IBM SYSTEM/360
ASSEMBLY LANGUAGE
INTERVAL ARITHMETIC SOFTWARE

(NASA-TH-X-65971) IBM SYSTEM/360 ASSEMBLY
LANGUAGE INTERVAL ARITHMETIC SOFTWARE E.J.
Phillips (NASA) Apr. 1972 20 p CSCL 09B

APRIL 1972

GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U.S. Department of Commerce
Springfield, VA 22151
IBM SYSTEM/360
ASSEMBLY LANGUAGE
INTERVAL ARITHMETIC SOFTWARE

E. J. Phillips
Computing and Software, Inc.
Data Systems Division

April 1972

Goddard Space Flight Center
Greenbelt, Maryland
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>v</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>INTERVAL ARITHMETIC PACKAGE</td>
<td>2</td>
</tr>
<tr>
<td>I Language</td>
<td>2</td>
</tr>
<tr>
<td>II Purpose</td>
<td>2</td>
</tr>
<tr>
<td>III Method</td>
<td>2</td>
</tr>
<tr>
<td>IV Calling Sequence</td>
<td>2</td>
</tr>
<tr>
<td>V Mathematics Used in Package</td>
<td>7</td>
</tr>
<tr>
<td>VI Restrictions</td>
<td>9</td>
</tr>
<tr>
<td>VII Appendix (Sample problem and output)</td>
<td>10</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>16</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENT

The author wishes to gratefully acknowledge the assistance of Dr. Ronald Tost and other members of the Gesellschaft Für Mathematik Und Datenverarbeitung, Bonn, Germany, who prepared and helped implement the software package on the IBM 360 system.
INTRODUCTION

This document describes some computer software designed to perform interval arithmetic. An interval is defined as the set of all real numbers between two given numbers including or excluding one or both endpoints. Interval arithmetic consists of the various elementary arithmetic operations defined on the set of all intervals, such as interval addition, subtraction, union, etc. The theoretical aspects of interval arithmetic are described in reference (2).

One of the main applications of interval arithmetic is in the area of error analysis of computer calculations. For example, it has been used successfully to compute bounds on sounding errors in the solution of linear algebraic systems\(^{(2)}\), error bounds in numerical solutions of ordinary differential equations\(^{(1)}\)\(^{(3)}\) as well as integral equations and boundary value problems. The described software should enable users to implement algorithms of the type described in these references efficiently on the IBM 360 system.

The subroutines in this package were supplied to Goddard Space Flight Center by Dr. Ronald Tost, Institute für Numerische Datenverarbeitung under an exchange agreement with Dr. C. E. Velez, GSFC Code 552. Although the subroutines were coded in IBM 360 assembly language, they were designed so that calls may be made from FORTRAN programs. The sample problem in Section VII and all of the examples in Sections I to VI are given in FORTRAN to illustrate how this package may be used in FORTRAN programs. The subroutines were tested under varying conditions by Computing and Software, Inc., personnel.

The following pages explain in detail the arithmetic operations performed by this package.
INTERVAL ARITHMETIC PACKAGE

I. Language

Assembler Language Code for IBM/360.

II. Purpose

This package is designed to perform arithmetic operations using floating point (64 bits) interval arithmetic.

III. Method

The interval members are defined as COMPLEX * 16, by the user. The basic arithmetic operations are performed along with some basic set theory applications. The arithmetic functions are: addition, subtraction, multiplication, division, exponentiation. The absolute value of an interval and a negative interval can be created. Real and integer numbers can be converted into interval numbers. The intersection and union between two sets of interval numbers can be found. An illegal interval can be tested for. There are tests to determine if zero is contained in (ε) an interval or if one interval is contained in another. The sum of two sets of interval numbers can be found by means of a scalar product. The above operations are accomplished by the following functions:

DLI, DRE, ST$, DT$, ADD, SUB, MUL, DIV, POT, MIN, ABS, SCH, VER, IZER, ICOR, ENT, SUM, INP.

IV. Calling Sequence

The interval numbers and functions are defined as COMPLEX * 16.

In the following set: A is an INTERVAL (AL, AR)
B is an INTERVAL (BL, BR)
C is an INTERVAL (CL, CR)
D: REAL * 8
N: INTEGER * 4

Functions:

1. D = DLI(A)  D: = AL  Left Corner of A
   Ex.  A = (2.5, 4.0)
        D = DLI(A) = 2.5
2. D = DRE(A)  D: = AR  Right Corner of A
   Ex. A = (2.5, 4.0)
   D = DRE(A) = 4.0

3. A = $IT$(N)  A: = (N, N)  Produce an interval from an integer
   Ex. N = 5
   A = $IT$(N) = (5.0, 5.0)

4. A = $DT$(D)  A: = (D, D)  Produce an interval from a real number
   Ex. D = 2.52
   A = $DT$(D) = (2.52, 2.52)
   A = $DT$ (.5DO) = (0.5D0, 0.5D0)

5. A = $ADD (B, C)  A: = B + C  Addition
   Ex. B = (3.0, 4.0)
       C = (2.0, 2.0)
       A = $ADD (B, C) = (3.0 + 2.0, 4.0 + 2.0)
           = (5.0, 6.0)
       B = (-1.5, -0.5)
       C = (2.5, 4.5)
       A = $ADD (B, C) = (-1.5 + 2.5, -0.5 + 4.5)
           = (1.0, 4.0)

6. A = $SUB (B, C)  A: = B - C  Subtraction
   Ex. B = (3.0, 4.0)
       C = (2.0, 2.0)
       A = $SUB (B, C) = (3.0 - 2.0, 4.0 - 2.0)
           = (1.0, 2.0)
       B = (3.0, 4.0)
       C = (-1.5, -2.5)
       A = $SUB (B, C) = (3.0 - (-1.5), 4.0 - (-2.5))
           = (4.5, 6.5)

7. A = $MUL (B, C)  A: = B * C  Multiplication
   Ex. B = (3.0, 4.0)
       C = (2.0, 2.5)
       A = $MUL (B, C) = (MIN (6.0, 7.5, 8.0, 10.0), MAX (6.0, 7.5, 8.0, 10.0))
           = (6.0, 10.0)

8. A = $DIV (B, C)  A: = B/C  Division
   Ex. B = (3.0, 4.0)  if 0 ∉ (B, C)
       C = (1.0, 2.0)
       A = $DIV (B, C) = (3.0, 4.0) * (0.5, 1.0)
           = (MIN (1.5, 3.0, 2.0, 4.0), MAX (1.5, 3.0, 2.0, 4.0))
           = (1.5, 4.0)
9. \( A = \$\text{DOT}(B, N) \quad A = B^N \quad \text{Exponentiation} \)

Ex. \( B = (3.0, 4.0) \)
\( N = 2 \)
\( A = \$\text{DOT}(B, N) = (9.0, 16.0) \)

10. \( A = \$\text{MIN}(B) \quad A = -B \quad \text{Negative Interval} \)

\( B = (3.0, 4.0) \)
\( A = \$\text{MIN}(B) = (-4.0, -3.0) \)

11. \( A = \$\text{ABS}(B) \quad A = B \text{ if } BL > 0 \quad \text{Absolute Value} \)

\( A = -B \text{ if } BR < 0 \text{ else} \)
\( A = (0, \text{MAX}(BL, BR)) \)

Examples:

(1) