970
1080 PRINT "  N=-3  ";TAB(23);"B TO -X";TAB(44);A$;TAB(49);B$;TAB(54);
1090 PRINT C$;TAB(59);D$;TAB(65);E$;TAB(70);F$
1100 PRINT
1110 PRINT "     IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN"
1120 PRINT "THE SYSTEM PASSED THE TEST."
1130 PRINT
1140 PRINT "                                    END TEST."
1150 PRINT
1160 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 13

ASSIGNED VALUES	COMPARISON OF	TRUTH TABLE =	<	>	<>	>=	<=
A=1	N TO -3	T	F	F	F	T	T
B=2	-A TO -1	T	F	F	F	T	T
N=-3	B TO -X	T	F	F	F	T	T

IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN THE SYSTEM PASSED THE TEST.

END TEST.

14.0 COMPARING QUOTED STRINGS AND STRING VARIABLES

The program for this test prints four columns. The first column prints the names of certain string variables used in the program. The second column lists the strings assigned to these variables, within the program, to the string variables in the first column. The third column lists the comparisons made and whether they are between string constants or assigned string variables. The last column prints the truth table for this test program. The user need only look for X's printed in the table. If there are none the system has passed the test.

The standard states that string comparisons can only be made for equality (=) and inequality (<>) (see section 10.2 and 10.4 of BSR X3.60). Tests of collating sequences are inapplicable for Minimal BASIC.

14.1 Comparing Identical Strings

The objective of this test is to show that a comparison of two strings having the same sequential order of characters will be made by the IF-THEN statement. This test generates the values in the first row as the column four truth table output in the sample output for this section.

14.2 Strings of Equal Length but Different Characters

The objective of this test is to show that a comparison of two different strings having the same number of characters will be made by the IF-THEN statement. This test outputs the values in the second row of the truth table.

14.3 Strings of the Same Length which have the Same Characters but not in the Same Sequential Order

The objective of this test is to show that a comparison of two strings, having the same character set, but not in the same sequential order, will be made by the IF-THEN statement. The output for this test lies in the third row of the truth table.

14.4 Leading Spaces

The objective of this test is to show that a proper comparison of two strings having the same non-space characters in the same sequential order, but with leading spaces for one of the strings, will be performed. The output for this test lies in the fourth row of the truth table. According to the standard these strings are not equal since spaces within quoted strings are considered significant.

14.5 Trailing Spaces

The objective of this test is to show that a proper comparison of two strings having the same non-space characters in the same sequential order, but with trailing spaces used in one of the strings, will also be performed. The output for this test lies in the fifth row of the truth table. Again, these two quoted strings are not equal.

14.6 Leading and Trailing Spaces

The objective of this test is to show the effect of using both leading and trailing spaces in the comparison of two strings. The strings in this test appear the same except for the leading and trailing spaces in one of them. The result should be that the two strings are not equal since spaces in strings are significant. The output for this test lies in the sixth row of the truth table.

14.7 A Variety of Unequal String

The objective of this test is to further verify that the relation of equality holds between two strings if and only if the two strings have the same length and contain identical sequences of characters. This test compares various strings of different characters as well as of different lengths. The output for this test lies in the seventh through ninth rows of the truth table.

```
********************
* PROGRAM FILE 14 *
********************
```

```
10 PRINT "PROGRAM FILE 14"
60 PRINT
70 PRINT
80 PRINT
90 LET D$="="
100 LET G$="<>"
110 LET A$="ZONE"
120 LET B$="NESTING"
130 LET C$="IMPLEMENTATION"
140 LET L$="NEST"
150 LET M$="KEYWORD"
160 LET N$="MARGIN"
170 LET Y$="LINE"
180 PRINT
190 PRINT "     IN THE COMPARISON COLUMN BELOW, THE UNDERLINE CHARAC-"
200 PRINT "TER IS USED TO REPRESENT THE SPACE CHARACTER."
210 PRINT
220 PRINT
230 PRINT " STRING  ";TAB(23);"ASSIGNED";TAB(44);"COMPARSION";TAB(64);
```

```
240 PRINT "TRUTH"
250 PRINT "VARIABLES";TAB(22);"CHARACTERS";TAB(48);"OF";TAB(64);"TABLE"
260 PRINT TAB(64);D$;TAB(67);G$
270 PRINT
280 REM
290 REM SECTION 14.1: STRINGS OF EQUAL LENGTH WITH THE SAME SEQUENTIAL
300 REM               CHARACTERS VS. EACH OTHER.
310 IF "MAY"="MAY" THEN 360
320 LET D$="X"
330 GOTO 370
340 LET G$="X"
350 GOTO 390
360 LET D$="T"
370 IF "MAY"<>"MAY" THEN 340
380 LET G$="F"
390 PRINT "   A$    ";TAB(25);"ZONE";TAB(44);"MAY TO MAY";TAB(64);D$;
400 PRINT TAB(68);G$
410 REM
420 REM SECTION 14.2: STRINGS OF EQUAL LENGTH BUT DIFFERENT CHARACTERS.
430 IF A$<>Y$ THEN 480
440 LET G$="X"
450 GOTO 490
460 LET D$="X"
470 GOTO 510
480 LET G$="T"
490 IF A$=Y$ THEN 460
500 LET D$="F"
510 PRINT "   B$    ";TAB(23);"NESTING";TAB(45);"A$ TO Y$";TAB(64);D$;
520 PRINT TAB(68);G$
530 REM
540 REM SECTION 14.3: STRINGS OF THE SAME LENGTH WHICH HAVE THE SAME
550 REM               CHARACTERS BUT NOT SEQUENTIALLY ORDERED THE SAME.
560 IF "MAY"="YAM" THEN 610
570 LET D$="F"
580 GOTO 620
590 LET G$="T"
600 GOTO 640
610 LET D$="X"
620 IF "MAY"<>"YAM" THEN 590
630 LET G$="X"
640 PRINT "   C$    ";TAB(20);"IMPLEMENTATION";TAB(44);"MAY TO YAM";
650 PRINT TAB(64);D$;TAB(68);G$
660 REM
670 REM SECTION 14.4: TESTING THE SIGNIFICANCE OF LEADING SPACES IN
680 REM               STRING COMPARISONS.
690 IF "MAY"<>"   MAY" THEN 740
700 LET G$="X"
710 GOTO 750
720 D$="X"
730 GOTO 770
740 LET G$="T"
750 IF "MAY"="   MAY" THEN 720
760 D$="F"
770 PRINT "   L$    ";TAB(25);"NEST";TAB(44);"MAY TO ___MAY";TAB(64);
780 PRINT D$;TAB(68);G$
790 REM
800 REM SECTION 14.5: TESTING THE SIGNIFICANCE OF TRAILING SPACES IN
```

```
810 REM                STRING COMPARISONS.
820 IF "MAY"="MAY    " THEN 850
830 LET D$="F"
840 GOTO 860
850 LET D$="X"
860 IF "MAY"<>"MAY    " THEN 890
870 LET G$="X"
880 GOTO 900
890 LET G$="T"
900 PRINT "   M$     ";TAB(23);"KEYWORD";TAB(44);"MAY TO MAY___";
910 PRINT TAB(64);D$;TAB(68);G$
920 REM
930 REM SECTION 14.6: TESTING LEADING AND TRAILING SPACES TOGETHER IN
940 REM                STRING COMPARISONS.
950 IF "MAY"<>"  MAY   " THEN 1000
960 LET G$="X"
970 GOTO 1010
980 LET D$="X"
990 GOTO 1030
1000 LET G$="T"
1010 IF "MAY"="  MAY   " THEN 980
1020 LET D$="F"
1030 PRINT "   N$     ";TAB(24);"MARGIN";TAB(44);"MAY TO ___MAY___";
1040 PRINT TAB(64);D$;TAB(68);G$
1050 REM
1060 REM SECTION 14.7: TESTING A VARIETY OF UNEQUAL STRINGS OF LENGTH
1070 REM                AND CHARACTERS FOR COMPARISONS.
1080 IF C$=B$ THEN 1210
1090 LET D$="F"
1100 GOTO 1220
1110 LET G$="T"
1120 IF C$<>N$ THEN 1250
1130 LET E$="X"
1140 GOTO 1260
1150 LET F$="X"
1160 GOTO 1280
1170 LET H$="T"
1180 IF "KEY"=M$ THEN 1310
1190 LET I$="F"
1200 GOTO 1320
1210 LET D$="X"
1220 IF L$<>B$ THEN 1110
1230 LET G$="X"
1240 GOTO 1120
1250 LET E$="T"
1260 IF C$=N$ THEN 1150
1270 LET F$="F"
1280 IF "KEY"<>M$ THEN 1170
1290 LET H$="X"
1300 GOTO 1180
1310 LET I$="X"
1320 PRINT "   Y$     ";TAB(25);"LINE";TAB(45);"L$ TO B$";TAB(64);D$;
1330 PRINT TAB(68);G$
1340 PRINT TAB(45);"C$ TO N$";TAB(64);D$;TAB(68);G$
1350 PRINT TAB(44);"KEY TO M$";TAB(64);D$;TAB(68);G$
1360 PRINT
1370 PRINT "     IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN"
```

```
1380 PRINT "THE SYSTEM PASSED THE TEST."
1390 PRINT
1400 PRINT "                                    END TEST."
1410 PRINT
1420 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 14

IN THE COMPARISON COLUMN BELOW, THE UNDERLINE CHARACTER IS USE TO REPRESENT THE SPACE CHARACTER.

STRING VARIABLES	ASSIGNED CHARACTERS	COMPARSION OF	TRUTH TABLE =	TRUTH TABLE <>
A$	ZONE	MAY TO MAY	T	F
B$	NESTING	A$ TO Y$	F	T
C$	IMPLEMENTATION	MAY TO YAM	F	T
L$	NEST	MAY TO ___MAY	F	T
M$	KEYWORD	MAY TO MAY___	F	T
N$	MARGIN	MAY TO ___MAY___	F	T
Y$	LINE	L$ TO B$	F	T
		C$ TO N$	F	T
		KEY TO M$	F	T

IF NO X(S) APPEARED IN THE ABOVE TRUTH TABLE, THEN THE SYSTEM PASSED THE TEST.

END TEST.

15.0 TEST OF IF-THEN TRANSFER TO ILLEGAL LINE NUMBER

The objective of this test is to verify that the execution of an IF-THEN statement, referring to a non-existent, line will cause program execution to be suspended by the system. To continue the program, execution should require some user-directed restart procedures which would be implementation-dependent. In the output for this test there should be an implementation specific diagnostic for an error.

****
* PROGRAM FILE 15 *
****

```
10 PRINT "PROGRAM FILE 15"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "     SECTION 15.0: FATAL ERROR CHECK ON IF-THEN STATEMENT."
100 PRINT
110 PRINT
120 PRINT
130 PRINT "     THE OBJECTIVE OF THIS SECTION IS TO USE AN IF-THEN"
140 PRINT "STATEMENT WHICH REFERS TO A NON-EXISTENT LINE-NUMBER.  IF"
150 PRINT "THE SYSTEM RECOGNIZES THIS AS A FATAL ERROR (THAT IS, SUS-"
160 PRINT "PENDING PROGRAM EXECTION SUCH THAT USER-DIRECTED RESTART"
170 PRINT "PROCEDURES ARE REQUIRED), THEN THE TEST WILL HAVE PASSED."
180 PRINT
190 PRINT
200 PRINT
210 PRINT "                                BEGIN TEST."
220 PRINT
230 LET A=5
240 IF A<>5 THEN 99
250 PRINT "TEST FAILED."
260 PRINT
270 PRINT "                                 END TEST."
280 PRINT
290 END
```

****
* SAMPLE OUTPUT *
****

```
?UNDEFINED LINE NUMBER 99 IN LINE 240
```

16.0 LINE LABELS WITH AND WITHOUT LEADING ZEROES

This unit tests the unique line-number which serves as a label for the statement on the line. The labels at the beginning of each line have the following restrictions: (1) they do not contain and are not preceded by spaces, and (2) they are within the range 1 to 9999 inclusive. Leading zeroes have ho effect (see section 4 of BSR X3.60). These tests are performed with GOTO, IF-THEN, and PRINT statements only. The GOTO Statement uses line numbers which may or may not have leading zeroes and these same line numbers when used as line labels may have fewer leading zeroes than in the GOTO statements. The output for this test is in the form of messages to the user that describe the sequence of label tests as they proceed and whether each has passed or failed. A final "passed" or "failed" message is given at the end.

```
********************
* PROGRAM FILE 16 *
********************
```

```
1 PRINT "PROGRAM FILE 16"
16 PRINT
19 PRINT
22 PRINT
25 PRINT "    SECTION 16.0: LINE LABELS, WITH/WITHOUT LEADING ZEROES."
28 PRINT
31 PRINT "                                BEGIN TEST."
34 PRINT
37 LET M=1
40 PRINT "       TEST 1, GOTO USING 1 LEADING ZEROES ON LABEL."
43 GOTO 058
0046 PRINT "       TEST 2 PASSED, LABEL GAINED 2 LEADING ZEROES."
49 LET Z=N
52 PRINT "       TEST 3, GOTO USING 2 LEADING ZEROES ON LABEL."
55 GOTO 0070
58 PRINT "     TEST 1 PASSED, LABEL LOST ALL OF ITS LEADING ZEROES."
61 LET N=M
64 PRINT "       TEST 2, GOTO USING NO LEADING ZEROES ON LABEL."
67 GOTO 46
070 PRINT "       TEST 3 PASSED, LABEL LOST 1 OF ITS LEADING ZEROES."
73 IF Z=1 THEN 82
76 PRINT "                        LEADING ZEROES TEST FAILED."
79 GOTO 91
82 PRINT "                        LEADING ZEROES TEST PASSED."
85 PRINT
88 PRINT "                                END TEST."
91 PRINT
9999 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 16

```
    SECTION 16.0: LINE LABELS, WITH/WITHOUT LEADING ZEROES.

                          BEGIN TEST.

      TEST 1, GOTO USING 1 LEADING ZEROES ON LABEL.
    TEST 1 PASSED, LABEL LOST ALL OF ITS LEADING ZEROES.
      TEST 2, GOTO USING NO LEADING ZEROES ON LABEL.
      TEST 2 PASSED, LABEL GAINED 2 LEADING ZEROES.
      TEST 3, GOTO USING 2 LEADING ZEROES ON LABEL.
      TEST 3 PASSED, LABEL LOST 1 OF ITS LEADING ZEROES.

                  LEADING ZEROES TEST PASSED.

                          END TEST.
```

17.0 ORDER OF LINES - TWO LINES WITH THE SAME LINE NUMBER

The objective of the next two tests is to determine whether the ordering of line numbers and/or using duplicate line numbers will be recognized by implementations as an error or be adjusted by a local editing facility. In the present test the implementation is asked to accept a program with a duplicate line at 260. There are two possible cases here. First, if the system does not have a program editor, then a diagnostic should be given. If, however, the system does automatically edit programs, then an output to the test program should be given without also giving an error diagnostic.

```
********************
* PROGRAM FILE 17 *
********************
```

```
10 PRINT "PROGRAM FILE 17"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                SECTION 17.0: TEST OF LINES WITH SAME NUMBER."
100 PRINT
110 PRINT "     NOTE:  LOCAL EDITING FACILITIES MAY ALLOW"
120 PRINT "FOR THE ENTRY OF STATEMENT LINES IN ANY ORDER AND ALSO"
130 PRINT "ALLOW FOR DUPLICATE LINE-NUMBERS AND LINES CONTAINING"
140 PRINT "ONLY A LINE-NUMBER.  THIS TEST MAY PRODUCE AN"
150 PRINT "ERROR DIAGNOSTIC ON OTHER SYSTEMS."
160 PRINT
170 PRINT
180 PRINT
230 PRINT "                                BEGIN TEST."
240 PRINT
260 LET Y=1111
260 LET Y=9999
270 PRINT "Y=";Y
280 PRINT
290 PRINT "     SINCE A VALUE FOR Y HAS BEEN PRINTED IT CAN BE AS-"
300 PRINT "SUMED THAT THIS SYSTEM DOES NOT RECOGNIZE DUPLICATE LINE-"
310 PRINT "NUMBERS AS AN ERROR.   IF Y= 9999 THEN THE LAST LINE"
320 PRINT "260 WAS RETAINED BY THE LOCAL EDITOR."
330 PRINT
340 PRINT "                                  END TEST."
350 PRINT
360 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

```
PROGRAM FILE 17

                SECTION 17.0: TEST OF LINES WITH SAME NUMBER

     NOTE:  LOCAL EDITING FACILITIES MAY ALLOW
FOR THE ENTRY OF STATEMENT LINES IN ANY ORDER AND ALSO
ALLOW FOR DUPLICATE LINE-NUMBERS AND LINES CONTAINING
ONLY A LINE-NUMBER.  THIS TEST MAY PRODUCE AN
ERROR DIAGNOSTIC ON OTHER SYSTEMS.


                          BEGIN TEST.

Y= 9999

     SINCE A VALUE FOR Y HAS BEEN PRINTED IT CAN BE AS-
SUMED THAT THIS SYSTEM DOES NOT RECOGNIZE DUPLICATE LINE-
NUMBERS AS AN ERROR.  IF Y= 9999 THEN THE LAST LINE
260 WAS RETAINED BY THE LOCAL EDITOR.

                          END TEST.
```

18.0 ORDER OF LINES - LINES OUT OF ORDER

This test verifies whether the test system will either recognize lines out of order as an error when no local editor is available or have a program editor that reorders the lines (see section 4.6 in BSR X3.60). On output, there should be a diagnostic for systems without editors or two numbers in order A1= 1111 and A2= 9999. If these numbers are produced in reverse order then the local editor did not reorder the lines. The test language processor executed one line after the other without regard to line order.

****
* PROGRAM FILE 18 *
****

```
10 PRINT "PROGRAM FILE 18"
60 PRINT
70 PRINT
80 PRINT
90 PRINT "                    SECTION 18.0: LINES OUT OF ORDER."
100 PRINT
110 PRINT
120 PRINT "                                BEGIN TEST."
130 PRINT
150 LET A=9999
145 PRINT "A1=";A
140 LET A=1111
160 PRINT "A2=";A
170 PRINT
180 PRINT "TWO NUMBERS SHOULD HAVE BEEN PRINTED AS:"
190 PRINT "A1= 1111"
200 PRINT "A2= 9999"
210 PRINT "IF THIS OUTPUT WAS REVERSED THEN SYSTEM DID"
220 PRINT "NOT RECOGNIZE LINES OUT OF ORDER."
230 PRINT
240 PRINT "                                  END TEST."
250 PRINT
260 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 18

SECTION 18.0: LINES OUT OF ORDER.

BEGIN TEST.

A1= 1111
A2= 9999

TWO NUMBERS SHOULD HAVE BEEN PRINTED AS:
A1= 1111
A2= 9999
IF THIS OUTPUT WAS REVERSED THEN SYSTEM DID
NOT RECOGNIZE LINES OUT OF ORDER.

END TEST.

19.0 THE ABS FUNCTION AND ELEMENTARY NUMERICAL EXPRESSIONS USING CONSTANTS

The objective of the next several test sections is to verify that numeric expressions can be constructed from simple variables and constants using the operations of addition, subtraction, multiplication, division, and involution. They further verify that simple variables and constants can be used as arguments for an absolute value function reference. One item the user should be wary of in these tests is that a difference in format of output does not force a test failure. For example, there are numbers that should be printed in NR2 format that some non-conforming systems might print in NR3. The non-conformance would only be in formatting of numbers which would have been tested previously. The user need only check the test output for asterisks after the last ouput column, where numerical errors are printed. At the lowest level, the tests are passed if there are no diagnostics for the ABS function or for the expressions. That is, the host system can evaluate them. However, in order to pass the test entirely, there should be no asterisks. If there are, this means that some operation has generated a result that is out of tolerance. The user should be familiar at this point with sections 7 and 8 of BSR X3.60.

These tests further extend the capability of the IF-THEN statement, since there is a comparison of the ABS function of an argument against a constant. The successful evaluation of this form of the IF-THEN Statement demonstrates that an intrinsic function can be referenced in the branch condition.

19.1 The ABS Function

The objective of this test section is to introduce a reference to the absolute value function. The arguments used will be the simplest of the numerical expressions. The expressions will consist of only one term and these terms will consist of only one factor.

Even if two results agree to within six digits, the actual machine representation of the test system may use a register or memory storage location that holds more bits than required for a six digit representation. The errors printed only represent differences between computed results and converted program constants. The only errors flagged are those that differ by a unit or more in the sixth significant digit of the result. Absolute error means the absolute value of the difference between the test system computed result and the comparison result, which is the true result rounded to six significant digits.

19.1.1 Using Numerical Constants for Arguments

The objective of this test is to verify that numerical constants can be used as arguments in an absolute value function. This test uses both negative and positive numerical constants as the arguments in an absolute value function reference. That is, both negative and positive NR1, NR2, and NR3 constants are used for the arguments. The output format is in three columns. The first column is labeled "ARGUMENT" and represents the constant used as the argument. The second column is labeled "TRUE EVALUATION", and the third column is labeled "SYSTEM EVALUATION". In the second column we

list the standard conforming output expected and in the third column we list the evaluations of the absolute value function on the system being tested. An asterisk should appear beside any evaluation that does not conform to the standard.

19.1.2 Using Assigned Variables for Arguments

The objective of this test is to verify that simple variables can be used as the arguments for an absolute value function. This test uses both negative and positive numbers assigned to the simple variables. Specifically, both negative and positive NR1, NR2, and NR3 constants have been assigned to the simple variables, and these simple variables are then used as arguments in absolute value function referencing. The test uses the same output format as in test 19.1.1.

19.2 Primitive Operations

The remainder of this section and the next two sections are devoted to verifying that numerical expressions can be constructed from two terms (involving addition or subtraction), two factors (involving multiplication or division), or from a base and an exponent (involving involution), using constants only.

19.3 Using Numerical Constants in Addition

This test initiates the construction of numerical expressions from numerical constants. This test and the next two proceed through each of the operations of addition, subtraction, multiplication, division, and involution. Each test is structured to use the same type of constants for each operation, that is, NR1 constants are not used with NR2 constants, NR2 constants are not used with NR3 constants, etc.

The objective of this test subsection is to verify that the operation of addition between two numerical constants will yield at least six decimal digits of precision. This routine tests the addition of two NR1 constants, two NR2 constants and two NR3 constants. The output of this test falls into four columns of which the first column is labeled "1st. ADDEND PLUS 2nd. ADDEND", the second column is labeled "REQUIRED SUM", the third column is labeled "SUM OF SYSTEM", and the fourth column is labeled "ABSOLUTE ERROR". The first column lists the pairs of numerical constants that are added; the second column lists the standard conforming expected output; the third column lists the sums of the constants as evaluated by the system being tested; and the fourth column lists the absolute difference between the expected values and the system generated values. If the implementation did not maintain at least six decimal digits of precision, an asterisk will appear to the right of the fourth column.

****
* PROGRAM FILE 19 *
****

```
0010 PRINT "PROGRAM FILE 19"
0020 PRINT
0030 PRINT
0040 PRINT
0090 PRINT "                    SECTION 19.1: THE ABS FUNCTION."
0100 PRINT
0110 PRINT
0120 PRINT "                       SECTION 19.1.1"
0130 PRINT
0140 PRINT "              (USING NUMERICAL CONSTANTS FOR ARGUMENTS.)"
0150 PRINT
0160 PRINT
0170 PRINT "                                BEGIN TEST."
0180 PRINT
0190 LET A=ABS(-10)
0200 IF A<>10 THEN 230
0210 LET A$=" "
0220 GOTO 240
0230 LET A$="*"
0240 LET B=ABS(15)
0250 IF B<>15 THEN 280
0260 LET B$=" "
0270 GOTO 290
0280 LET B$="*"
0290 LET A1=ABS(-.5)
0300 IF A1<>.5 THEN 330
0310 LET C$=" "
0320 GOTO 340
0330 LET C$="*"
0340 LET B1=ABS(.125)
0350 IF B1<>.125 THEN 380
0360 LET D$=" "
0370 GOTO 390
0380 LET D$="*"
0390 LET A2=ABS(-62.5E-3)
0400 IF A2<>62.5E-3 THEN 430
0410 GOTO 440
0420 GOTO 440
0430 LET E$="*"
0440 LET B2=ABS(312.5E-4)
0450 IF B2<>312.5E-4 THEN 480
0460 LET F$=" "
0470 GOTO 490
0480 LET F$="*"
0490 PRINT
0500 PRINT "     EACH EVALUATED FAILURE OF THE ABS FUNCTION WILL BE DE-"
0510 PRINT "NOTED BY AN ASTERISK BEING PRINTED ON THAT COMPARATIVE ROW"
0520 PRINT "OF OUTPUT.  TEST PASSED IF THERE ARE NOT ANY ASTER-"
0530 PRINT "ISKS."
0540 PRINT
0550 PRINT
0560 PRINT "        ","   TRUE   ","  SYSTEM  "
0570 PRINT "ARGUMENT","EVALUATION","EVALUATION"
0580 PRINT
0590 PRINT "-10 "," 10 ",A;A$
0600 PRINT " 15 "," 15 ",B;B$
0610 PRINT "-.5 "," .5 ",A1;C$
```

```
1190 PRINT
1200 PRINT "VALUE OF","   TRUE   "," SYSTEM  "
1210 PRINT "ARGUMENT","EVALUATION","EVALUATION"
1220 PRINT
1230 PRINT "-10 "," 10 ",M1;A$
1240 PRINT " 15 "," 15 ",M2;B$
1250 PRINT "-.5 "," .5 ",M3;C$
1260 PRINT " .125 "," .125 ",M4;D$
1270 PRINT "-62.5E-3 "," .0625 ",M5;E$
1280 PRINT " 312.5E-4 "," .03125 ",M6;F$
1290 PRINT
1300 PRINT "                                END TEST."
1310 PRINT
1320 PRINT
1330 PRINT
1350 PRINT
1360 PRINT
1370 PRINT
1380 PRINT "                         SECTION 19.2.1"
1390 PRINT
1410 PRINT
1420 PRINT
1430 PRINT
1440 PRINT "     AN OPERATION TEST FAILURE MEANS A FAILURE TO MAINTAIN"
1450 PRINT "AT LEAST 6 DIGITS OF PRECISION."
1460 PRINT
1470 PRINT
1480 PRINT "                                BEGIN TEST."
1490 PRINT
1500 PRINT
1520 PRINT
1530 PRINT "        .                      ++++++++++++"
1540 PRINT "                               + ADDITION +"
1550 PRINT "                               ++++++++++++"
1560 PRINT
1570 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1580 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1590 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1600 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1610 PRINT "SIX PLACE ACCURACY."
1620 PRINT
1630 PRINT
1640 PRINT "1ST. ADDEND"
1650 PRINT "   PLUS    ","REQUIRED","SUM OF","ABSOLUTE"
1660 PRINT "2ND. ADDEND","  SUM   ","SYSTEM"," ERROR  "
1670 PRINT
1680 PRINT
1690 LET L2=2+144
1700 LET L3=(2+144)+(-146)
1710 IF ABS(L3)<=1E-3 THEN 1760
1720 PRINT "  2       "
1730 PRINT "  +      =","  146 ",L2,L3;"*"
1740 PRINT " 144      "
1750 GOTO 1790
1760 PRINT "  2       "
1770 PRINT "  +      =","  146 ",L2,L3
1780 PRINT " 144      "
```

```
1790 PRINT
1800 PRINT
1810 LET O3=15.25+5.50
1820 LET O4=(15.25+5.50)+(-20.75)
1830 IF ABS(O4)<=1E-4 THEN 1880
1840 PRINT " 15.25     "
1850 PRINT "    +      =","  20.75 ",O3,O4;"*"
1860 PRINT "  5.50     "
1870 GOTO 1910
1880 PRINT " 15.25     "
1890 PRINT "    +      =","  20.75 ",O3,O4
1900 PRINT "  5.50     "
1910 PRINT
1920 PRINT
1930 LET P1=2.55E20+3.6E21
1940 LET P2=(2.55E20+3.6E21)+(-3.855E21)
1950 IF ABS(P2)<=1E16 THEN 2000
1960 PRINT " 2.55E20   "
1970 PRINT "     +     =","  3.85500E21 ",P1,P2;"*"
1980 PRINT "  3.6E21   "
1990 GOTO 2030
2000 PRINT " 2.55E20   "
2010 PRINT "     +     =","  3.85500E21 ",P1,P2
2020 PRINT "  3.6E21   "
2030 PRINT
2040 PRINT
2050 LET W1=44E-20+55E-19
2060 LET W2=(44E-20+55E-19)+(-59.4E-19)
2070 IF ABS(W2)<=1E-23 THEN 2120
2080 PRINT " 44E-20    "
2090 PRINT "    +      =","  5.94000E-18 ",W1,W2;"*"
2100 PRINT " 55E-19    "
2110 GOTO 2160
2120 PRINT " 44E-20    "
2130 PRINT "    +      =","  5.94000E-18 ",W1,W2
2140 PRINT " 55E-19    "
2150 PRINT
2160 PRINT "                                    END TEST."
2170 PRINT
2180 PRINT
2190 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 19

SECTION 19.1: THE ABS FUNCTION.

SECTION 19.1.1

(USING NUMERICAL CONSTANTS FOR ARGUMENTS.)

BEGIN TEST.

EACH EVALUATED FAILURE OF THE ABS FUNCTION WILL BE DENOTED BY AN ASTERISK BEING PRINTED ON THAT COMPARATIVE ROW OF OUTPUT. TEST PASSED IF THERE ARE NOT ANY ASTERISKS.

ARGUMENT	TRUE EVALUATION	SYSTEM EVALUATION
-10	10	10
15	15	15
-.5	.5	.5
.125	.125	.125
-62.5E-3	.0625	.0625
312.5E-4	.03125	.03125

END TEST.

SECTION 19.1.2

(USING ASSIGNED VARIABLES FOR ARGUMENTS.)

BEGIN TEST.

EACH EVALUATED FAILURE OF THE ABS FUNCTION WILL BE DENOTED BY AN ASTERISK BEING PRINTED ON THAT COMPARATIVE ROW OF OUTPUT. CHECK TEST PASSED IF THERE ARE NOT ANY ASTERISKS.

VALUE OF ARGUMENT	TRUE EVALUATION	SYSTEM EVALUATION
-10	10	10
15	15	15
-.5	.5	.5
.125	.125	.125
-62.5E-3	.0625	.0625
312.5E-4	.03125	.03125

END TEST.

SECTION 19.2.1

AN OPERATION TEST FAILURE MEANS A FAILURE TO MAINTAIN AT LEAST 6 DIGITS OF PRECISION.

BEGIN TEST.

++++++++++++
+ ADDITION +
++++++++++++

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

1ST. ADDEND PLUS 2ND. ADDEND		REQUIRED SUM	SUM OF SYSTEM	ABSOLUTE ERROR
2 + 144	=	146	146	0
15.25 + 5.50	=	20.75	20.75	0
2.55E20 + 3.6E21	=	3.85500E21	3.85500E+21	0
44E-20 + 55E-19	=	5.94000E-18	5.94000E-18	0

END TEST.

20.0 ELEMENTARY NUMERICAL EXPRESSIONS USING CONSTANTS (CONTINUED)

20.1 Subtraction

The objective of this test is to verify that the operation of subtraction of numerical constants will maintain at least six decimal digits of precision. The test forms the difference each of two NR1 constants, two NR2 constants, and two NR3 constants. There is again a four column output of which the first column is labeled "MINUEND MINUS SUBTRAHEND", the second column is labeled "REQUIRED DIFFERENCE", the third column is labeled "DIFFERENCE OF SYSTEM", and the fourth column is labeled "ABSOLUTE ERROR". The first column contains the pairs of numerical constants that are subtracted; the second column lists the standard conforming expected output; the third column lists the differences of the constants as evaluated by the system being tested; and the fourth column contains the absolute errors generated in the evaluation. If the implementation did not maintain at least six decimal digits of precision, an asterisk will appear beside any such case to the right of the fourth column.

20.2 Multiplication

The objective of this test is to verify that the operation of multiplication of two numerical constants will maintain six significant digits of precision. The test multiplies each of two NR1 constants, two NR2 constants, and two NR3 constants. The test outputs four columns of which the first column is labeled "1st FACTOR TIMES 2nd. FACTOR", the second column is labeled "REQUIRED PRODUCT", the third column is labeled "PRODUCT OF SYSTEM", and the fourth column is labeled "ABSOLUTE ERROR". The first column lists the pairs of numerical constants that are multiplied; the second column lists the standard conforming output; the third column lists the products of the constants as evaluated by the system being tested; and the fourth column contains the absolute errors. Again an asterisk will appear to the right of the fourth column for the products not satisfying the six significant digit criteria.

```
********************
* PROGRAM FILE 20 *
********************
```

```
0010 PRINT "PROGRAM FILE 20"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "                                BEGIN TEST."
0100 PRINT
0110 PRINT
```

```
0120 PRINT "                          SECTION 20.1"
0130 PRINT
0140 PRINT "                                ---------------"
0150 PRINT "                                - SUBTRACTION -"
0160 PRINT "                                ---------------"
0170 PRINT
0180 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0190 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0200 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0210 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0220 PRINT "SIX PLACE ACCURACY."
0230 PRINT
0240 PRINT
0250 PRINT " MINUEND  ","          ","DIFFERENCE"
0260 PRINT "  MINUS   "," REQUIRED ","    OF    ","ABSOLUTE"
0270 PRINT "SUBTRAHEND","DIFFERENCE","  SYSTEM  "," ERROR  "
0280 PRINT
0290 PRINT
0300 LET P4=956-482
0310 LET P5=(956-482)-474
0320 IF ABS(P5)<=1E-3 THEN 370
0330 PRINT " 956       "
0340 PRINT "  -       =","  474 ",P4,P5;"*"
0350 PRINT " 482       "
0360 GOTO 400
0370 PRINT " 956       "
0380 PRINT "  -       =","  474 ",P4,P5
0390 PRINT " 482       "
0400 PRINT
0410 PRINT
0420 LET P6=356.41-574.32
0430 LET P7=(356.41-574.32)-(-217.91)
0440 IF ABS(P7)<=1E-3 THEN 490
0450 PRINT " 356.41    "
0460 PRINT "   -      =","-217.91 ",P6,P7;"*"
0470 PRINT " 574.32    "
0480 GOTO 520
0490 PRINT " 356.41    "
0500 PRINT "   -      =","-217.91 ",P6,P7
0510 PRINT " 574.32    "
0520 PRINT
0530 PRINT
0540 LET P8=15.7E-24-18.5E-23
0550 LET P9=(15.7E-24-18.5E-23)-(-16.93E-23)
0560 IF ABS(P9)<=1E-27 THEN 610
0570 PRINT " 15.7E-24  "
0580 PRINT "    -     =","-1.69300E-22 ",P8,P9;"*"
0590 PRINT " 18.5E-23  "
0600 GOTO 640
0610 PRINT " 15.7E-24  "
0620 PRINT "    -     =","-1.69300E-22 ",P8,P9
0630 PRINT " 18.5E-23  "
0640 PRINT
0650 PRINT
0660 LET Q1=17E20-14E19
0670 LET Q2=(17E20-14E19)-15.6E20
0680 IF ABS(Q2)<=1E16 THEN 730
```

```
0690 PRINT " 17E20      "
0700 PRINT "   -       =","  1.56000E21 ",Q1,Q2;"*"
0710 PRINT " 14E19      "
0720 GOTO 760
0730 PRINT " 17E20      "
0740 PRINT "   -       =","  1.56000E21 ",Q1,Q2
0750 PRINT " 14E19      "
0760 PRINT
0770 PRINT "                                  END TEST."
0780 PRINT
0790 PRINT
0800 PRINT
0810 PRINT "                                BEGIN TEST."
0820 PRINT
0830 PRINT
0840 PRINT "                         SECTION 20.2"
0850 PRINT
0860 PRINT "                             XXXXXXXXXXXXXXXXXX"
0870 PRINT "                             X MULTIPLICATION X"
0880 PRINT "                             XXXXXXXXXXXXXXXXXX"
0890 PRINT
0900 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0910 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0920 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0930 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0940 PRINT "SIX PLACE ACCURACY."
0950 PRINT
0960 PRINT
0970 PRINT "1ST. FACTOR"
0980 PRINT "   TIMES   ","REQUIRED","PRODUCT OF","ABSOLUTE"
0990 PRINT "2ND. FACTOR","PRODUCT ","  SYSTEM  "," ERROR  "
1000 PRINT
1010 PRINT
1020 LET Q3=95*15
1030 LET Z2=(95*15)-1425
1040 IF ABS(Z2)<=1E-2 THEN 1090
1050 PRINT " 95       "
1060 PRINT " *       =","  1425 ",Q3,Z2;"*"
1070 PRINT " 15       "
1080 GOTO 1120
1090 PRINT " 95       "
1100 PRINT " *       =","  1425 ",Q3,Z2
1110 PRINT " 15       "
1120 PRINT
1130 PRINT
1140 LET Q4=96.2*10.3
1150 LET O5=(96.2*10.3)-990.86
1160 IF ABS(O5)<=1E-3 THEN 1210
1170 PRINT " 96.2     "
1180 PRINT "  *      =","  990.86 ",Q4,O5;"*"
1190 PRINT " 10.3     "
1200 GOTO 1240
1210 PRINT " 96.2     "
1220 PRINT "  *      =","  990.86 ",Q4,O5
1230 PRINT " 10.3     "
1240 PRINT
1250 PRINT
```

```
1260 LET Q5=3.5E18*1.2E15
1270 LET O7=(3.5E18*1.2E15)-4.2E33
1280 IF ABS(O7)<=1E28 THEN 1330
1290 PRINT " 3.5E18  "
1300 PRINT "   *     =","  4.20000E33 ",Q5,O7;"*"
1310 PRINT " 1.2E15  "
1320 GOTO 1360
1330 PRINT " 3.5E18  "
1340 PRINT "   *     =","  4.20000E33 ",Q5,O7
1350 PRINT " 1.2E15  "
1360 PRINT
1370 PRINT
1380 LET Q6=4E20*2E15
1390 LET O8=(4E20*2E15)-8E35
1400 IF ABS(O8)<=1E30 THEN 1450
1410 PRINT " 4E20    "
1420 PRINT "  *      =","  8.00000E35 ",Q6,O8;"*"
1430 PRINT " 2E15    "
1440 GOTO 1480
1450 PRINT " 4E20    "
1460 PRINT "  *      =","  8.00000E35 ",Q6,O8
1470 PRINT " 2E15    "
1480 PRINT
1490 PRINT "                                  END TEST."
1500 PRINT
1510 PRINT
1520 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

```
PROGRAM FILE 20

                        BEGIN TEST.

                    SECTION 20.1

                    ---------------
                    - SUBTRACTION -
                    ---------------

     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR
COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS
A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-
CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF
```

SIX PLACE ACCURACY.

MINUEND MINUS SUBTRAHEND		REQUIRED DIFFERENCE	DIFFERENCE OF SYSTEM	ABSOLUTE ERROR
956 - 482	=	474	474	0
356.41 - 574.32	=	-217.91	-217.91	0
15.7E-24 - 18.5E-23	=	-1.69300E-22	-1.69300E-22	0
17E20 - 14E19	=	1.56000E21	1.56000E+21	0

END TEST.

BEGIN TEST.

SECTION 20.2

XXXXXXXXXXXXXXXXX
X MULTIPLICATION X
XXXXXXXXXXXXXXXXX

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

1ST. FACTOR TIMES 2ND. FACTOR		REQUIRED PRODUCT	PRODUCT OF SYSTEM	ABSOLUTE ERROR
95 * 15	=	1425	1425	0

```
96.2
 *      =       990.86          990.86          0
10.3

3.5E18
  *     =       4.20000E33      4.20000E+33     0
1.2E15

4E20
 *      =       8.00000E35      8.00000E+35     0
2E15

                                    END TEST.
```

21.1 Division

The objective of this test is to verify that the operation of division between two numerical constants will yield at least six decimal digits of precision. This test computes the quotient of two NR1 constants, two NR2 constants, and two NR3 constants. There is a four column output of which the first column is labeled "DIVIDEND DIVIDED BY DIVISOR", the second column is labeled "REQUIRED QUOTIENT", the third column is labeled "QUOTIENT OF SYSTEM", and the fourth column is labeled "ABSOLUTE ERROR".

The first column lists the pairs of numerical constants which enter into the quotient, the second column lists the standard conforming output, the third column lists the quotients as evaluated by the system being tested, and the fourth column contains the absolute errors between the generated quotients and the true quotient. An asterisk to the right would indicate failure to maintain six significant digits.

21.2 Involution

The objective of this test is to verify that the operation of involution of numerical constants will yield at least six decimal digits of precision. This test forms the involution of two NR1 constants, two NR2 constants, and two NR3 constants . Its output falls in four columns again of which the first column is labeled "BASE ^ EXPONENT", the second column is labeled "REQUIRED POWER", the third column is labeled "POWER OF SYSTEM", and the fourth column is labeled "ABSOLUTE ERROR".

The first column lists the pairs of numerical constants which should have been involuted, the second column lists the standard conforming output, the third column lists the involution of the constants as generated by the system being tested, and the fourth column contains the absolute errors. An asterisk appears to the right of this column as before for failure to maintain six significant digits.

```
**********************
* PROGRAM FILE 21 *
**********************
```

```
0020 PRINT "PROGRAM FILE 21"
0070 PRINT
0080 PRINT
```

```
0090 PRINT
0100 PRINT "                                   BEGIN TEST."
0110 PRINT
0120 PRINT
0130 PRINT "                              SECTION 21.1"
0140 PRINT
0150 PRINT "                                   ////////////"
0160 PRINT "                                   / DIVISION /"
0170 PRINT "                                   ////////////"
0180 PRINT
0190 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0200 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0210 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0220 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OF OF"
0230 PRINT "SIX PLACE ACCURACY."
0240 PRINT
0250 PRINT
0260 PRINT " DIVIDEND "
0270 PRINT "DIVIDED BY","REQUIRED","QUOTIENT OF","ABSOLUTE"
0280 PRINT " DIVISOR  ","QUOTIENT","   SYSTEM  "," ERROR  "
0290 PRINT
0300 PRINT
0310 LET R1=105/5
0320 LET Z3=(105/5)-21
0330 IF ABS(Z3)<=1E-4 THEN 380
0340 PRINT " 105      "
0350 PRINT "  /      =","  21 ",R1,Z3;"*"
0360 PRINT "  5       "
0370 GOTO 410
0380 PRINT " 105      "
0390 PRINT "  /      =","  21 ",R1,Z3
0400 PRINT "  5       "
0410 PRINT
0420 PRINT
0430 LET R2=6.2/0.5
0440 LET O6=(6.2/0.5)-12.4
0450 IF ABS(O6)<=1E-4 THEN 500
0460 PRINT " 6.2      "
0470 PRINT "  /      =","  12.4 ",R2,O6;"*"
0480 PRINT " 0.5      "
0490 GOTO 530
0500 PRINT " 6.2      "
0510 PRINT "  /      =","  12.4 ",R2,O6
0520 PRINT " 0.5      "
0530 PRINT
0540 PRINT
0550 LET R3=1.0E29/5.0E10
0560 LET Q8=(1.0E29/5.0E10)-2.E18
0570 IF ABS(Q8)<=1E13 THEN 620
0580 PRINT " 1.0E29   "
0590 PRINT "    /     =","  2.00000E18 ",R3,Q8;"*"
0600 PRINT " 5.0E10   "
0610 GOTO 650
0620 PRINT " 1.0E29   "
0630 PRINT "    /     =","  2.00000E18 ",R3,Q8
0640 PRINT " 5.0E10   "
0650 PRINT
```

```
0660 PRINT
0670 LET R4=15E30/5E15
0680 LET O9=(15E30/5E15)-3E15
0690 IF ABS(O9)<=1E10 THEN 740
0700 PRINT " 15E30    "
0710 PRINT "    /     =","  3.00000E15 ",R4,O9;"*"
0720 PRINT "   5E15    "
0730 GOTO 770
0740 PRINT " 15E30    "
0750 PRINT "    /     =","  3.00000E15 ",R4,O9
0760 PRINT "   5E15    "
0770 PRINT
0780 PRINT
0790 PRINT "                                  END TEST."
0800 PRINT
0810 PRINT
0820 PRINT
0830 PRINT "                          SECTION 21.2"
0840 PRINT
0850 PRINT "                          ^^^^^^^^^^^^"
0853 PRINT "                          ^INVOLUTION^"
0855 PRINT "                          ^^^^^^^^^^^^"
0860 PRINT
0870 PRINT
0880 PRINT "                                  BEGIN TEST."
0890 PRINT
0900 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0910 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0920 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0930 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0940 PRINT "SIX PLACE ACCURACY."
0950 PRINT
0960 PRINT
0970 PRINT "  BASE   "
0980 PRINT "    ^    ","REQUIRED","POWER OF","ABSOLUTE"
0990 PRINT "EXPONENT","  POWER   "," SYSTEM "," ERROR   "
1000 PRINT
1010 PRINT
1020 LET R5=30^2
1030 LET S5=(30^2)-900
1040 IF ABS(S5)<=1E-3 THEN 1090
1050 PRINT " 30       "
1060 PRINT " ^       =","  900 ",R5,S5;"*"
1070 PRINT " 2        "
1080 GOTO 1120
1090 PRINT " 30       "
1100 PRINT " ^       =","  900 ",R5,S5
1110 PRINT " 2        "
1120 PRINT
1130 PRINT
1140 LET T5=0^0
1150 LET U5=(0^0)-1
1160 IF ABS(U5)<=1E-5 THEN 1210
1170 PRINT " 0        "
1180 PRINT " ^       =","  1 ",T5,U5;"*"
1190 PRINT " 0        "
1200 GOTO 1240
```

```
1210 PRINT " 0        "
1220 PRINT " ^       =","  1 ",T5,U5
1230 PRINT " 0        "
1240 PRINT
1250 PRINT
1260 LET W5=5.0^(-2.0)
1270 LET X5=(5.0^(-2.0))-.04
1280 IF ABS(X5)<=1E-6 THEN 1330
1290 PRINT " 5.0      "
1300 PRINT "  ^       =","  .04 ",W5,X5;"*"
1310 PRINT "-2.0      "
1320 GOTO 1360
1330 PRINT " 5.0      "
1340 PRINT "  ^       =","  .04 ",W5,X5
1350 PRINT "-2.0      "
1360 PRINT
1370 PRINT
1380 LET X2=90.E-2^.2E01
1390 LET X3=(90.E-2^.2E01)-81.E-2
1400 IF ABS(X3)<=1E-6 THEN 1450
1410 PRINT " 90.E-2   "
1420 PRINT "   ^      =","  .81 ",X2,X3;"*"
1430 PRINT "  .2E01   "
1440 GOTO 1480
1450 PRINT " 90.E-2   "
1460 PRINT "   ^      =","  .81 ",X2,X3
1470 PRINT "  .2E01   "
1480 PRINT
1490 PRINT
1500 LET X4=4E4^50E-2
1510 LET Y4=(4E4^50E-2)-2E2
1520 IF ABS(Y4)<=1E-3 THEN 1570
1530 PRINT "  4E4     "
1540 PRINT "   ^      =","  200 ",X4,Y4;"*"
1550 PRINT " 50E-2    "
1560 GOTO 1600
1570 PRINT "  4E4     "
1580 PRINT "   ^      =","  200 ",X4,Y4
1590 PRINT " 50E-2    "
1600 PRINT
1610 PRINT
1620 PRINT "                                    END TEST."
1630 PRINT
1640 PRINT
1650 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 21

BEGIN TEST.

SECTION 21.1

////////////
/ DIVISION /
////////////

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OF OF SIX PLACE ACCURACY.

DIVIDEND DIVIDED BY DIVISOR		REQUIRED QUOTIENT	QUOTIENT OF SYSTEM	ABSOLUTE ERROR
105 / 5	=	21	21	0
6.2 / 0.5	=	12.4	12.4	0
1.0E29 / 5.0E10	=	2.00000E18	2.00000E+18	0
15E30 / 5E15	=	3.00000E15	3.00000E+15	0

END TEST.

SECTION 21.2

^^^^^^^^^^^^
^INVOLUTION^
^^^^^^^^^^^^

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

BASE ^ EXPONENT		REQUIRED POWER	POWER OF SYSTEM	ABSOLUTE ERROR
30 ^ 2	=	900	900	0
0 ^ 0	=	1	1	0
5.0 ^ -2.0	=	.04	0.04	0
90.E-2 ^ .2E01	=	.81	0.81	0
4E4 ^ 50E-2	=	200	200.	0

END TEST.

22.0 ELEMENTARY EXPRESSIONS USING SIMPLE VARIABLES

The objective here is to extend the construction process of numerical expressions by using simple numeric variables with the operations of addition, subtraction, multiplication, division, and involution. Each test is constructed to use only simple variables assigned the same type of numerical constants with each other, that is, simple variables that have been assigned NR1 constants are not used with simple variables that have been assigned NR2 or NR3 constants, etc.

22.1 Addition

The objective of this test is to verify that the operation of addition of two numerically assigned simple variables will yield at least six decimal digits of precision. This test is similar to test 19.3, except for the simple variables and the values of their assigned numerical constants. The labeling of the first column is different in that for this test the first column is labeled "Assignment 1 plus Assignment 2". This test generates the same output format as in test 19.3.

22.2 Subtraction

The objective of this test is to verify that the operation of subtraction of two numerically assigned simple variables will yield at least six decimal digits of precision. This test is similar to test test 20.1, except for the simple variables and their assigned numerical constants. The labeling of the first column is different, in that, for this test the first column is labeled "Assignment 1 minus Assignment 2". The output format is the same as that in 20.1.

```
********************
* PROGRAM FILE 22 *
********************
```

```
0010 PRINT "PROGRAM FILE 22"
0060 PRINT
0070 PRINT
0080 PRINT
0140 PRINT "                                BEGIN TEST."
0150 PRINT
0160 PRINT
0170 PRINT "                          SECTION 22.1"
0180 PRINT
0190 PRINT "                                ++++++++++++"
0200 PRINT "                                + ADDITION +"
```

```
0210 PRINT "                                    ++++++++++++"
0220 PRINT
0230 PRINT "      IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0240 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0250 PRINT " A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0260 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0270 PRINT "SIX PLACE ACCURACY."
0280 PRINT
0290 PRINT
0300 PRINT "ASSIGNMENT 1"
0310 PRINT "    PLUS    ","REQUIRED","SUM OF","ABSOLUTE"
0320 PRINT "ASSIGNMENT 2","  SUM    ","SYSTEM"," ERROR  "
0330 PRINT
0340 PRINT
0350 LET K2=2
0360 LET K3=-12
0370 LET K4=K2+K3
0380 LET Z1=10
0390 LET X1=K4+Z1
0400 IF ABS(X1)<=1E-4 THEN 450
0410 PRINT " 2       "
0420 PRINT " +      =","-10 ",K4,X1;"*"
0430 PRINT "-12      "
0440 GOTO 480
0450 PRINT " 2       "
0460 PRINT " +      =","-10 ",K4,X1
0470 PRINT "-12      "
0480 PRINT
0490 PRINT
0500 LET M4=10.5
0510 LET M5=-32.5
0520 LET M8=M4+M5
0530 LET Z2=22.0
0540 LET X2=M8+Z2
0550 IF ABS(X2)<=1E-4 THEN 600
0560 PRINT " 10.5    "
0570 PRINT "  +     =","-22.0 ",M8,X2;"*"
0580 PRINT "-32.5    "
0590 GOTO 630
0600 PRINT " 10.5    "
0610 PRINT "  +     =","-22.0 ",M8,X2
0620 PRINT "-32.5    "
0630 PRINT
0640 PRINT
0650 LET N4=2.5E20
0660 LET N5=3.5E21
0670 LET N6=N4+N5
0680 LET Z3=-3.75E21
0690 LET X3=N6+Z3
0700 IF ABS(X3)<=1E16 THEN 750
0710 PRINT " 2.5E20  "
0720 PRINT "   +    ="," 3.75000E21 ",N6,X3;"*"
0730 PRINT " 3.5E21  "
0740 GOTO 780
0750 PRINT " 2.5E20  "
0760 PRINT "   +    ="," 3.75000E21 ",N6,X3
0770 PRINT " 3.5E21  "
```

```
0780 PRINT
0790 PRINT
0800 LET K5=3E20
0810 LET K6=4E20
0820 LET K7=K5+K6
0830 LET Z4=-7E20
0840 LET X4=K7+Z4
0850 IF ABS(X4)<=1E15 THEN 900
0860 PRINT " 3E20    "
0870 PRINT "  +     =","  7.00000E20 ",K7,X4;"*"
0880 PRINT " 4E20    "
0890 GOTO 930
0900 PRINT " 3E20    "
0910 PRINT "  +     =","  7.00000E20 ",K7,X4
0920 PRINT " 4E20    "
0930 PRINT
0940 PRINT "                                   END TEST."
0950 PRINT
0960 PRINT
0970 PRINT
0980 PRINT "                                 BEGIN TEST."
0990 PRINT
1000 PRINT
1010 PRINT "                          SECTION 22.2"
1020 PRINT
1030 PRINT "                                ---------------"
1040 PRINT "                                - SUBTRACTION -"
1050 PRINT "                                ---------------"
1060 PRINT
1070 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1080 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1090 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1100 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1110 PRINT "SIX PLACE ACCURACY."
1120 PRINT
1130 PRINT
1140 PRINT "ASSIGNMENT 1","          ","DIFFERENCE"
1150 PRINT "    MINUS   "," REQUIRED ","    OF    ","ABSOLUTE"
1160 PRINT "ASSIGNMENT 2","DIFFERENCE","  SYSTEM  "," ERROR  "
1170 PRINT
1180 PRINT
1190 LET K8=156
1200 LET K9=6
1210 LET L5=K8-K9
1220 LET Z5=150
1230 LET Y1=L5-Z5
1240 IF ABS(Y1)<=1E-3 THEN 1290
1250 PRINT " 156      "
1260 PRINT "  -      ="," 156 ",L5,Y1;"*"
1270 PRINT "  6       "
1280 GOTO 1320
1290 PRINT " 156      "
1300 PRINT "  -      ="," 150 ",L5,Y1
1310 PRINT "  6       "
1320 PRINT
1330 PRINT
1340 LET L7=2.55
```

```
1350 LET L8=-12.55
1360 LET L9=L8-L7
1370 LET Z6=-15.1
1380 LET Y2=L9-Z6
1390 IF ABS(Y2)<=1E-4 THEN 1440
1400 PRINT "-12.55     "
1410 PRINT "   -      =","-15.1 ",L9,Y2;"*"
1420 PRINT "  2.55     "
1430 GOTO 1470
1440 PRINT "-12.55     "
1450 PRINT "   -      =","-15.1 ",L9,Y2
1460 PRINT "  2.55     "
1470 PRINT
1480 PRINT
1490 LET A4=2.55E20
1500 LET A5=-2.55E20
1510 LET M6=A4-A5
1520 LET Z7=5.1E20
1530 LET Y3=M6-Z7
1540 IF ABS(Y3)<=1E15 THEN 1590
1550 PRINT " 2.55E20   "
1560 PRINT "    -     ="," 5.10000E20 ",M6,Y3;"*"
1570 PRINT "-2.55E20   "
1580 GOTO 1620
1590 PRINT " 2.55E20   "
1600 PRINT "    -     ="," 5.10000E20 ",M6,Y3
1610 PRINT "-2.55E20   "
1620 PRINT
1630 PRINT
1640 LET M7=1E30
1650 LET M3=19E30
1660 LET M9=M3-M7
1670 LET Z8=18E30
1680 LET Y=M9-Z8
1690 IF ABS(Y4)<=1E26 THEN 1740
1700 PRINT " 19E30     "
1710 PRINT "   -      ="," 1.80000E30 ",M9,Y4;"*"
1720 PRINT "  1E30     "
1730 GOTO 1770
1740 PRINT " 19E30     "
1750 PRINT "   -      ="," 1.80000E30 ",M9,Y4
1760 PRINT "  1E30     "
1770 PRINT
1780 PRINT "                                  END TEST."
1790 PRINT
1800 PRINT
1810 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 22

BEGIN TEST.

SECTION 22.1

++++++++++++
+ ADDITION +
++++++++++++

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 PLUS ASSIGNMENT 2		REQUIRED SUM	SUM OF SYSTEM	ABSOLUTE ERROR
2 + -12	=	-10	-10	0
10.5 + -32.5	=	-22.0	-22	0
2.5E20 + 3.5E21	=	3.75000E21	3.75000E+21	0
3E20 + 4E20	=	7.00000E20	7.00000E+20	0

END TEST.

BEGIN TEST.

SECTION 22.2

- SUBTRACTION -

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 MINUS ASSIGNMENT 2		REQUIRED DIFFERENCE	DIFFERENCE OF SYSTEM	ABSOLUTE ERROR
156 / 6	=	150	150	0
-12.55 / 2.55	=	-15.1	-15.1	0
2.55E20 / -2.55E20	=	5.10000E20	5.10000E+20	0
19E30 / 1E30	=	1.80000E30	1.80000E+31	0

END TEST.

23.0 ELEMENTARY EXPRESSIONS USING SIMPLE VARIABLES (CONTINUED)

23.1 Multiplication

The objective of this test is to verify that the operation of multiplication between two numeric assigned simple variables will yield an accuracy of at least six decimal digits of precision. The test is similar to test 20.2, except for the simple variables and the values of their assigned numerical constants. The labeling of the first column is different in that for this test the first column is labeled "Assignment 1 times Assignment 2". Finally, this test has the same output format as test 20.2.

23.2 Division

The objective of this test is to verify that the operation of division between two numerically assigned simple variables will yield an accuracy of at least six decimal digits of precision. The test is similar to test 21.1, except for the simple variables and the values of their assigned numerical constants. The labeling of the first column is different in that for this test the first column is labeled "Assignment 1 divided by Assignment 2." Again, this test has the same output format as test 21.1.

23.3 Involution

The objective of this test is verify that the operation of involution between two numerically assigned simple variables will yield an accuracy of at least six decimal digits of precision. The test is similar to test 21.2, except for the simple variables and the values of their assigned numerical constants. The labeling of the first column is different in that for this test the first column is labeled "Assignment 1 ^ Assignment 2." Again, this test has the same output format as test 21.2.

```
********************
* PROGRAM FILE 23 *
********************
```

```
0010 PRINT "PROGRAM FILE 23"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "                                BEGIN TEST."
0100 PRINT
0110 PRINT
0120 PRINT "                          SECTION 23.1"
0130 PRINT
```

```
0140 PRINT "                              XXXXXXXXXXXXXXXXXX"
0150 PRINT "                              X MULTIPLICATION X"
0160 PRINT "                              XXXXXXXXXXXXXXXXXX"
0170 PRINT
0180 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0190 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0200 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0210 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0220 PRINT "SIX PLACE ACCURACY."
0230 PRINT
0240 PRINT
0250 PRINT "ASSIGNMENT 1"
0260 PRINT "    TIMES   ","REQUIRED","PRODUCT OF","ABSOLUTE"
0270 PRINT "ASSIGNMENT 2","PRODUCT ","  SYSTEM  "," ERROR  "
0280 PRINT
0290 PRINT
0300 LET V1=15
0310 LET V2=20
0320 LET U1=V1*V2
0330 LET A1=300
0340 LET X5=U1-A1
0350 IF ABS(X5)<=1E-3 THEN 400
0360 PRINT " 15       "
0370 PRINT " *       =","  300 ",U1,X5;"*"
0380 PRINT " 20       "
0390 GOTO 430
0400 PRINT " 15       "
0410 PRINT " *       =","  300 ",U1,X5
0420 PRINT " 20       "
0430 PRINT
0440 PRINT
0450 LET V3=3.6
0460 LET V4=4.2
0470 LET U2=V3*V4
0480 LET A2=15.12
0490 LET X6=U2-A2
0500 IF ABS(X6)<=1E-4 THEN 550
0510 PRINT " 3.6      "
0520 PRINT "  *      =","  15.12 ",U2,X6;"*"
0530 PRINT " 4.2      "
0540 GOTO 580
0550 PRINT " 3.6      "
0560 PRINT "  *      =","  15.12 ",U2,X6
0570 PRINT " 4.2      "
0580 PRINT
0590 PRINT
0600 LET V5=3.6E15
0610 LET V6=1.2E3
0620 LET U3=V5*V6
0630 LET A3=4.32E18
0640 LET X7=U3-A3
0650 IF ABS(X7)<=1E13 THEN 700
0660 PRINT " 3.6E15   "
0670 PRINT "   *     =","  4.32000E18 ",U3,X7;"*"
0680 PRINT " 1.2E3    "
0690 GOTO 730
0700 PRINT " 3.6E15   "
```

```
0710 PRINT "    *     =","  4.32000E18 ",U3,X7
0720 PRINT " 1.2E3    "
0730 PRINT
0740 PRINT
0750 LET V7=3E18
0760 LET V8=2E-3
0770 LET V9=V7*V8
0780 LET A4=6E15
0790 LET X8=V9-A4
0800 IF ABS(X8)<=1E10 THEN 850
0810 PRINT " 3E18     "
0820 PRINT "  *      =","  6.00000E15 ",V9,X8;"*"
0830 PRINT " 2E-3     "
0840 GOTO 880
0850 PRINT " 3E18     "
0860 PRINT "  *      =","  6.00000E15 ",V9,X8
0870 PRINT " 2E-3     "
0880 PRINT
0890 PRINT "                                  END TEST."
0900 PRINT
0910 PRINT
0920 PRINT
0930 PRINT "                                BEGIN TEST."
0940 PRINT
0950 PRINT
0960 PRINT "                         SECTION 23.2"
0970 PRINT
0980 PRINT "                               ////////////"
0990 PRINT "                               / DIVISION /"
1000 PRINT "                               ////////////"
1010 PRINT
1020 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1030 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1040 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1050 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1060 PRINT "SIX PLACE ACCURACY."
1070 PRINT
1080 PRINT
1090 PRINT "ASSIGNMENT 1"
1100 PRINT " DIVIDED BY ","REQUIRED","QUOTIENT OF","ABSOLUTE"
1110 PRINT "ASSIGNMENT 2","QUOTIENT","  SYSTEM   "," ERROR  "
1120 PRINT
1130 PRINT
1140 LET S1=15
1150 LET S2=5
1160 LET U4=S1/S2
1170 LET A5=3
1180 LET Y5=U4-A5
1190 IF ABS(Y5)<=1E-5 THEN 1240
1200 PRINT " 15       "
1210 PRINT " /       =","  3 ",U4,Y5;"*"
1220 PRINT " 5        "
1230 GOTO 1270
1240 PRINT " 15       "
1250 PRINT " /       =","  3 ",U4,Y5
1260 PRINT " 5        "
1270 PRINT
```

```
1280 PRINT
1290 LET S3=14.2
1300 LET S4=7.1
1310 LET U5=S3/S4
1320 LET A6=2.0
1330 LET Y6=U5-A6
1340 IF ABS(Y6)<=1E-5 THEN 1390
1350 PRINT " 14.2     "
1360 PRINT "  /      =","  2.0 ",U5,Y6;"*"
1370 PRINT "  7.1     "
1380 GOTO 1420
1390 PRINT " 14.2     "
1400 PRINT "  /      =","  2.0 ",U5,Y6
1410 PRINT "  7.1     "
1420 PRINT
1430 PRINT
1440 LET Q7=3.5E30
1450 LET Q9=7.0E10
1460 LET U6=Q7/Q9
1470 LET A7=5.0E19
1480 LET Y7=U6-5.0E19
1490 IF ABS(Y7)<=1E14 THEN 1540
1500 PRINT " 3.5E30   "
1510 PRINT "    /     =","  5.00000E19 ",U6,Y7;"*"
1520 PRINT " 7.0E10   "
1530 GOTO 1570
1540 PRINT " 3.5E30   "
1550 PRINT "    /     =","  5.00000E19 ",U6,Y7
1560 PRINT " 7.0E10   "
1570 PRINT
1580 PRINT
1590 LET S5=18E20
1600 LET S6=9E-2
1610 LET U7=S5/S6
1620 LET A8=2E22
1630 LET Y8=U7-2E22
1640 IF ABS(Y8)<=1E17 THEN 1690
1650 PRINT " 18E20    "
1660 PRINT "    /     =","  2.00000E22 ",U7,Y8;"*"
1670 PRINT "  9E-2    "
1680 GOTO 1720
1690 PRINT " 18E20    "
1700 PRINT "    /     =","  2.00000E22 ",U7,Y8
1710 PRINT "  9E-2    "
1720 PRINT
1730 PRINT "                                    END TEST."
1740 PRINT
1750 PRINT
1760 PRINT
1770 PRINT "                         SECTION 23.3"
1780 PRINT
1790 PRINT "                         ^^^^^^^^^^^^"
1793 PRINT "                         ^INVOLUTION^"
1795 PRINT "                         ^^^^^^^^^^^^"
1800 PRINT
1810 PRINT
1820 PRINT "                                    BEGIN TEST."
```

```
1830 PRINT
1840 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1850 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1860 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1870 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1880 PRINT "SIX PLACE ACCURACY."
1890 PRINT
1900 PRINT
1910 PRINT "ASSIGNMENT 1"
1920 PRINT "     ^        ","REQUIRED","POWER OF","ABSOLUTE"
1930 PRINT "ASSIGNMENT 2"," POWER  "," SYSTEM "," ERROR  "
1940 PRINT
1950 PRINT
1960 LET T6=-5
1970 LET U6=4
1980 LET W6=T6^U6
1990 LET B1=625
2000 LET T7=W6-B1
2010 IF ABS(T7)<=1E-3 THEN 2060
2020 PRINT "-5          "
2030 PRINT " ^         =","  625 ",W6,T7;"*"
2040 PRINT " 4          "
2050 GOTO 2090
2060 PRINT "-5          "
2070 PRINT " ^         =","  625 ",W6,T7
2080 PRINT " 4          "
2090 PRINT
2100 PRINT
2110 LET R6=.625
2120 LET S6=0.0
2130 LET X6=R6^S6
2140 LET B2=1.0
2150 LET Y6=X6-B2
2160 IF ABS(Y6)<=1E-5 THEN 2210
2170 PRINT " .625       "
2180 PRINT " ^         =","  1 ",X6,Y6;"*"
2190 PRINT " 0.0        "
2200 GOTO 2240
2210 PRINT " .625       "
2220 PRINT " ^         =","  1 ",X6,Y6
2230 PRINT " 0.0        "
2240 PRINT
2250 PRINT
2260 LET T8=-.005E3
2270 LET U7=-200.E-2
2280 LET W7=T8^U7
2290 LET B3=4.0E-2
2300 LET X7=W7-B3
2310 IF ABS(X7)<=1E-6 THEN 2360
2320 PRINT "-.005E3     "
2330 PRINT "    ^      =","  .04 ",W7,X7;"*"
2340 PRINT "-200.E-2    "
2350 GOTO 2390
2360 PRINT "-.005E3     "
2370 PRINT "    ^      =","  .04 ",W7,X7
2380 PRINT "-200.E-2    "
2390 PRINT
```

```
2400 PRINT
2410 LET Y7=400E-2
2420 LET Y8=5E-1
2430 LET W8=Y7^Y8
2440 LET B4=200E-2
2450 LET W9=W8-B4
2460 IF ABS(W9)<=1E-5 THEN 2510
2470 PRINT " 400E-2    "
2480 PRINT "    ^      =","  2 ",W8,W9;"*"
2490 PRINT "  5E-1     "
2500 GOTO 2540
2510 PRINT " 400E-2    "
2520 PRINT "    ^      =","  2 ",W8,W9
2530 PRINT "  5E-1     "
2540 PRINT
2550 PRINT "                                    END TEST."
2560 PRINT
2570 PRINT
2580 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

```
PROGRAM FILE 23

                          BEGIN TEST.

                      SECTION 23.1

                          XXXXXXXXXXXXXXXXXX
                          X MULTIPLICATION X
                          XXXXXXXXXXXXXXXXXX

     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR
COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS
A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-
CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF
SIX PLACE ACCURACY.


ASSIGNMENT 1
    TIMES      REQUIRED       PRODUCT OF     ABSOLUTE
ASSIGNMENT 2   PRODUCT          SYSTEM        ERROR
```

15 * 20	=	300	300	0
3.6 * 4.2	=	15.12	15.12	0
3.6E15 * 1.2E3	=	4.32000E18	4.32000E+18	0
3E18 * 2E-3	=	6.00000E15	6.00000E+15	0

END TEST.

BEGIN TEST.

SECTION 23.2

////////////
/ DIVISION /
////////////

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 DIVIDED BY ASSIGNMENT 2		REQUIRED QUOTIENT	QUOTIENT OF SYSTEM	ABSOLUTE ERROR
15 / 5	=	3	3	0
14.2 / 7.1	=	2.0	2	0
3.5E30 / 7.0E10	=	5.00000E19	5.00000E+19	0

18E20 / 9E-2	=	2.00000E22	2.00000E+22	0

END TEST.

SECTION 23.3

^INVOLUTION^

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 ^ ASSIGNMENT 2		REQUIRED POWER	POWER OF SYSTEM	ABSOLUTE ERROR
-5 ^ 4	=	625	625	0
.625 ^ 0.0	=	1	1	0
-.005E3 ^ -200.E-2	=	.04	.04	0
400E-2 ^ 5E-1	=	2	2	0

END TEST.

24.0 ELEMENTARY OPERATIONS ON MIXED TYPE CONSTANTS

The objective of the next few tests is to determine whether the terms of a numerical expression can be of mixed type, that is, whether NR1 constants can be added to NR2 or NR3 constants, whether NR1 or NR2 constants can be subtracted from NR3 constants, etc. For this test section, each sub-test will at most use two terms, since expressions with more than two terms will be tested later. The reader should remember that the word type here only refers to graphical form and not to the formal meaning of a typed constant or variable as in FORTRAN, for example. The reader should be familiar with section 7 of BSR X3.60 at this point.

24.1 Addition

The objective of this test is to verify that the operation of addition on two numerical constants that are not of the same type will still yield accuracy of at least six decimal digits of precision. The test is similar to test 19.3, except for the mixing of types and the labeling of the first output column which for this test is labeled "1st Addend plus 2nd Addend". This test has the same output format as test 19.3.

24.2 Subtraction

The objective of this test is to verify that the operation of subtraction between two numerical constants, not of the same type, will maintain at least six decimal digits of precision. The test is similar to test 20.1, except for the mixing of types, and the word minus used in labeling of the first column output. Finally this test has the same output format as test 20.1.

```
********************
* PROGRAM FILE 24 *
********************
```

```
0010 PRINT "PROGRAM FILE 24"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "        SECTION 24.0: PRIMITIVE OPERATIONS ON MIXED MODES."
0100 PRINT
0110 PRINT
0120 PRINT
0180 PRINT "                              BEGIN TEST."
0190 PRINT
0200 PRINT
0210 PRINT "                         SECTION 24.1"
```

```
0220 PRINT
0230 PRINT "                                    ++++++++++++"
0240 PRINT "                                    + ADDITION +"
0250 PRINT "                                    ++++++++++++"
0260 PRINT
0270 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0280 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0290 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0300 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0310 PRINT "SIX PLACE ACCURACY."
0320 PRINT
0330 PRINT
0340 PRINT "1ST. ADDEND"
0350 PRINT "     +     ","REQUIRED","SUM OF","ABSOLUTE"
0360 PRINT "2ND. ADDEND","  SUM    ","SYSTEM"," ERROR  "
0370 PRINT
0380 LET A$=" "
0390 LET R8=12+2.5
0400 LET S8=(12+2.5)+(-14.5)
0410 IF ABS(S8)<=1E-4 THEN 430
0420 LET A$="*"
0430 PRINT " 12          "
0440 PRINT "  +         =","  14.5 ",R8,S8;A$
0450 PRINT " 2.5         "
0460 PRINT
0470 LET A$=" "
0480 LET S9=14+12.5E-3
0490 LET T8=(14+12.5E-3)+(-14.0125)
0500 IF ABS(T8)<=1E-4 THEN 520
0510 LET A$="*"
0520 PRINT "   14        "
0530 PRINT "    +       =","  14.0125 ",S9,T8;A$
0540 PRINT " 12.5E-3     "
0550 PRINT
0560 LET A$=" "
0570 LET R9=-9+(-15E-4)
0580 LET T9=(-9+(-15E-4))+9.0015
0590 IF ABS(T9)<=1E-5 THEN 610
0600 LET A$="*"
0610 PRINT " -9          "
0620 PRINT "  +         =","-9.0015 ",R9,T9;A$
0630 PRINT "-15E-4       "
0640 PRINT
0650 LET A$=" "
0660 LET U8=.625+(-.00005E7)
0670 LET U9=(.625+(-.00005E7))+499.375
0680 IF ABS(U9)<=1E-3 THEN 700
0690 LET A$="*"
0700 PRINT " .625        "
0710 PRINT "   +        =","-499.375 ",U8,U9;A$
0720 PRINT "-.00005E7    "
0730 PRINT
0740 LET A$=" "
0750 LET W8=1234.2+36000E-5
0760 LET X9=(1234.2+36000E-5)+(-1234.56)
0770 IF ABS(X9)<=1E-2 THEN 790
0780 LET A$="*"
```

```
0790 PRINT " 1234.2       "
0800 PRINT "    +        =","  1234.56 ",W8,X9;A$
0810 PRINT " 36000E-5     "
0820 PRINT
0830 LET A$=" "
0840 LET X8=65.4321E21+12345E17
0850 LET W9=(65.4321E21+12345E17)+(-6.66666E22)
0860 IF ABS(W9)<=1E17 THEN 880
0870 LET A$="*"
0880 PRINT " 65.4321E21   "
0890 PRINT "     +       =","  6.66666E22 ",X8,W9;A$
0900 PRINT " 12345E17     "
0910 PRINT
0920 PRINT "                                  END TEST."
0930 PRINT
0940 PRINT
0950 PRINT
0960 PRINT "                                BEGIN TEST."
0970 PRINT
0980 PRINT
0990 PRINT "                          SECTION 24.2"
1000 PRINT
1010 PRINT "                                ---------------"
1020 PRINT "                                - SUBTRACTION -"
1030 PRINT "                                ---------------"
1040 PRINT
1050 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1060 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1070 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1080 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1090 PRINT "SIX PLACE ACCURACY."
1100 PRINT
1110 PRINT
1120 PRINT " MINUEND  ","           ","DIFFERENCE"
1130 PRINT "    -     "," REQUIRED ","    OF    ","ABSOLUTE"
1140 PRINT "SUBTRAHEND","DIFFERENCE","  SYSTEM  "," ERROR  "
1150 PRINT
1160 LET A$=" "
1170 LET Y8=2-.76544
1180 LET Y9=(2-.76544)-1.23456
1190 IF ABS(Y9)<=1E-5 THEN 1210
1200 LET A$="*"
1210 PRINT "    2         "
1220 PRINT "    -        =","  1.23456 ",Y8,Y9;A$
1230 PRINT " .76544       "
1240 PRINT
1250 LET A$=" "
1260 LET Z8=15-(-.00520E3)
1270 LET Z9=(15-(-.00520E3))-20.20
1280 IF ABS(Z9)<=1E-4 THEN 1300
1290 LET A$="*"
1300 PRINT "   15         "
1310 PRINT "    -        =","  20.2 ",Z8,Z9;A$
1320 PRINT "-.00520E3     "
1330 PRINT
1340 LET A$=" "
1350 LET A=-9-87600E-5
```

```
1360 LET B0=(-9-87600E-5)-(-9.876)
1370 IF ABS(B0)<=1E-5 THEN 1390
1380 LET A$="*"
1390 PRINT "   -9          "
1400 PRINT "    -        =","-9.876 ",A,B0;A$
1410 PRINT " 87600E-5     "
1420 PRINT
1430 LET A$=" "
1440 LET C1=8.8-.000231E6
1450 LET D2=(8.8-.000231E6)-(-222.2)
1460 IF ABS(D2)<=1E-3 THEN 1480
1470 LET A$="*"
1480 PRINT "    9.8       "
1490 PRINT "     -        =","-222.2 ",C1,D2;A$
1500 PRINT " .000231E6    "
1510 PRINT
1520 LET A$=" "
1530 LET E3=177.177-540540E-4
1540 LET F4=(177.177-540540E-4)-123.123
1550 IF ABS(F4)<=1E-3 THEN 1570
1560 LET A$="*"
1570 PRINT "  177.177     "
1580 PRINT "     -        ="," 123.123 ",E3,F4;A$
1590 PRINT " 540540E-4    "
1600 PRINT
1610 LET A$=" "
1620 LET G5=-90.1233E20-(-12345E16)
1630 LET H6=(-90.1233E20-(-12345E16))-(-8.88888E21)
1640 IF ABS(H6)<=1E16 THEN 1660
1650 LET A$="*"
1660 PRINT "-90.1233E20   "
1670 PRINT "     -        =","-8.88888E21 ",G5,H6;A$
1680 PRINT " -12345E16    "
1690 PRINT
1700 PRINT "                                   END TEST."
1710 PRINT
1720 PRINT
1730 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

```
PROGRAM FILE 24

        SECTION 24.0: PRIMITIVE OPERATIONS ON MIXED MODES.
```

BEGIN TEST.

SECTION 24.1

++++++++++++
+ ADDITION +
++++++++++++

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

1ST. ADDEND + 2ND. ADDEND		REQUIRED SUM	SUM OF SYSTEM	ABSOLUTE ERROR
12 + 2.5	=	14.5	14.5	0
14 + 12.5E-3	=	14.0125	14.0125	0
-9 + -15E-4	=	-9.0015	-9.0015	0
.625 + -.00005E7	=	-499.375	-499.375	0
1234.2 + 36000E-5	=	1234.56	1234.56	0
65.4321E21 + 12345E17	=	6.66666E22	6.66666E+22	0

END TEST.

BEGIN TEST.

SECTION 24.2

- SUBTRACTION -

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, CHECK TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, CHECK TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

MINUEND - SUBTRAHEND		REQUIRED DIFFERENCE	DIFFERENCE OF SYSTEM	ABSOLUTE ERROR
2 - .76544	=	1.23456	1.23456	0
15 - -.00520E3	=	20.2	20.2	0
-9 - 87600E-5	=	-9.876	-9.876	0
9.8 - .000231E6	=	-222.2	-222.2	0
177.177 - 540540E-4	=	123.123	123.123	0
-90.1233E20 - -12345E16	=	-8.88888E21	-8.88888E+21	0

END TEST.

25.0 ELEMENTARY OPERATIONS ON MIXED TYPE CONSTANTS (CONTINUED)

25.1 Multiplication

The objective of this test is to verify that the operation of multiplication of two numerical constants that are not of the same type will maintain at least six decimal digits of precision. The test is the same as test 20.3, except for the mixing of types. This test has the same output format as test 20.3.

25.2 Division

The objective of this test is to verify that the operation of division between two numerical constants that are not of the same type will maintain at least six decimal digits of precision. The test is the same in structure as test 21.1, except for the mixing of types. This test has the same output format as test 21.1.

25.3 Involution

The objective of this test is to verify that the operation of involution between two numerical constants that are not of the same type will maintain at least six decimal digits of precision. The test is the same structurally to test 21.2, except for the mixing of types. This test has the same output format as test 21.2.

```
********************
* PROGRAM FILE 25 *
********************
```

```
0010 PRINT "PROGRAM FILE 25"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "                                BEGIN TEST."
0100 PRINT
0110 PRINT
0120 PRINT "                          SECTION 25.1"
0130 PRINT
0140 PRINT "                             XXXXXXXXXXXXXXXXXX"
0150 PRINT "                             X MULTIPLICATION X"
0160 PRINT "                             XXXXXXXXXXXXXXXXXX"
0170 PRINT
0180 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0190 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
```

```
0200 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0210 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0220 PRINT "SIX PLACE ACCURACY."
0230 PRINT
0240 PRINT
0250 PRINT "1ST. FACTOR"
0260 PRINT "     *     ","REQUIRED","PRODUCT OF","ABSOLUTE"
0270 PRINT "2ND. FACTOR","PRODUCT ","  SYSTEM  "," ERROR  "
0280 PRINT
0290 LET A$=" "
0300 LET I7=5*1.25
0310 LET J8=(5*1.25)-6.25
0320 IF ABS(J8)<=1E-5 THEN 340
0330 LET A$="*"
0340 PRINT "  5          "
0350 PRINT "  *         =","  6.25 ",I7,J8;A$
0360 PRINT " 1.25        "
0370 PRINT
0380 LET A$=" "
0390 LET J9=21*(-.47619E5)
0400 LET K9=(21*(-.47619E5))-(-999999)
0410 IF ABS(K9)<=1E0 THEN 430
0420 LET A$="*"
0430 PRINT "   21        "
0440 PRINT "    *       =","-999999 ",J9,K9;A$
0450 PRINT "-.47619E5    "
0460 PRINT
0470 LET A$=" "
0480 LET B=-99*(-919100E-2)
0490 LET C0=(-99*(-919100E-2))-909909
0500 IF ABS(C0)<=1E0 THEN 520
0510 LET A$="*"
0520 PRINT "    -99       "
0530 PRINT "     *       =","  909909 ",B,C0;A$
0540 PRINT "-919100E-2    "
0550 PRINT
0560 LET A$=" "
0570 LET D1=.0015*6.25E4
0580 LET E2=(.0015*6.25E4)-93.75
0590 IF ABS(E2)<=1E-4 THEN 610
0600 LET A$="*"
0610 PRINT " .0015        "
0620 PRINT "   *         =","  93.75 ",D1,E2;A$
0630 PRINT " 6.25E4       "
0640 PRINT
0650 LET A$=" "
0660 LET F3=1.92*6430E-4
0670 LET G4=(1.92*6430E-4)-1.23456
0680 IF ABS(G4)<=1E-5 THEN 700
0690 LET A$="*"
0700 PRINT "   1.92       "
0710 PRINT "    *       =","  1.23456 ",F3,G4;A$
0720 PRINT " 6430E-4      "
0730 PRINT
0740 LET A$=" "
0750 LET H5=10.631E27*(-72000E5)
0760 LET I6=(10.631E27*(-72000E5))-(-7.65432E37)
```

```
0770 IF ABS(I6)<=1E32 THEN 790
0780 LET A$="*"
0790 PRINT " 10.631E27  "
0800 PRINT "     *     =","-7.65432E37 ",H5,I6;A$
0810 PRINT "-72000E5    "
0820 PRINT
0830 PRINT "                                END TEST."
0840 PRINT
0850 PRINT
0860 PRINT
0870 PRINT "                                BEGIN TEST."
0880 PRINT
0890 PRINT
0900 PRINT "                          SECTION 25.2"
0910 PRINT
0920 PRINT "                                ////////////"
0930 PRINT "                                / DIVISION /"
0940 PRINT "                                ////////////"
0950 PRINT
0960 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0970 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0980 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0990 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1000 PRINT "SIX PLACE ACCURACY."
1010 PRINT
1020 PRINT
1030 PRINT "DIVIDEND"
1040 PRINT "   /    ","REQURIED","QUOTIENT OF","ABSOLUTE"
1050 PRINT "DIVISOR ","QUOTIENT","   SYSTEM  ","  ERROR "
1060 PRINT
1070 LET A$=" "
1080 LET D=625/1.25
1090 LET D0=(625/1.25)-500
1100 IF ABS(D0)<=1E-3 THEN 1120
1110 LET A$="*"
1120 PRINT " 625        "
1130 PRINT "  /        ="," 500 ",D,D0;A$
1140 PRINT " 1.25       "
1150 PRINT
1160 LET A$=" "
1170 LET E1=84.876E7/(-6875)
1180 LET F2=(84.876E7/(-6875))-(-123456)
1190 IF ABS(F2)<=1E0 THEN 1210
1200 LET A$="*"
1210 PRINT " 84.876E7   "
1220 PRINT "    /      =","-123456 ",E1,F2;A$
1230 PRINT " -6875      "
1240 PRINT
1250 LET A$=" "
1260 LET G2=-198765/(-5E-20)
1270 LET H3=(-198765/(-5E-20))-3.9753E24
1280 IF ABS(H3)<=1E19 THEN 1300
1290 LET A$="*"
1300 PRINT "-198765     "
1310 PRINT "   /       ="," 3.9753E24 ",G2,H3;A$
1320 PRINT "-5E-20      "
1330 PRINT
```

```
1340 LET A$=" "
1350 LET I4=6.25/2.5E-4
1360 LET J5=(6.25/2.5E-4)-25000
1370 IF ABS(J5)<=1E-1 THEN 1390
1380 LET A$="*"
1390 PRINT "  6.25      "
1400 PRINT "   /       =","  25000 ",I4,J5;A$
1410 PRINT " 2.5E-4     "
1420 PRINT
1430 LET A$=" "
1440 LET K4=.1728/12E12
1450 LET L5=(.1728/12E12)-1.44000E-14
1460 IF ABS(L5)<=1E-19 THEN 1480
1470 LET A$="*"
1480 PRINT " .1728      "
1490 PRINT "   /       =","  1.44000E-14 ",K4,L5;A$
1500 PRINT " 12E12      "
1510 PRINT
1520 LET A$=" "
1530 LET K5=-1.25E-10/625E16
1540 LET L6=(-1.25E-10/625E16)-(-2.00000E-29)
1550 IF ABS(L6)<=1E-34 THEN 1570
1560 LET A$="*"
1570 PRINT "-1.25E-10   "
1580 PRINT "    /      =","-2.00000E-29 ",K5,L6;A$
1590 PRINT "  625E16    "
1600 PRINT
1610 PRINT "                                   END TEST."
1620 PRINT
1630 PRINT
1640 PRINT
1650 PRINT "                          SECTION 25.3"
1660 PRINT
1670 PRINT "                          ^^^^^^^^^^^^"
1673 PRINT "                          ^INVOLUTION^"
1675 PRINT "                          ^^^^^^^^^^^^"
1680 PRINT
1690 PRINT
1700 PRINT "                                   BEGIN TEST."
1710 PRINT
1720 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1730 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1740 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1750 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1760 PRINT "SIX PLACE ACCURACY."
1770 PRINT
1780 PRINT
1790 PRINT "  BASE   "
1800 PRINT "    ^    ","REQUIRED","POWER OF","ABSOLUTE"
1810 PRINT "EXPONENT","  POWER  "," SYSTEM "," ERROR  "
1820 PRINT
1830 LET A$=" "
1840 LET L6=144^.5
1850 LET L7=(144^.5)-12
1860 IF ABS(L7)<=1E-4 THEN 1880
1870 LET A$="*"
1880 PRINT " 144        "
```

```
1890 PRINT "  ^        =","  12 ",L6,L7;A$
1900 PRINT "  .5        "
1910 PRINT
1920 LET A$=" "
1930 LET M6=5^(-.004E3)
1940 LET M7=(5^(-.004E3))-1.6E-3
1950 IF ABS(M7)<=1E-6 THEN 1970
1960 LET A$="*"
1970 PRINT "   5        "
1980 PRINT "   ^        =","  .0016 ",M6,M7;A$
1990 PRINT "-.004E3     "
2000 PRINT
2010 LET A$=" "
2020 LET L8=65536^(-625E-4)
2030 LET M8=(65536^(-625E-4))-5E-1
2040 IF ABS(M8)<=1E-6 THEN 2060
2050 LET A$="*"
2060 PRINT " 65536       "
2070 PRINT "   ^         =","  .5 ",L8,M8;A$
2080 PRINT "-625E-4      "
2090 PRINT
2100 LET A$=" "
2110 LET L9=.03125^(-.0002E3)
2120 LET M9=(.03125^(-.0002E3))-2
2130 IF ABS(M9)<=1E-5 THEN 2150
2140 LET A$="*"
2150 PRINT " .03125       "
2160 PRINT "   ^         =","  2 ",L9,M9;A$
2170 PRINT "-.0002E3     "
2180 PRINT
2190 LET A$=" "
2200 LET N6=1.2^5000E-3
2210 LET N7=(1.2^5000E-3)-2.48832
2220 IF ABS(N7)<=1E-5 THEN 2240
2230 LET A$="*"
2240 PRINT "   1.2       "
2250 PRINT "    ^        =","  2.48832 ",N6,N7;A$
2260 PRINT " 5000E-3     "
2270 PRINT
2280 LET A$=" "
2290 LET N8=1.024E13^(-10E-2)
2300 LET N9=(1.024E13^(-10E-2))-5E-2
2310 IF ABS(N9)<=1E-6 THEN 2330
2320 LET A$="*"
2330 PRINT " 1.024E13    "
2340 PRINT "    ^        =","  .05 ",N8,N9;A$
2350 PRINT " -10E-2      "
2360 PRINT
2370 PRINT "                                    END TEST."
2380 PRINT
2390 PRINT
2400 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 25

BEGIN TEST.

SECTION 25.1

XXXXXXXXXXXXXXXXX
X MULTIPLICATION X
XXXXXXXXXXXXXXXXX

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

1ST. FACTOR * 2ND. FACTOR		REQUIRED PRODUCT	PRODUCT OF SYSTEM	ABSOLUTE ERROR
5 * 1.25	=	6.25	6.25	0
21 * -.47619E5	=	-999999	-999999	0
-99 * -919100E-2	=	909909	909909	0
.0015 * 6.25E4	=	93.75	93.75	0
1.92 * 6430E-4	=	1.23456	1.23456	0
10.631E27 * -72000E5	=	-7.65432E37	-7.65432E+37	0

END TEST.

BEGIN TEST.

SECTION 25.2

/////////////
/ DIVISION /
/////////////

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, CHECK TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

DIVIDEND / DIVISOR		REQUIRED QUOTIENT	QUOTIENT OF SYSTEM	ABSOLUTE ERROR
625 / 1.25	=	500	500	0
84.876E7 / -0875	=	-123456	-123456	0
-198765 / -5E-20	=	3.97530E24	3.97530E+24	0
6.25 / 2.5E-4	=	25000	25000	0
.1728 / 12E12	=	1.44000E-14	1.44000E-14	0
-1.25E-10 / 625E16	=	-2.00000E-29	-2.00000E-29	0

END TEST.

SECTION 25.3

^^^^^^^^^^^^
^INVOLUTION^
^^^^^^^^^^^^

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

BASE ^ EXPONENT		REQUIRED POWER	POWER OF SYSTEM	ABSOLUTE ERROR
144 ^ .5	=	12	12	0
5 ^ -.004E3	=	.0016	.0016	0
65536 ^ -625E-4	=	.5	.5	0
.03125 ^ -.0002E3	=	2	2	0
1.2 ^ 5000E-3	=	2.48832	2.48832	0
1.024E13 ^ -10E-2	=	.05	.05	0

END TEST.

26.0 ELEMENTARY OPERATIONS ON VARIABLES ASSIGNED MIXED TYPE CONSTANTS

The objective of these tests is to verify further that numerical expressions can be constructed by mixing simple numeric variables that have not been assigned the same type of numeric constant, with the operations of addition subtraction, multiplication, division, and involution.

26.1 Addition

The objective of this test is to verify that the operation of addition of two numerically assigned simple variables, that have not been assigned the same type of numeric constants, maintains at least six decimal digits of precision. The test is similar in structure and output format to test 22.1.

26.2 Subtraction

The objective of this test is to verify that the operation of subtraction between two numerically assigned simple variables, that have not been assigned the same type constants, maintains at least six decimal digits of precision. The test is similar in structure and output format to test 22.2.

```
********************
* PROGRAM FILE 26 *
********************
```

```
0010 PRINT "PROGRAM FILE 26"
0060 PRINT
0070 PRINT
0080 PRINT
0140 PRINT "                                BEGIN TEST."
0150 PRINT
0160 PRINT
0170 PRINT "                          SECTION 26.1"
0180 PRINT
0190 PRINT "                                ++++++++++++"
0200 PRINT "                                + ADDITION +"
0210 PRINT "                                ++++++++++++"
0220 PRINT
0230 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0240 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0250 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0260 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0270 PRINT "SIX PLACE ACCURACY."
```

```
0280 PRINT
0290 PRINT
0300 PRINT "ASSIGNMENT 1"
0310 PRINT "     +      ","REQUIRED","SUM OF","ABSOLUTE"
0320 PRINT "ASSIGNMENT 2","  SUM    ","SYSTEM"," ERROR  "
0330 PRINT
0340 LET A$=" "
0350 LET K=12
0360 LET K0=2.5
0370 LET K1=-14.5
0380 LET K2=K+K0
0390 LET K3=K2+K1
0400 IF ABS(K3)<=1E-3 THEN 420
0410 LET A$="*"
0420 PRINT " 12          "
0430 PRINT "  +         =","  14.5 ",K2,K3;A$
0440 PRINT " 2.5         "
0450 PRINT
0460 LET A$=" "
0470 LET L=14
0480 LET M0=12.5E-3
0490 LET N1=-.140125E2
0500 LET O2=L+M0
0510 LET P3=O2+N1
0520 IF ABS(P3)<=1E-4 THEN 540
0530 LET A$="*"
0540 PRINT "   14        "
0550 PRINT "    +       =","  14.0125 ",O2,P3;A$
0560 PRINT " 12.5E-3     "
0570 PRINT
0580 LET A$=" "
0590 LET A5=-9
0600 LET B5=-15E-4
0610 LET C5=9.0015
0620 LET D5=A5+B5
0630 LET E5=D5+C5
0640 IF ABS(E5)<=1E-5 THEN 660
0650 LET A$="*"
0660 PRINT " -9          "
0670 PRINT "  +         =","-9.0015 ",D5,E5;A$
0680 PRINT "-15E-4       "
0690 PRINT
0700 LET A$=" "
0710 LET F5=.625
0720 LET E6=-.00005E7
0730 LET D7=499.375
0740 LET C8=F5+E6
0750 LET B9=C8+D7
0760 IF ABS(B9)<=1E-3 THEN 780
0770 LET A$="*"
0780 PRINT " .625        "
0790 PRINT "   +        =","-499.375 ",C8,B9;A$
0800 PRINT "-.0005E7     "
0810 PRINT
0820 LET A$=" "
0830 LET A4=1234.2
0840 LET A6=36000E-5
```

```
0850 LET A7=-1234.56
0860 LET A8=A4+A6
0870 LET A9=A8+A7
0880 IF ABS(A9)<=1E-2 THEN 900
0890 LET A$="*"
0900 PRINT " 1234.2       "
0910 PRINT "    +        =","  1234.56 ",A8,A9;A$
0920 PRINT " 36000E-5     "
0930 PRINT
0940 LET A$=" "
0950 LET C9=65.4321E21
0960 LET D9=12345E17
0970 LET E9=-6.66666E22
0980 LET F9=C9+D9
0990 LET G9=F9+E9
1000 IF ABS(G9)<=1E17 THEN 1020
1010 LET A$="*"
1020 PRINT " 65.4321E21   "
1030 PRINT "     +       =","  6.66666E22 ",F9,G9;A$
1040 PRINT " 12345E17     "
1050 PRINT
1060 PRINT "                                  END TEST."
1070 PRINT
1080 PRINT
1090 PRINT
1100 PRINT "                                 BEGIN TEST."
1110 PRINT
1120 PRINT
1130 PRINT "                          SECTION 26.2"
1140 PRINT
1150 PRINT "                                 ---------------"
1160 PRINT "                                 - SUBTRACTION -"
1170 PRINT "                                 ---------------"
1180 PRINT
1190 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1200 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1205 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1210 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1220 PRINT "SIX PLACE ACCURACY."
1230 PRINT
1240 PRINT
1250 PRINT "ASSIGNMENT 1","           ","DIFFERENCE"
1260 PRINT "     -      "," REQUIRED ","    OF    ","ABSOLUTE"
1270 PRINT "ASSIGNMENT 2","DIFFERENCE","  SYSTEM  "," ERROR  "
1280 PRINT
1290 LET A$=" "
1300 LET H7=2
1310 LET H8=.76544
1320 LET I8=1.23456
1330 LET H9=H7-H8
1340 LET I9=H9-I8
1350 IF ABS(I9)<=1E-5 THEN 1370
1360 LET A$="*"
1370 PRINT "    2         "
1380 PRINT "    -        =","  1.23456 ",H9,I9;A$
1390 PRINT " .76544       "
1400 PRINT
```

```
1410 LET A$=" "
1420 LET A0=15
1430 LET A1=-.00520E3
1440 LET B1=.202E2
1450 LET A2=A0-A1
1460 LET B2=A2-B1
1470 IF ABS(B2)<=1E-4 THEN 1490
1480 LET A$="*"
1490 PRINT "   15         "
1500 PRINT "    -        =","  20.2 ",A2,B2;A$
1510 PRINT "-.00520E3     "
1520 PRINT
1530 LET A$=" "
1540 LET C2=-9
1550 LET A3=87600E-5
1560 LET B3=-9876E-3
1570 LET H4=C2-A3
1580 LET B4=H4-B3
1590 IF ABS(B4)<=1E-5 THEN 1610
1600 LET A$="*"
1610 PRINT "   -9         "
1620 PRINT "    -        =","-9.876 ",H4,B4;A$
1630 PRINT " 87600E-5     "
1640 PRINT
1650 LET A$=" "
1660 LET C=8.8
1670 LET C3=.000231E6
1680 LET D3=-222.2
1690 LET C4=C-C3
1700 LET D4=C4-D3
1710 IF ABS(D4)<=1E-3 THEN 1730
1720 LET A$="*"
1730 PRINT "    8.8        "
1740 PRINT "     -        =","-222.2 ",C4,D4;A$
1750 PRINT " .000231E6    "
1760 PRINT
1770 LET A$=" "
1780 LET B6=177.177
1790 LET C6=540540E-4
1800 LET B7=123.123
1810 LET C7=B6-C6
1820 LET B8=C7-B7
1830 IF ABS(B8)<=1E-3 THEN 1850
1840 LET A$="*"
1850 PRINT "  177.177     "
1860 PRINT "     -        =","  123.123 ",C7,B8;A$
1870 PRINT " 540540E-4    "
1880 PRINT
1890 LET A$=" "
1900 LET E=-90.1233E20
1910 LET F=-12345E16
1920 LET E0=-8.88888E21
1930 LET F0=E-F
1940 LET F1=F0-E0
1950 IF ABS(F1)<=1E16 THEN 1970
1960 LET A$="*"
1970 PRINT "-90.1233E20   "
```

```
1980 PRINT "      -       =","-8.88888E21 ",F0,F1;A$
1990 PRINT " -12345E16    "
2000 PRINT
2010 PRINT "                                    END TEST."
2020 PRINT
2030 PRINT
2040 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

```
PROGRAM FILE 26

                          BEGIN TEST.

                    SECTION 26.1

                         ++++++++++++
                         + ADDITION +
                         ++++++++++++

     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR
COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS
A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-
CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF
SIX PLACE ACCURACY.


ASSIGNMENT 1
     +         REQUIRED       SUM OF         ABSOLUTE
ASSIGNMENT 2     SUM          SYSTEM          ERROR

 12
  +          =   14.5           14.5           0
 2.5

   14
    +        =   14.0125        14.0125        0
 12.5E-3

 -9
  +          = -9.0015         -9.0015         0
-15E-4

 .625
```

```
   +          = -499.375       -499.375        0
-.0005E7

 1234.2
   +          =  1234.56        1234.56        0
 36000E-5

 65.4321E21
   +          =  6.66666E22     6.66666E+22    0
 12345E17
```

END TEST.

BEGIN TEST.

SECTION 26.2

- SUBTRACTION -

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

```
ASSIGNMENT 1                  DIFFERENCE
     -          REQUIRED         OF          ABSOLUTE
ASSIGNMENT 2  DIFFERENCE       SYSTEM         ERROR

    2
    -         =  1.23456        1.23456        0
 .76544

   15
    -         =  20.2           20.2           0
-.00520E3

   -9
    -         = -9.876         -9.876          0
 87600E-5

    8.8
    -         = -222.2         -222.2          0
 .000231E6

  177.177
    -         =  123.123        123.123        0
 540540E-4

-90.1233E20
    -         = -8.88888E21    -8.88888E+21    0
```

```
-12345E16

                    END TEST.
```

27.0 ELEMENTARY OPERATIONS ON VARIABLES ASSIGNED MIXED TYPE CONSTANTS (CONTINUED)

27.1 Multiplication

The objective of this test is to verify that the operation of multiplication of two numerically assigned simple variables, which have not been assigned the same type constants, will maintain at least six decimal digits of precision. This test is similar in structure and output format to test 23.1.

27.2 Division

The objective of this test is to verify that the operation of division between two numerically assigned simple variables, which have not been assigned the same type numeric constants, will maintain at least six decimal digits of precision. The test is similar in structure and output format to test 23.2.

27.3 Involution

The objective of this test is to verify that the operation of involution of two numerically assigned simple variables, which have not been assigned the same type constants, will maintain at least six decimal digits of precision. The test is similar in structure and output format to test 23.3.

```
********************
* PROGRAM FILE 27 *
********************
```

```
0010 PRINT "PROGRAM FILE 27"
0020 PRINT
0030 PRINT
0040 PRINT
0090 PRINT "                                BEGIN TEST."
0100 PRINT
0110 PRINT
0120 PRINT "                          SECTION 27.1"
0130 PRINT
0140 PRINT "                              XXXXXXXXXXXXXXXXXX"
0150 PRINT "                              X MULTIPLICATION X"
0160 PRINT "                              XXXXXXXXXXXXXXXXXX"
```

```
0170 PRINT
0180 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0190 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0200 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0210 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0220 PRINT "SIX PLACE ACCURACY."
0230 PRINT
0240 PRINT
0250 PRINT "ASSIGNMENT 1"
0260 PRINT "     *      ","REQUIRED","PRODUCT OF","ABSOLUTE"
0270 PRINT "ASSIGNMENT 2","PRODUCT ","  SYSTEM  "," ERROR  "
0280 PRINT
0290 LET A$=" "
0300 LET G=5
0310 LET H=1.25
0320 LET I=6.25
0330 LET J=G*H
0340 LET G0=J-I
0350 IF ABS(G0)<=1E-5 THEN 370
0360 LET A$="*"
0370 PRINT "  5          "
0380 PRINT "  *         =","  6.25 ",J,G0;A$
0390 PRINT " 1.25        "
0400 PRINT
0410 LET A$=" "
0420 LET H0=21
0430 LET I0=-.47619E5
0440 LET J0=-999999
0450 LET G1=H0*I0
0460 LET H1=G1-J0
0470 IF ABS(H1)<=1E0 THEN 490
0480 LET A$="*"
0490 PRINT "   21         "
0500 PRINT "    *       =","-999999 ",G1,H1;A$
0510 PRINT "-.47619E5    "
0520 PRINT
0530 LET A$=" "
0540 LET I1=-99
0550 LET J1=-919100E-2
0560 LET H2=909909
0570 LET I2=I1*J1
0580 LET J2=I2-H2
0590 IF ABS(J2)<=1E0 THEN 610
0600 LET A$="*"
0610 PRINT "    -99       "
0620 PRINT "     *      =","  909909 ",I2,J2;A$
0630 PRINT "-919100E-2   "
0640 PRINT
0650 LET A$=" "
0660 LET I3=.0015
0670 LET J3=6.25E4
0680 LET J4=93.75
0690 LET L0=I3*J3
0700 LET L1=L0-J4
0710 IF ABS(L1)<=1E-4 THEN 730
0720 LET A$="*"
0730 PRINT " .0015       "
```

```
0740 PRINT "    *       =","  93.75 ",L0,L1;A$
0750 PRINT " 6.25E4      "
0760 PRINT
0770 LET A$=" "
0780 LET M1=1.92
0790 LET L2=6430E-4
0800 LET M2=1.23456
0810 LET N2=M1*L2
0820 LET L3=N2-M2
0830 IF ABS(L3)<=1E-5 THEN 850
0840 LET A$="*"
0850 PRINT "   1.92      "
0860 PRINT "    *       =","  1.23456 ",N2,L3;A$
0870 PRINT " 6430E-4     "
0880 PRINT
0890 LET A$=" "
0900 LET M3=10.631E27
0910 LET N3=-72000E5
0920 LET L4=-7.65432E37
0930 LET M4=M3*N3
0940 LET N4=M4-L4
0950 IF ABS(N4)<=1E32 THEN 970
0960 LET A$="*"
0970 PRINT " 10.631E27   "
0980 PRINT "     *      =","-7.65432E37 ",M4,N4;A$
0990 PRINT "-72000E5     "
1000 PRINT
1010 PRINT "                                  END TEST."
1020 PRINT
1030 PRINT
1040 PRINT
1050 PRINT "                                BEGIN TEST."
1060 PRINT
1070 PRINT
1080 PRINT "                    SECTION 27.2"
1090 PRINT
1100 PRINT "                                ////////////"
1110 PRINT "                                / DIVISION /"
1120 PRINT "                                ////////////"
1130 PRINT
1140 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
1150 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
1160 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
1170 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
1180 PRINT "SIX PLACE ACCURACY."
1190 PRINT
1200 PRINT
1210 PRINT "ASSIGNMENT 1"
1220 PRINT "     /       ","REQUIRED","QUOTIENT OF","ABSOLUTE"
1230 PRINT "ASSIGNMENT 2","QUOTIENT","   SYSTEM   ","  ERROR "
1240 PRINT
1250 LET A$=" "
1260 LET O3=625
1270 LET O4=1.25
1280 LET M5=500
1290 LET N5=O3/O4
1300 LET O5=N5-M5
```

```
1310 IF ABS(O5)<=1E-3 THEN 1330
1320 LET A$="*"
1330 PRINT " 625        "
1340 PRINT "  /        =","  500 ",N5,O5;A$
1350 PRINT " 1.25      "
1360 PRINT
1370 LET A$=" "
1380 LET Q4=84.876E7
1390 LET R5=-6875
1400 LET S6=-123456
1410 LET T7=Q4/R5
1420 LET U8=T7-S6
1430 IF ABS(U8)<=1E0 THEN 1450
1440 LET A$="*"
1450 PRINT " 84.876E7  "
1460 PRINT "    /      =","-123456 ",T7,U8;A$
1470 PRINT " -6875     "
1480 PRINT
1490 LET A$=" "
1500 LET P4=-198765
1510 LET P5=-5E-20
1520 LET Q5=39753E20
1530 LET J6=P4/P5
1540 LET K6=J6-Q5
1550 IF ABS(K6)<=1E19 THEN 1570
1560 LET A$="*"
1570 PRINT "-198765     "
1580 PRINT "   /       =","  3.9753E24 ",J6,K6;A$
1590 PRINT "-5E-20      "
1600 PRINT
1610 LET A$=" "
1620 LET T=6.25
1630 LET T0=2.5E-4
1640 LET T1=25000.0
1650 LET T2=T/T0
1660 LET T3=T2-T1
1670 IF ABS(T3)<=1E-1 THEN 1690
1680 LET A$="*"
1690 PRINT "  6.25      "
1700 PRINT "   /       =","  25000 ",T2,T3;A$
1710 PRINT " 2.5E-4     "
1720 PRINT
1730 LET A$=" "
1740 LET M=.1728
1750 LET N=12E12
1760 LET O=144000E-19
1770 LET P=M/N
1780 LET Q=P-O
1790 IF ABS(Q)<=1E-19 THEN 1810
1800 LET A$="*"
1810 PRINT " .1728      "
1820 PRINT "   /       =","  1.44000E-14 ",P,Q;A$
1830 PRINT " 12E12      "
1840 PRINT
1850 LET A$=" "
1860 LET R=-1.25E-10
1870 LET S=625E16
```

```
1880 LET N0=-2.00000E-29
1890 LET O0=R/S
1900 LET P0=O0-N0
1910 IF ABS(P0)<=1E-34 THEN 1930
1920 LET A$="*"
1930 PRINT "-1.25E-10  "
1940 PRINT "    /     =","-2.00000E-29 ",O0,P0;A$
1950 PRINT "  625E16   "
1960 PRINT
1970 PRINT "                                   END TEST."
1980 PRINT
1990 PRINT
2000 PRINT
2010 PRINT "                          SECTION 27.3"
2020 PRINT
2030 PRINT "                         ^^^^^^^^^^^^"
2033 PRINT "                         ^INVOLUTION^"
2035 PRINT "                         ^^^^^^^^^^^^"
2040 PRINT
2050 PRINT
2060 PRINT "                                   BEGIN TEST."
2070 PRINT
2080 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
2090 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
2100 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
2110 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
2120 PRINT "SIX PLACE ACCURACY."
2130 PRINT
2140 PRINT
2150 PRINT "ASSIGNMENT 1"
2160 PRINT "      ^       ","REQUIRED","POWER OF","ABSOLUTE"
2170 PRINT "ASSIGNMENT 2"," POWER  "," SYSTEM "," ERROR  "
2180 PRINT
2190 LET A$=" "
2200 LET N0=144
2210 LET O0=.5
2220 LET P0=12
2230 LET Q0=N0^O0
2240 LET R0=Q0-P0
2250 IF ABS(R0)<=1E-4 THEN 2270
2260 LET A$="*"
2270 PRINT " 144        "
2280 PRINT "  ^        =","  12 ",Q0,R0;A$
2290 PRINT "  .5        "
2300 PRINT
2310 LET A$=" "
2320 LET O1=5
2330 LET P1=-.004E3
2340 LET Q1=1.6E-3
2350 LET R1=O1^P1
2360 LET S1=R1-Q1
2370 IF ABS(S1)<=1E-6 THEN 2390
2380 LET A$="*"
2390 PRINT "   5        "
2400 PRINT "   ^       =","  .0016 ",R1,S1;A$
2410 PRINT "-.004E3     "
2420 PRINT
```

```
2430 LET A$=" "
2440 LET U=65536
2450 LET V=-625E-4
2460 LET W=5E-1
2470 LET X=U^V
2480 LET Y=X-W
2490 IF ABS(Y)<=1E-6 THEN 2510
2500 LET A$="*"
2510 PRINT " 65536      "
2520 PRINT "    ^       =","  .5 ",X,Y;A$
2530 PRINT "-625E-4     "
2540 PRINT
2550 LET A$=" "
2560 LET R=.03125
2570 LET S=-.0002E3
2580 LET Z=2
2590 LET R0=R^S
2600 LET S0=R0-Z
2610 IF ABS(S0)<=1E-5 THEN 2630
2620 LET A$="*"
2630 PRINT " .03125     "
2640 PRINT "    ^       =","  2 ",R0,S0;A$
2650 PRINT "-.0002E3    "
2660 PRINT
2670 LET A$=" "
2680 LET U0=1.2
2690 LET V0=5000E-3
2700 LET W0=2.48832
2710 LET X0=U0^V0
2720 LET Y0=X0-W0
2730 IF ABS(Y0)<=1E-5 THEN 2750
2740 LET A$="*"
2750 PRINT "  1.2       "
2760 PRINT "    ^       =","  2.48832 ",X0,Y0;A$
2770 PRINT " 5000E-3    "
2780 PRINT
2790 LET A$=" "
2800 LET U1=1.024E13
2810 LET V1=-10E-2
2820 LET W1=5E-2
2830 LET X1=U1^V1
2840 LET Y1=X1-W1
2850 IF ABS(Y1)<=1E-6 THEN 2870
2860 LET A$="*"
2870 PRINT " 1.024E13   "
2880 PRINT "    ^       =","  .05 ",X1,Y1;A$
2890 PRINT " -10E-2     "
2900 PRINT
2910 PRINT "                                    END TEST."
2920 PRINT
2930 PRINT
2940 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 27

BEGIN TEST.

SECTION 27.1

XXXXXXXXXXXXXXXXX
X MULTIPLICATION X
XXXXXXXXXXXXXXXXX

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 * ASSIGNMENT 2		REQUIRED PRODUCT	PRODUCT OF SYSTEM	ABSOLUTE ERROR
5 * 1.25	=	6.25	6.25	0
21 * -.47619E5	=	-999999	-999999	0
-99 * -919100E-2	=	909909	909909	0
.0015 * 6.25E4	=	93.75	93.75	0
1.92 * 6430E-4	=	1.23456	1.23456	0
10.631E27 * -72000E5	=	-7.65432E37	-7.65432E+37	0

END TEST.

BEGIN TEST.

SECTION 27.2

////////////
/ DIVISION /
////////////

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 / ASSIGNMENT 2		REQUIRED QUOTIENT	QUOTIENT OF SYSTEM	ABSOLUTE ERROR
625 / 1.25	=	500	500	0
84.876E7 / -6875	=	-123456	-123456	0
-198765 / -5E-20	=	3.9753E24	3.97530E+24	0
6.25 / 2.5E-4	=	25000	25000	0
.1728 / 12E12	=	1.44000E-14	1.44000E-14	0
-1.25E-10 / 625E16	=	-2.00000E-29	-2.00000E-29	0

END TEST.

SECTION 27.3

^^^^^^^^^^^^
^INVOLUTION^
^^^^^^^^^^^^

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

ASSIGNMENT 1 ^ ASSIGNMENT 2		REQUIRED POWER	POWER OF SYSTEM	ABSOLUTE ERROR
144 ^ .5	=	12	12	0
5 ^ -.004E3	=	.0016	.0016	0
65536 ^ -625E-4	=	.5	.5	0
.03125 ^ -.0002E3	=	2	2	0
1.2 ^ 5000E-3	=	2.48832	2.48832	0
1.024E13 ^ -10E-2	=	.05	.05	0

END TEST.

28.0 ADDITION OF THREE OR MORE TERMS

The objective of this section is to verify that three or more terms can be used in the construction of numerical expressions using the operation of addition.

28.1 Using only Numerical Constants

The objective of this test is to verify that the operation of addition of three or more numerical constants will maintain at least six decimal digits of precision. The test forms the sum of three or more numerical constants of different type. There are four columns of output of which the first column is labeled "Number of Terms," the second column is labeled "Required Sum," the third column is labeled "Sum of System," and the fourth column is labeled "Absolute Error." The first column lists numbers indicating the number of terms that should have been summed for each row of output; the second column lists a standard conforming output; the third column lists the sums of the constants as evaluated by the system being tested; and the fourth column contains lists of the differences between the expected values and the system generated values. If the implementation did not maintain at least six decimal digits of precision, an asterisk will appear beside any such case.

28.2 Using only Numerically Assigned Variables

The objective of this test is to verify that the operation of addition of three or more numerically assigned simple variables will maintain at least six decimal digits of precision. The test is similar to 28.1, except for the use of simple variables. The test has the same output format as test 28.1.

28.3 Numerical Constants and Assigned Variables, Mixed

The objective of this test is to construct numerical expressions using both numerical constants and simple variables together. It verifies that the operation of addition of several terms composed of both numerical constants and simple variables will maintain at least six decimal digits of precision. The test is similar, both in structure and output to test 28.1.

```
**********
* PROGRAM FILE 28 *
**********
```

```
0010 PRINT "PROGRAM FILE 28"
0060 PRINT
0070 PRINT
```

```
0080 PRINT
0090 PRINT "              SECTION 28.0: ADDITION OF THREE OR MORE TERMS."
0100 PRINT
0110 PRINT
0120 PRINT
0130 PRINT "                                  BEGIN TEST."
0140 PRINT
0150 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0160 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0170 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0180 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0190 PRINT "SIX PLACE ACCURACY."
0200 PRINT
0210 PRINT
0220 PRINT "NUMBER OF","REQUIRED","SUM OF","ABSOLUTE"
0230 PRINT "  TERMS  "," SUM   ","SYSTEM"," ERROR  "
0240 PRINT "*************************************************";
0250 PRINT "**********************"
0260 PRINT
0270 PRINT
0280 PRINT "                          SECTION 28.1"
0290 PRINT
0300 PRINT "                   (USING ONLY NUMERICAL CONSTANTS.)"
0310 PRINT
0320 PRINT
0330 LET A$=" "
0340 LET A=12+56+92
0350 LET A1=(12+56+92)+(-160)
0360 IF ABS(A1)<=1E-3 THEN 380
0370 LET A$="*"
0380 PRINT "    3    "," 160 ",A,A1;A$
0390 LET A$=" "
0400 LET B=4.9+95+9E2+9.9E-2
0410 LET B1=(4.9+95+9E2+9.9E-2)+(-999.999)
0420 IF ABS(B1)<=1E-3 THEN 440
0430 LET A$="*"
0440 PRINT "    4    "," 999.999 ",B,B1;A$
0450 PRINT
0460 PRINT
0470 PRINT "*************************************************";
0480 PRINT "**********************"
0490 PRINT
0500 PRINT
0510 PRINT "                          SECTION 28.2"
0520 PRINT
0530 PRINT "              (USING ONLY NUMERICALLY ASSIGNED VARIABLES.)"
0540 PRINT
0550 PRINT
0560 LET A$=" "
0570 LET A2=2.4
0580 LET A3=23.05
0590 LET A4=230.004
0600 LET A5=432.1
0610 LET A6=300.1
0620 LET A7=314
0630 LET A8=84E-2
0640 LET A9=.5E-2
```

```
0650 LET A0=-987.654
0660 LET C=A2+A3+A4+A5+A6
0670 LET C1=C+A0
0680 IF ABS(C1)<=1E-3 THEN 700
0690 LET A$="*"
0700 PRINT "    5    "," 987.654 ",C,C1;A$
0710 LET A$=" "
0720 LET B2=-999.999
0730 LET B3=A3+A4+A5+A7+A8+A9
0740 LET B4=B3+B2
0750 IF ABS(B4)<=1E-3 THEN 770
0760 LET A$="*"
0770 PRINT "    6    "," 999.999 ",B3,B4;A$
0780 PRINT
0790 PRINT
0800 PRINT "*****************************************************";
0810 PRINT "**********************"
0820 PRINT
0830 PRINT
0840 PRINT "                          SECTION 28.3"
0850 PRINT
0860 PRINT "    (NUMERICAL CONS. AND ASSIGNED NUMERICAL VARS., MIXED.)"
0870 PRINT
0880 PRINT
0890 LET A$=" "
0900 LET C2=110000
0910 LET C3=10.1E7
0920 LET C4=7.4E8
0930 LET C5=8E9
0940 LET C6=1.6E32
0950 LET C7=1.2E34
0960 LET C8=1.0E36
0970 LET D1=C6+1.3E33+C7+100.1E33+100.0E33+C8+2.1E34
0980 LET D2=D1+(-1.23456E36)
0990 IF ABS(D2)<=1E31 THEN 1010
1000 LET A$="*"
1010 PRINT "    7    "," 1.23456E36 ",D1,D2;A$
1020 LET A$=" "
1030 LET D3=80000.0+C2+58E5+C3+5.3E7+C4+1.1E9+C5
1040 LET D4=D3+(-9.99999E9)
1050 IF ABS(D4)<=1E4 THEN 1070
1060 LET A$="*"
1070 PRINT "    8    "," 9.99999E9 ",D3,D4;A$
1080 PRINT
1090 PRINT
1100 PRINT "*****************************************************";
1110 PRINT "**********************"
1120 PRINT
1130 PRINT "                                  END TEST."
1140 PRINT
1150 PRINT
1160 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 28

SECTION 28.0: ADDITION OF THREE OR MORE TERMS.

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

NUMBER OF TERMS	REQUIRED SUM	SUM OF SYSTEM	ABSOLUTE ERROR

SECTION 28.1

(USING ONLY NUMERICAL CONSTANTS.)

3	160	160	0
4	999.999	999.999	0

SECTION 28.2

(USING ONLY NUMERICALLY ASSIGNED VARIABLES.)

5	987.654	987.654	0
6	999.999	999.999	0

SECTION 28.3

(NUMERICAL CONS. AND ASSIGNED NUMERICAL VARS., MIXED.)

```
    7           1.23456E36     1.23456E+36    0
    8           9.99999E9      9.99999E+9     0

*************************************************************************

                                END TEST.
```

29.0 MULTIPLICATION OF THREE OR MORE FACTORS

The purpose of this section is to show that three or more factors can be used in the construction of numerical expressions using the operation of multiplication.

29.1 Using Numerical Constants Only

The objective of this test is to verify that the operation of multiplication of three or more numerical constants will maintain at least six decimal digits of precision. The test forms the products of three or more factors of different types. There are four columns of output, of which the first column is labeled "Number of Terms", the second column is labeled "Required Product", the third column is labeled "Product of System", and the fourth column is labeled "Absolute Error". The first column lists the number of factors that should have been multiplied for each row, the second column lists the standard conforming expected output, the third column lists the products of the constants as evaluated by the system being tested, and the fourth column lists the difference between the expected values and the system generated values. If the implementation did not maintain at least six decimal digits of precision, an asterisk will appear beside any such case.

29.2 Using Only Assigned Variables

The objective of this test is to verify that the operation of multiplication of three or more numerically assigned simple variables will maintain at least six decimal digits of precision. The test is similar both in structure and output format to test 29.1.

29.3 Numerical Constants and Assigned Numerical Variables, Mixed

The objective of this test is to verify that the operation of multiplication upon several numerical constants and simple variables will maintain at least six decimal digits of precision. The test is similar in structure and output to test 29.1.

```
**********************
* PROGRAM FILE 29 *
**********************
```

```
0010 PRINT "PROGRAM FILE 29"
0060 PRINT
0070 PRINT
0080 PRINT
```

```
0090 PRINT "      SECTION 29.0: MULTIPLICATION OF THREE OR MORE FACTORS."
0100 PRINT
0110 PRINT
0120 PRINT
0130 PRINT "                                BEGIN TEST."
0140 PRINT
0150 PRINT "     IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR"
0160 PRINT "COLUMN, TEST PASSED.  HOWEVER, IF AN ASTERISK FOLLOWS"
0170 PRINT "A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BE-"
0180 PRINT "CAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF"
0190 PRINT "SIX PLACE ACCURACY."
0200 PRINT
0210 PRINT
0220 PRINT "NUMBER OF","REQUIRED","PRODUCT OF","ABSOLUTE"
0230 PRINT "  TERMS  ","PRODUCT ","  SYSTEM  "," ERROR  "
0240 PRINT "*************************************************";
0250 PRINT "**********************"
0260 PRINT
0270 PRINT
0280 PRINT "                          SECTION 29.1"
0290 PRINT
0300 PRINT "                   (USING ONLY NUMERICAL CONSTANTS.)"
0310 PRINT
0320 PRINT
0330 LET A$=" "
0340 LET Z0=2*4*8179
0350 LET Z1=(2*4*8179)+(-65432)
0360 IF ABS(Z1)<=1E-1 THEN 380
0370 LET A$="*"
0380 PRINT "    3    "," 65432 ",Z0,Z1;A$
0390 LET A$=" "
0400 LET P2=3*.1*9E-4*3.7037E24
0410 LET Q2=(3*.1*9E-4*3.7037E24)+(-9.99999E20)
0420 IF ABS(Q2)<=1E16 THEN 440
0430 LET A$="*"
0440 PRINT "    4    "," 9.99999E20 ",P2,Q2;A$
0450 PRINT
0460 PRINT
0470 PRINT "*************************************************";
0480 PRINT "**********************"
0490 PRINT
0500 PRINT
0510 PRINT "                          SECTION 29.2"
0520 PRINT
0530 PRINT "              (USING ONLY NUMERICALLY ASSIGNED VARIABLES.)"
0540 PRINT
0550 PRINT
0560 LET A$=" "
0570 LET R2=1.5
0580 LET S2=0.2
0590 LET U2=3.7
0600 LET V2=2.0
0610 LET W2=2.557
0620 LET X2=3367
0630 LET Y2=3E-11
0640 LET Z2=.14
0650 LET X3=.35E34
```

```
0660 LET Y3=2E-11
0670 LET Z3=.1
0680 LET Q3=-5.67654
0690 LET R3=-9.89898E13
0700 LET U3=R2*S2*U2*V2*W2
0710 LET V3=U3+Q3
0720 IF ABS(V3)<=1E-5 THEN 740
0730 LET A$="*"
0740 PRINT "    5    "," 5.67654 ",U3,V3;A$
0750 LET A$=" "
0760 LET S3=X2*Y2*Z2*X3*Y3*Z3
0770 LET W3=S3+R3
0780 IF ABS(W3)<=1E8 THEN 800
0790 LET A$="*"
0800 PRINT "    6    "," 9.89898E13 ",S3,W3;A$
0810 PRINT
0820 PRINT
0830 PRINT "**************************************************";
0840 PRINT "**********"
0850 PRINT
0860 PRINT
0870 PRINT "                         SECTION 29.3"
0880 PRINT
0890 PRINT "    (NUMERICAL CONS. AND ASSIGNED NUMERICAL VARS., MIXED.)"
0900 PRINT
0910 PRINT
0920 LET A$=" "
0930 LET X2=1.5E-5
0940 LET Y2=0.8E20
0950 LET Z2=64.3E8
0960 LET X3=2.0E-6
0970 LET Y3=3
0980 LET Z3=6.25
0990 LET Q3=10101
1000 LET R3=1.1
1010 LET U3=X2*.2E8*Y2*.4E-10*Z2*.1E-5*X3
1020 LET V3=U3+(-1.23456E10)
1030 IF ABS(V3)<=1E5 THEN 1050
1040 LET A$="*"
1050 PRINT "    7    "," 1.23456E10 ",U3,V3;A$
1060 LET A$=" "
1070 LET S3=375E10*Y3*1.6E-12*Z3*4E21*Q3*.2E-10*R3
1080 LET W3=S3+(-9.99999E16)
1090 IF ABS(W3)<=1E11 THEN 1110
1100 LET A$="*"
1110 PRINT "    8    "," 9.99999E16 ",S3,W3;A$
1120 PRINT
1130 PRINT
1140 PRINT "**************************************************";
1150 PRINT "**********"
1160 PRINT
1170 PRINT "                                  END TEST."
1180 PRINT
1190 PRINT
1200 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

PROGRAM FILE 29

SECTION 29.0: MULTIPLICATION OF THREE OR MORE FACTORS.

BEGIN TEST.

IF NO ASTERISK FOLLOWS ANY VALUE IN THE ABSOLUTE ERROR COLUMN, TEST PASSED. HOWEVER, IF AN ASTERISK FOLLOWS A VALUE IN THE ABSOLUTE ERROR COLUMN, TEST FAILED BECAUSE SYSTEM WOULD HAVE FAILED THE ERROR BOUND ROUND-OFF OF SIX PLACE ACCURACY.

NUMBER OF TERMS	REQUIRED PRODUCT	PRODUCT OF SYSTEM	ABSOLUTE ERROR
SECTION 29.1			
(USING ONLY NUMERICAL CONSTANTS.)			
3	65432	65432	0
4	9.99999E20	9.99999E+20	0
SECTION 29.2			
(USING ONLY NUMERICALLY ASSIGNED VARIABLES.)			
5	5.67654	5.67654	0
6	9.89898E13	9.89898E+13	0
SECTION 29.3			
(NUMERICAL CONS. AND ASSIGNED NUMERICAL VARS., MIXED.)			

```
 7          1.23456E10     1.23456E+10    0
 8          9.99999E16     9.99999E+16    0

*******************************************************************************

                                END TEST.
```

30.0 HIERARCHY OF OPERATIONS AND PARENTHESES

In this and the succeeding section we verify that for any numerical expression, involutions are performed first, then multiplications and divisions, and finally additions and subtractions (unless parentheses dictate otherwise). Furthermore, in the absence of parentheses, and where it matters mathematically, we determine whether any numerical expression, having operations of the same precedence, is associated to the left. In this section the tests will concentrate on evaluating expression formed from constants. Variables will be introduced in section 31. The reader should be familiar with the operator precedence specifications in section 7.4 of BSR X3.60.

30.1 Expressions with Operators of Equal Priority

The objective of this test is to verify that in numerical expressions, in which there are no parentheses ordering operations, operators of the same precedence will associate to the left and maintain at least six decimal digits of precision. The test uses only numerical constants in the expressions and the operators of equal precedence are used in the following order: (1) operators are all division symbols, (2) operators are all subtraction symbols, (3) operators are all involution symbols, (4) operators are both subtraction and addition symbols, and (5) operators are both division and multiplication.

There are two columns of output. The first column is labeled "Operator(s) of Expression" and the second column is labeled "Evaluation of System". The first column lists the operator(s) that are used in each numeric expression tested, while the second column lists whether the implementation passed or failed according to the expected values of this test.

30.2 Expressions with Operators of Different Priorities and no Parenthesis

The objective of this test is to verify that in numerical expressions in which there are no parentheses to dictate the order of operations, operators of different precedence will be performed in the following sequence: first, involutions, then multiplications and divisions, and finally, additions and subtraction. The test also determines whether the expressions maintain at least six decimal digits of precision. The expressions used have only numerical constants in them. Only numerical expressions, in which there are operators of unequal precedence, in different combinations, for the different numeric expressions, are used.

There are two columns of output. The first column is labeled "Priority of Operator" and the second column is labeled "Evaluation of System". The first column lists the operators and their priorities according to their usage in each numeric expression to be tested, while the second column lists whether the implementation passed or failed.

30.3 Expressions with Operators of Different Priorities and Parentheses

The objective of this test is to verify that numerical expressions, with parentheses to order the operations, will be evaluated properly and maintain at least six decimal digits of precision. The test uses two approaches. First, parentheses are used to alter the evaluation for numeric expressions using operators of equal precedence. Then, parentheses are used to alter the evaluation for numeric expressions using operators of unequal precedence.

There is a two column output of which the first column is labeled "Alteration of Priority by Parentheses", and the second column is labeled "Evaluation of System". The first column lists the classes of numeric expressions which should have been evaluated, while the second column lists whether implementation evaluated each class of expressions successfullly or unsuccessfully.

```
********************
* PROGRAM FILE 30 *
********************
```

```
0010 PRINT "PROGRAM FILE 30"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "      SECTION 30.0: HIERARCHY OF OPERATORS AND PARENTHESES."
0100 PRINT
0110 PRINT
0120 PRINT "     SINCE THIS TEST IS ONLY CONCERNED WITH THE ORDER OF"
0130 PRINT "OPERATIONS, THE SELECTED NUMBERS USED FOR THIS TEST ARE IN"
0140 PRINT "INTEGER FORM ONLY."
0150 PRINT
0160 PRINT
0170 PRINT
0180 PRINT "                          SECTION 30.1"
0190 PRINT
0200 PRINT "     LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERA-"
0210 PRINT "TORS OF EQUAL PRIORITY, USING CONSTANTS ONLY."
0220 PRINT
0230 PRINT
0240 PRINT "*****NOTE: LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH"
0250 PRINT "OPERATORS OF ONLY ADDITION OR MULTIPLICATION DOES NOT  "
0260 PRINT "APPLY, THEREFORE, SUCH EXPRESSIONS ARE NOT TESTED IN"
0270 PRINT "THIS TEST.*****"
0280 PRINT
0290 PRINT
0300 PRINT "                                BEGIN TEST."
0310 PRINT
0320 PRINT TAB(18);"OPERATOR(S)";TAB(40);"EVALUATION"
0330 PRINT TAB(18);"    OF     ";TAB(40);"    OF    "
0340 PRINT TAB(18);"EXPRESSION ";TAB(40);"  SYSTEM  "
0350 PRINT "************************************************";
```

```
0360 PRINT "**********************"
0370 PRINT
0380 PRINT
0390 LET A$="PASSED"
0400 LET A=81/9/3
0410 LET A1=A-3
0420 IF ABS(A1)<=1E-5 THEN 440
0430 LET A$="FAILED"
0440 PRINT TAB(18);"DIVISION";TAB(40);A$
0450 PRINT
0460 LET A$="PASSED"
0470 LET A=23-13-2
0480 LET A1=A-8
0490 IF ABS(A1)<=1E-5 THEN 510
0500 LET A$="FAILED"
0510 PRINT TAB(18);"SUBTRACTION";TAB(40);A$
0520 PRINT
0530 LET A$="PASSED"
0540 LET A=9^3^2
0550 LET A1=A-531441
0560 IF ABS(A1)<=1E0 THEN 580
0570 LET A$="FAILED"
0580 PRINT TAB(18);"EXPONENTIATION";TAB(40);A$
0590 PRINT
0600 LET A$="PASSED"
0610 LET A=23-13+2
0620 LET A1=A-12
0630 IF ABS(A-12)<=1E-4 THEN 650
0640 LET A$="FAILED"
0650 PRINT TAB(18);"SUBTRACTION"
0660 PRINT TAB(18);"    AND     ";TAB(40);A$
0670 PRINT TAB(18);" ADDITION   "
0680 PRINT
0690 LET A$="PASSED"
0700 LET A=81/9*3
0710 LET A1=A-27
0720 IF ABS(A1)<=1E-4 THEN 740
0730 LET A$="FAILED"
0740 PRINT TAB(18);"   DIVISION    "
0750 PRINT TAB(18);"      AND       ";TAB(40);A$
0760 PRINT TAB(18);"MULTIPLICATION"
0770 PRINT
0780 PRINT "                                 END TEST."
0790 PRINT
0800 PRINT
0810 PRINT "                          SECTION 30.2"
0820 PRINT
0830 PRINT "     EVALUATION OF THE PRECEDENCE OF OPERATORS FOR EXPRES-"
0840 PRINT "SIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES IN"
0850 PRINT "THE ABSENCE OF PARENTHESES, USING CONSTANTS ONLY."
0860 PRINT
0870 PRINT
0880 PRINT "                                BEGIN TEST."
0890 PRINT
0900 PRINT TAB(21);"PRIORITY";TAB(51);"EVALUATION"
0910 PRINT TAB(24);"OF";TAB(55);"OF"
0920 PRINT TAB(21);"OPERATOR";TAB(53);"SYSTEM"
```

```
0930 PRINT "**************************************************";
0940 PRINT "**********"
0950 PRINT
0960 PRINT
0970 LET A$="PASSED"
0980 LET A=81+2*3-7+20
0990 LET A1=A-100
1000 IF ABS(A1)<=1E-3 THEN 1020
1010 LET A$="FAILED"
1020 PRINT TAB(18);"MULTIPLICATION"
1030 PRINT TAB(22);"BEFORE";TAB(53);A$
1040 PRINT TAB(11);"ADDITION OR SUBTRACTION"
1050 PRINT
1060 LET A$="PASSED"
1070 LET A=100+72/9-3-6
1080 LET A1=A-99
1090 IF ABS(A1)<=1E-4 THEN 1110
1100 LET A$="FAILED"
1110 PRINT TAB(21);"DIVISION"
1120 PRINT TAB(22);"BEFORE";TAB(53);A$
1130 PRINT TAB(11);"ADDITION OR SUBTRACTION"
1140 PRINT
1150 LET A$="PASSED"
1160 LET F=0
1170 LET A=10+100^2-11
1180 LET A1=A-9999
1190 IF ABS(A1)<=1E-2 THEN 1220
1200 LET F=F+1
1210 PRINT "EXPONENTIATION BEFORE ADDITION OR SUBTRACTION FAILED."
1220 LET A=11*3^2/9
1230 LET A1=A-11
1240 IF ABS(A1)<=1E-4 THEN 1270
1250 LET F=F+1
1260 PRINT "EXPONENTIATION BEFORE MULTIPLICATION OR DIVISIONFAILED."
1270 LET A=-15^2
1280 LET A1=A-(-225)
1290 IF ABS(A1)<=1E-3 THEN 1320
1300 LET F=F+1
1310 PRINT "EXPONENTIATION BEFORE UNARY FAILED."
1320 IF F=0 THEN 1340
1330 GOTO 1370
1340 PRINT TAB(18);"EXPONENTIATION"
1350 PRINT TAB(19);"AS FIRST OF";TAB(53);A$
1360 PRINT TAB(20);"OPERATIONS"
1370 PRINT
1380 PRINT "                                    END TEST."
1390 PRINT
1400 PRINT
1410 PRINT "                           SECTION 30.3"
1420 PRINT
1430 PRINT "     EVALUATION OF THE PRECEDENCE OF OPERATIONS FOR THOSE"
1440 PRINT "EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES"
1450 PRINT "BUT ARE INFLUENCED BY THE USE OF PARENTHESES, USING CON-"
1460 PRINT "STANTS ONLY."
1470 PRINT
1480 PRINT
1490 PRINT "                                  BEGIN TEST."
```

```
1500 PRINT
1510 PRINT TAB(14);"ALTERATION OF PRIORITY";TAB(51);"EVALUATION"
1520 PRINT TAB(24);"BY";TAB(55);"OF"
1530 PRINT TAB(19);"PARENTHESES";TAB(53);"SYSTEM"
1540 PRINT "**************************************************";
1550 PRINT "**********"
1560 PRINT
1570 PRINT
1580 LET F=0
1590 LET A=81/(9/3)
1600 LET A1=A-27
1610 IF ABS(A1)<=1E-4 THEN 1630
1620 LET F=1
1630 LET A=23-(13-2)
1640 LET A1=A-12
1650 IF ABS(A1)<=1E-4 THEN 1670
1660 LET F=1
1670 LET A=4^(3^2)
1680 LET A1=A-262144
1690 IF ABS(A1)<=1E0 THEN 1710
1700 LET F=1
1710 LET A=23-(13+2)
1720 LET A1=A-8
1730 IF ABS(A1)<=1E-5 THEN 1750
1740 LET F=1
1750 LET A=81/(9*3)
1760 LET A1=A-3
1770 IF ABS(A1)<=1E-5 THEN 1790
1780 LET F=1
1790 IF F=0 THEN 1820
1800 LET A$="UNSUCCESSFUL"
1810 GOTO 1830
1820 LET A$="SUCCESSFUL"
1830 PRINT TAB(18);"EXPRESSIONS OF"
1840 PRINT TAB(18);"LEFT TO RIGHT";TAB(51);A$
1850 PRINT TAB(20);"EVALUATION"
1860 PRINT
1870 LET F=0
1880 LET A=(81+2)*(3-7)+20
1890 LET A1=A-(-312)
1900 IF ABS(A1)<=1E-3 THEN 1920
1910 LET F=1
1920 LET A=(100+92)/(9-3)-6
1930 LET A1=A-26
1940 IF ABS(A1)<+1E-4 THEN 1960
1950 LET F=1
1960 LET A=(725+274)^(4-2)+1998
1970 LET A1=A-999999
1980 IF ABS(A1)<=1E0 THEN 2000
1990 LET F=1
2000 LET A=(11*3)^(9/3)/27
2010 LET A1=A-1331
2020 IF ABS(A1)<=1E-2 THEN 2040
2030 LET F=1
2040 LET A=(-15)^2
2050 LET A1=A-225
2060 IF ABS(A1)<=1E-3 THEN 2080
```

```
2070 LET F=1
2080 IF F=0 THEN 2110
2090 LET A$="UNSUCCESSFUL"
2100 GOTO 2120
2110 LET A$="SUCCESSFUL"
2120 PRINT TAB(14);"EXPRESSIONS EVALUATED"
2130 PRINT TAB(18);"BY PRIORITY OF";TAB(51);A$
2140 PRINT TAB(19);"THE OPERATOR"
2150 PRINT
2160 PRINT "                                    END TEST."
2170 PRINT
2180 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 30

SECTION 30.0: HIERACHY OF OPERATORS AND PARENTHESES.

SINCE THIS TEST IS ONLY CONCERNED WITH THE ORDER OF OPERATIONS, THE SELECTED NUMBERS USED FOR THIS TEST ARE IN INTEGER FORM ONLY.

SECTION 30.1

LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERATORS OF EQUAL PRIORITY, USING CONSTANTS ONLY.

*****NOTE: LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERATORS OF ONLY ADDITION OR MULTIPLICATION DOES NOT APPLY, THEREFORE, SUCH EXPRESSIONS ARE NOT TESTED IN THIS TEST.*****

BEGIN TEST.

OPERATOR(S) OF EXPRESSION	EVALUATION OF SYSTEM

****

DIVISION	PASSED
SUBTRACTION	PASSED
EXPONENTIATION	PASSED
SUBTRACTION AND ADDITION	PASSED
DIVISION AND MULTIPLICATION	PASSED

END TEST.

SECTION 30.2

EVALUATION OF THE PRECEDENCE OF OPERATORS FOR EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES IN THE ABSENCE OF PARENTHESES, USING CONSTANTS ONLY.

BEGIN TEST.

PRIORITY OF OPERATOR	EVALUATION OF SYSTEM
MULTIPLICATION BEFORE ADDITION OR SUBTRACTION	PASSED
DIVISION BEFORE ADDITION OR SUBTRACTION	PASSED
EXPONENTIATION AS FIRST OF OPERATIONS	PASSED

END TEST.

SECTION 30.3

EVALUATION OF THE PRECEDENCE OF OPERATIONS FOR THOSE EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES BUT ARE INFLUENCED BY THE USE OF PARENTHESES, USING CONSTANTS ONLY.

BEGIN TEST.

ALTERATION OF PRIORITY BY PARENTHESES	EVALUATION OF SYSTEM
EXPRESSIONS OF LEFT TO RIGHT EVALUATION	SUCCESSFUL
EXPRESSIONS EVALUATED BY PRIORITY OF THE OPERATOR	SUCCESSFUL

END TEST.

31.0 HIERARCHY OF OPERATORS AND PARENTHESES (CONTINUED)

In this section expression will be formed using assigned variables. The test will then consider the two major cases of operations of equal precedence and then of unequal precedence.

31.1 Operators of Equal Priority

The objective of this test is the same as that in test 30.1. This test uses simple numeric variables in its numerical expressions; otherwise, the test is similar in structure and output format to test 30.1.

31.2 Operators of Different Priorities

The objective of this test is the same as the objective stated for test 30.2. This test uses simple numeric variables in its numerical expressions; otherwise, the test is similar in structure and output format to test 30.2.

31.3 Operators of Different Priorities with Parentheses

The objective of this test is the same as that for 30.3. This test uses simple numeric variables in its numerical expressions; otherwise, this test is similar in structure and output format to 30.3.

```
**********
* PROGRAM FILE 31 *
**********
```

```
0010 PRINT "PROGRAM FILE 31"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "                         SECTION 31.1"
0100 PRINT
0110 PRINT "     LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERA-"
0120 PRINT "TORS OF EQUAL PRIORITY, USING ASSIGNED SIMPLE VARIABLES."
0130 PRINT
0140 PRINT
0150 PRINT "*****NOTE: LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH"
0160 PRINT "OPERATORS OF ONLY ADDITION OR MULTIPLICATION DOES NOT NE-"
0170 PRINT "CESSARILY APPLY, THEREFORE, SUCH EXPRESSIONS ARE NOT TESTED"
0180 PRINT "IN THIS TEST.*****"
0190 PRINT
0200 PRINT
```

```
0210 PRINT "                                   BEGIN TEST."
0220 PRINT
0230 PRINT TAB(18);"OPERATOR(S)";TAB(40);"EVALUATION"
0240 PRINT TAB(18);"    OF     ";TAB(40);"    OF    "
0250 PRINT TAB(18);"EXPRESSION ";TAB(40);"  SYSTEM  "
0260 PRINT "************************************************";
0270 PRINT "**********************"
0280 PRINT
0290 PRINT
0300 LET A$="PASSED"
0310 LET A=81
0320 LET B=9
0330 LET C=3
0340 LET D=23
0350 LET E=13
0360 LET F=2
0370 LET X=A/B/C
0380 LET X1=X-3
0390 IF ABS(X1)<=1E-5 THEN 410
0400 LET A$="FAILED"
0410 PRINT TAB(18);"DIVISION";TAB(40);A$
0420 PRINT
0430 LET A$="PASSED"
0440 LET X=D-E-F
0450 LET X1=X-8
0460 IF ABS(X1)<=1E-5 THEN 480
0470 LET A$="FAILED"
0480 PRINT TAB(18);"SUBTRACTION";TAB(40);A$
0490 PRINT
0500 LET A$="PASSED"
0510 LET X=B^C^F
0520 LET X1=X-531441
0530 IF ABS(X1)<=1E0 THEN 550
0540 LET A$="FAILED"
0550 PRINT TAB(18);"EXPONENTIATION";TAB(40);A$
0560 PRINT
0570 LET A$="PASSED"
0580 LET X=D-E+F
0590 LET X1=X-12
0600 IF ABS(X1)<=1E-4 THEN 620
0610 LET A$="FAILED"
0620 PRINT TAB(18);"SUBTRACTION"
0630 PRINT TAB(18);"    AND    ";TAB(40);A$
0640 PRINT TAB(18);" ADDITION  "
0650 PRINT
0660 LET A$="PASSED"
0670 LET X=A/B*C
0680 LET X1=X-27
0690 IF ABS(X1)<=1E-4 THEN 710
0700 LET A$="FAILED"
0710 PRINT TAB(18);"   DIVISION   "
0720 PRINT TAB(18);"     AND      ";TAB(40);A$
0730 PRINT TAB(18);"MULTIPLICATION"
0740 PRINT
0750 PRINT "                                   END TEST."
0760 PRINT
0770 PRINT
```

```
0780 PRINT "                          SECTION 31.2"
0790 PRINT
0800 PRINT "     EVALUATION OF THE PRECEDENCE OF OPERATORS FOR EXPRES-"
0810 PRINT "SIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES IN"
0820 PRINT "THE ABSENCE OF PARENTHESES, USING ASSIGNED SIMPLE VARIA-"
0830 PRINT "BLES."
0840 PRINT
0850 PRINT
0860 PRINT "                                  BEGIN TEST."
0870 PRINT
0880 PRINT TAB(21);"PRIORITY";TAB(51);"EVALUATION"
0890 PRINT TAB(24);"OF";TAB(55);"OF"
0900 PRINT TAB(21);"OPERATOR";TAB(53);"SYSTEM"
0910 PRINT "************************************************";
0920 PRINT "**********************"
0930 PRINT
0940 PRINT
0950 LET A$="PASSED"
0960 LET G=7
0970 LET H=20
0980 LET I=100
0990 LET J=72
1000 LET K=6
1010 LET L=10
1020 LET M=11
1030 LET X=A+F*C-G+H
1040 LET X1=X-100
1050 IF ABS(X1)<=1E-3 THEN 1070
1060 LET A$="FAILED"
1070 PRINT TAB(18);"MULTIPLICATION"
1080 PRINT TAB(22);"BEFORE";TAB(53);A$
1090 PRINT TAB(11);"ADDITION OR SUBTRACTION"
1100 PRINT
1110 LET A$="PASSED"
1120 LET X=I+J/B-C-K
1130 LET X1=X-99
1140 IF ABS(X1)<=1E-4 THEN 1160
1150 LET A$="FAILED"
1160 PRINT TAB(21);"DIVISION"
1170 PRINT TAB(22);"BEFORE";TAB(53);A$
1180 PRINT TAB(11);"ADDITION OR SUBTRACTION"
1190 PRINT
1200 LET A$="PASSED"
1210 LET F1=0
1220 LET X=L+I^F-M
1230 LET X1=X-9999
1240 IF ABS(X1)<=1E-2 THEN 1270
1250 LET F1=1
1260 PRINT "EXPONENTIATION BEFORE ADDITION OR SUBTRACTION FAILED."
1270 LET X=M*C^F/B
1280 LET X1=X-11
1290 IF ABS(X1)<=1E-4 THEN 1320
1300 LET F1=1
1310 PRINT "EXPONENTIATION BEFORE MULTIPLICATION OR DIVISION FAILED."
1320 LET X=-A^F
1330 LET X1=X-(-6561)
1340 IF ABS(X1)<=1E-2 THEN 1370
```

```
1350 LET F1=1
1360 PRINT "EXPONENTIATION BEFORE UNARY FAILED."
1370 IF F1=0 THEN 1390
1380 GOTO 1420
1390 PRINT TAB(18);"EXPONENTIATION"
1400 PRINT TAB(19);"AS FIRST OF";TAB(53);A$
1410 PRINT TAB(20);"OPERATIONS"
1420 PRINT
1430 PRINT "                                    END TEST."
1440 PRINT
1450 PRINT
1460 PRINT "                          SECTION 31.3"
1470 PRINT
1480 PRINT "     EVALUATION OF THE PRECEDENCE OF OPERATIONS FOR THOSE"
1490 PRINT "EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES"
1500 PRINT "BUT ARE INFLUENCED BY THE USE OF PARENTHESES, USING SIM-"
1510 PRINT "PLE VARIABLES ONLY."
1520 PRINT
1530 PRINT
1540 PRINT "                                    BEGIN TEST."
1550 PRINT
1560 PRINT TAB(14);"ALTERATION OF PRIORITY";TAB(51);"EVALUATION"
1570 PRINT TAB(24);"BY";TAB(55);"OF"
1580 PRINT TAB(19);"PARENTHESES";TAB(53);"SYSTEM"
1590 PRINT "*****************************************************";
1600 PRINT "**********************"
1610 PRINT
1620 PRINT
1630 LET F1=0
1640 LET N=4
1650 LET O=92
1660 LET P=725
1670 LET Q=274
1680 LET R=1998
1690 LET S=27
1700 LET X=A/(B/C)
1710 LET X1=X-27
1720 IF ABS(X1)<=1E-4 THEN 1740
1730 LET F1=1
1740 LET X=D-(E-F)
1750 LET X1=X-12
1760 IF ABS(X1)<=1E-4 THEN 1780
1770 LET F1=1
1780 LET X=N^(C^F)
1790 LET X1=X-262144
1800 IF ABS(X1)<=1E0 THEN 1820
1810 LET F1=1
1820 LET X=D-(E+F)
1830 LET X1=X-8
1840 IF ABS(X1)<=1E-5 THEN 1860
1850 LET F1=1
1860 LET X=A/(B*C)
1870 LET X1=X-3
1880 IF ABS(X1)<=1E-5 THEN 1900
1890 LET F1=1
1900 IF F1=0 THEN 1930
1910 LET A$="UNSUCCESSFUL"
```

```
1920 GOTO 1940
1930 LET A$="SUCCESSFUL"
1940 PRINT TAB(18);"EXPRESSIONS OF"
1950 PRINT TAB(18);"LEFT TO RIGHT";TAB(51);A$
1960 PRINT TAB(20);"EVALUATION"
1970 PRINT
1980 LET F1=0
1990 LET X=(A+F)*(C-G)+H
2000 LET X1=X-(-312)
2010 IF ABS(X1)<=1E-3 THEN 2030
2020 LET F1=1
2030 LET X=(I+O)/(B-C)-K
2040 LET X1=X-26
2050 IF ABS(X1)<=1E-4 THEN 2070
2060 LET F1=1
2070 LET X=(P+Q)^(N-F)+R
2080 LET X1=X-999999
2090 IF ABS(X1)<=1E0 THEN 2110
2100 LET F1=1
2110 LET X=(M*C)^(B/C)/S
2120 LET X1=X-1331
2130 IF ABS(X1)<=1E-2 THEN 2150
2140 LET F1=1
2150 LET X=(-A)^F
2160 LET X1=X-6561
2170 IF ABS(X1)<=1E-2 THEN 2190
2180 LET F1=1
2190 IF F1=0 THEN 2220
2200 LET A$="UNSUCCESSFUL"
2210 GOTO 2230
2220 LET A$="SUCCESSFUL"
2230 PRINT TAB(14);"EXPRESSIONS EVALUATED"
2240 PRINT TAB(18);"BY PRIORITY OF";TAB(51);A$
2250 PRINT TAB(19);"THE OPERATOR"
2260 PRINT
2270 PRINT "                                    END TEST."
2280 PRINT
2290 END
```

```
******************
* SAMPLE OUTPUT *
******************
```

```
PROGRAM FILE 31
```

SECTION 31.1

LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERATORS OF EQUAL PRIORITY, USING ASSIGNED SIMPLE VARIABLES.

*****NOTE: LEFT TO RIGHT EVALUATION FOR EXPRESSIONS WITH OPERATORS OF ONLY ADDITION OR MULTIPLICATION DOES NOT NECESSARILY APPLY, THEREFORE, SUCH EXPRESSIONS ARE NOT TESTED IN THIS TEST.*****

BEGIN TEST.

OPERATOR(S) OF EXPRESSION	EVALUATION OF SYSTEM
DIVISION	PASSED
SUBTRACTION	PASSED
EXPONENTIATION	PASSED
SUBTRACTION AND ADDITION	PASSED
DIVISION AND MULTIPLICATION	PASSED

END TEST.

SECTION 31.2

EVALUATION OF THE PRECEDENCE OF OPERATORS FOR EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES IN THE ABSENCE OF PARENTHESES, USING ASSIGNED SIMPLE VARIABLES.

BEGIN TEST.

PRIORITY OF OPERATOR	EVALUATION OF SYSTEM
MULTIPLICATION BEFORE ADDITION OR SUBTRACTION	PASSED
DIVISION BEFORE	PASSED

ADDITION OR SUBTRACTION

EXPONENTIATION AS FIRST OF OPERATIONS	PASSED

END TEST.

SECTION 31.3

EVALUATION OF THE PRECEDENCE OF OPERATIONS FOR THOSE EXPRESSIONS WHICH CONTAIN OPERATORS OF DIFFERENT PRIORITIES BUT ARE INFLUENCED BY THE USE OF PARENTHESES, USING SIMPLE VARIABLES ONLY.

BEGIN TEST.

ALTERATION OF PRIORITY BY PARENTHESES	EVALUATION OF SYSTEM
EXPRESSIONS OF LEFT TO RIGHT EVALUATION	SUCCESSFUL
EXPRESSIONS EVALUATED BY PRIORITY OF THE OPERATOR	SUCCESSFUL

END TEST.

32.0 EVALUATION OF EXPRESSIONS HAVING A VARIETY OF OPERATORS

The expressions used are formed from both numerical constants and simple-numeric-variabless. Each expression is characterized by either the absence of parentheses, the use of non-nested parentheses, or the use of nested parentheses. The objective of this test is to verify that numerical expressions can be constructed from both numerical constants and simple-numeric-variables using a combination of operators, with and without parentheses, and that the system will maintain at least six decimal digits of precision. There is a two column output with the first column labeled "Category of Expression", and the second column labeled "Evaluation of System". The first column lists the classes or categories of numeric expressions that are evaluated, while the second indicates whether the implementation evaluated each expression within tolerance, i.e. whether the system passed or failed.

```
********************
* PROGRAM FILE 32 *
********************
```

```
0010 PRINT "PROGRAM FILE 32"
0060 PRINT
0070 PRINT
0080 PRINT
0090 PRINT "                          SECTION 32.0"
0100 PRINT
0110 PRINT "     EVALUATION OF EXPRESSIONS WHICH HAVE A VARIETY OF"
0120 PRINT "OPERATORS AND ARE OF ONE OF THREE CATEGORIES:"
0130 PRINT
0140 PRINT "      (1) NO PARENTHESES,"
0150 PRINT "      (2) NON-NESTED PARENTHESES, AND"
0160 PRINT "      (3) NESTED PARENTHESES."
0170 PRINT
0180 PRINT "ALSO, THESE EXPRESSIONS ARE FORMED FROM THE USE OF BOTH"
0190 PRINT "NUMERICAL CONSTANTS AND SIMPLE-NUMERIC-VARIABLES."
0210 PRINT
0220 PRINT
0230 PRINT TAB(18);" CATEGORY ";TAB(36);"EVALUATION"
0240 PRINT TAB(18);"    OF    ";TAB(36);"    OF    "
0250 PRINT TAB(18);"EXPRESSION";TAB(36);"  SYSTEM  "
0260 PRINT "************************************************";
0270 PRINT "**********************"
0280 PRINT
0290 PRINT
0300 PRINT "                                BEGIN TEST."
0310 PRINT
0320 PRINT
0330 LET F1=0
```

```
0340 LET X=3
0350 LET Y=2
0360 LET Z=5
0370 LET W=-60
0380 LET M=92
0390 LET A=X+Y*Z-W/5-W/5/6+18
0400 LET A1=A-45
0410 IF ABS(A1)<=1E-4 THEN 430
0420 LET F1=1
0430 LET B=Y^3*4+216/3^Y*2+M-82
0440 LET B1=B-90
0450 IF ABS(B1)<=1E-4 THEN 470
0460 LET F1=1
0470 LET C=Y*X+Y*W-Y+170
0480 LET C1=C-54
0490 IF ABS(C1)<=1E-4 THEN 510
0500 LET F1=1
0510 LET D=W/Y+105/Z*Y^2+3-24
0520 LET D1=D-33
0530 IF ABS(D1)<=1E-4 THEN 550
0540 LET F1=1
0550 LET E=Y*Y+W*Y+167*X+Y-124
0560 LET E1=E-263
0570 IF ABS(E1)<=1E-3 THEN 590
0580 LET F1=1
0590 LET F=Y*Y+Y*Y+Y*Y+Y*Y+Y-15
0600 LET F2=F-3
0610 IF ABS(F2)<=1E-5 THEN 630
0620 LET F1=1
0630 LET G=W+Z+X+Y+Y-X-9
0640 LET G1=G-(-60)
0650 IF ABS(G1)<=1E-4 THEN 670
0660 LET F1=1
0670 LET H=W/Z+X+Y*Y^Y-X+10
0680 LET H1=H-6
0690 IF ABS(H1)<=1E-5 THEN 710
0700 LET F1=1
0710 IF F1<>0 THEN 740
0720 LET A$="PASSED"
0730 GOTO 750
0740 LET A$="FAILED"
0750 LET F1=0
0760 LET A=(X+Y)*(Z-W)/5-W/5/6+18
0770 LET A1=A-85
0780 IF ABS(A1)<=1E-4 THEN 800
0790 LET F1=1
0800 LET B=Y^(3*4)+(216/3)^Y*2+(M-82)
0810 LET B1=B-14474
0820 IF ABS(B1)<=1E-1 THEN 840
0830 LET F1=1
0840 LET C=Y*(X+Y)*(W-Y)+170
0850 LET C1=C-(-450)
0860 IF ABS(C1)<=1E-3 THEN 880
0870 LET F1=1
0880 LET D=W/Y+105/Z*Y^(2+3)-24
0890 LET D1=D-618
0900 IF ABS(D1)<=1E-3 THEN 920
```

```
0910 LET F1=1
0920 LET E=Y*(Y+W)*(Y+167)*X+(Y-124)
0930 LET E1=E-(-58934)
0940 IF ABS(E1)<=1E-1 THEN 960
0950 LET F1=1
0960 LET F=Y*(Y+Y)*Y+(Y*Y+Y)*Y+(Y-15)
0970 LET F2=F-15
0980 IF ABS(F2)<=1E-4 THEN 1000
0990 LET F1=1
1000 LET G=(W+Z+X+Y+Y)-(X-9)
1010 LET G1=G-(-42)
1020 IF ABS(G1)<=1E-4 THEN 1040
1030 LET F1=1
1040 LET H=W/(Z+X)+Y*Y^(Y-X)+10
1050 LET H1=H-3.5
1060 IF ABS(H1)<=1E-4 THEN 1080
1070 LET F1=1
1080 IF F1<>0 THEN 1110
1090 LET B$="PASSED"
1100 GOTO 1120
1110 LET B$="FAILED"
1120 LET F1=0
1130 LET A=Y/(184/(M/(30/(W/(12/X)))))
1140 LET A1=A-(-.5)
1150 IF ABS(A1)<=1E-6 THEN 1170
1160 LET F1=1
1170 LET B=Y^3*4+405/(3^(Y*2))+M-82
1180 LET B1=B-47
1190 IF ABS(B1)<=1E-4 THEN 1210
1200 LET F1=1
1210 LET C=Y*(X+Y*(W-Y))+170
1220 LET C1=C-(-72)
1230 IF ABS(C1)<=1E-4 THEN 1250
1240 LET F1=1
1250 LET D=W/Y+121/(Z*(Y^2+3)-24)
1260 LET D1=D-(-19)
1270 IF ABS(D1)<=1E-4 THEN 1290
1280 LET F1=1
1290 LET E=Y*(Y+W*(Y+167*(X+Y)))-124
1300 LET E1=E-(-100560)
1310 IF ABS(E1)<=1E0 THEN 1330
1320 LET F1=1
1330 LET F=Y*(Y+Y*(Y+Y*(Y+Y*(Y+Y))))-15
1340 LET F2=F-77
1350 IF ABS(F2)<=1E-4 THEN 1370
1360 LET F1=1
1370 LET G=W+(Z+(X+(Y+(Y-(X-9)))))
1380 LET G1=G-(-42)
1390 IF ABS(G1)<=1E-4 THEN 1410
1400 LET F1=1
1410 LET H=W/(Z+4+(Y*(Y^(Y-X))+10))
1420 LET H1=H-(-3)
1430 IF ABS(H1)<=1E-5 THEN 1450
1440 LET F1=1
1450 IF F1<>0 THEN 1480
1460 LET C$="PASSED"
1470 GOTO 1490
```

```
1480 LET C$="FAILED"
1490 PRINT TAB(18);"     NO     "
1500 PRINT TAB(18);"PARENTHESES";TAB(38);A$
1510 PRINT
1520 PRINT TAB(18);"PARENTHESES"
1530 PRINT TAB(18);"    BUT     ";TAB(38);B$
1540 PRINT TAB(18);"NON-NESTED "
1550 PRINT
1560 PRINT TAB(18);"   NESTED   "
1570 PRINT TAB(18);"PARENTHESES";TAB(38);C$
1580 PRINT
1590 PRINT "                                   END TEST."
1600 PRINT
1610 END
```

```
*****************
* SAMPLE OUTPUT *
*****************
```

PROGRAM FILE 32

SECTION 32.0

EVALUATION OF EXPRESSIONS WHICH HAVE A VARIETY OF OPERATORS AND ARE OF ONE OF THREE CATEGORIES:

(1) NO PARENTHESES,
(2) NON-NESTED PARENTHESES, AND
(3) NESTED PARENTHESES.

ALSO, THESE EXPRESSIONS ARE FORMED FROM THE USE OF BOTH NUMERICAL CONSTANTS AND NUMERICALLY ASSIGNED SIMPLE VARIABLES.

```
                  CATEGORY          EVALUATION
                     OF                 OF
                 EXPRESSION          SYSTEM
*************************************************************************
```

BEGIN TEST.

```
                      NO
                  PARENTHESES          PASSED
```

```
PARENTHESES
    BUT                 PASSED
NON-NESTED

   NESTED
PARENTHESES             PASSED

                END TEST.
```

33.0 INSERTION OF SPACES BETWEEN ELEMENTS OF NUMERIC EXPRESSIONS

This test verifies that spaces are allowed between elements of numeric expressions. This test uses several expressions with varying amounts of space between component elements. On output, a message will indicate that spaces are recognized if the test is successful; otherwise the user should expect some form of syntax diagnostic. If no syntax error is given and the system fails to evaluate the expressions properly a message will indicate that spacing within expressions is not allowed even though it is not diagnosed. For the specifications on spacing within programs the reader is referred to section 3 of BSR X3.60.

```
********************
* PROGRAM FILE 33 *
********************
```

```
0010 PRINT "PROGRAM FILE 33"
0035 PRINT
0040 PRINT
0045 PRINT
0050 PRINT "                    SECTION 33.0"
0055 PRINT
0110 PRINT "     INSERTION OF SPACES BETWEEN THE COMPONENT ELEMENTS"
0120 PRINT "OF NUMERIC EXPRESSIONS."
0130 PRINT
0140 PRINT
0150 PRINT "                                BEGIN TEST."
0160 PRINT
0170 LET A=6
0180 LET B=-8
0190 LET C=-6
0200 LET Y=A  ^ 3 +  3* A  ^ 2   * B +  3 *A * B  ^  2+  B  ^  3
0210 LET Z=A     *     (   C      -     5    )    ^           3
0220 IF Y<>-8 THEN 0240
0230 IF Z=-7986 THEN 0270
0240 PRINT "SPACES BETWEEN COMPONENT ELEMENTS OF NUMERIC EXPRESSIONS"
0250 PRINT "ARE NOT ALLOWED BY SYSTEM."
0260 GOTO 0300
0270 PRINT "SPACES BETWEEN COMPONENT ELEMENTS OF NUMERIC EXPRESSIONS"
0280 PRINT "ARE ALLOWED BY SYSTEM."
0290 PRINT
0300 PRINT "                              END TEST."
0310 PRINT
0320 END
```

****
* SAMPLE OUTPUT *
****

PROGRAM FILE 33

SECTION 33.0

INSERTION OF SPACES BETWEEN THE COMPONENT ELEMENTS OF NUMERIC EXPRESSIONS.

BEGIN TEST.

SPACES BETWEEN COMPONENT ELEMENTS OF NUMERIC EXPRESSIONS ARE ALLOWED BY SYSTEM.

END TEST.

NBS-114A (REV. 7-73)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBS IR 78-1420-2	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE

NBS Minimal BASIC Test Programs - Version 1
User's Manual
Volume 2 - General Program Structure, Output, Assignment
Simple Control Structures, Simple Expressions

5. Publication Date January 1978

6. Performing Organization Code

7. AUTHOR(S) David E. Gilsinn, Charles L. Sheppard

8. Performing Organ. Report No.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

10. Project/Task/Work Unit No. 6401121

11. Contract/Grant No.

12. Sponsoring Organization Name and Complete Address (Street, City, State, ZIP)

National Bureau of Standards

13. Type of Report & Period Covered Final

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

16. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)*

This volume is the second of four volumes that comprise the user's guide to the NBS Minimal BASIC Test Programs. The programs test whether a BASIC processor accepts the syntactical forms and produces semantically meaningful results according to the specifications given in BSR X3.60 Proposed American National Standard for Minimal BASIC The object of this volume is to introduce elementary variable, control and expression forms and exercise them by using a large sample of variations of the appropriate statements. There are thirty-three programs in this volume. They cover specifically the test for output, simple assignment of variables, elementary control structures, general program structure, formulating simple expressions, and the hierarchy of operators. The entire set of test programs is available on magnetic tape.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*

BASIC; BASIC standard; BASIC validation; compiler validation; computer programming language; computer standards.

18. AVAILABILITY

☐ Unlimited

☒ For Official Distribution. Do Not Release to NTIS

☐ Order From Sup. of Doc., U.S. Government Printing Office Washington, D.C. 20402, SD Cat. No. C13

☐ Order From National Technical Information Service (NTIS) Springfield, Virginia 22151

19. SECURITY CLASS (THIS REPORT) UNCLASSIFIED

21. NO. OF PAGES 200

20. SECURITY CLASS (THIS PAGE) UNCLASSIFIED

22. Price

USCOMM-DC 29042-P74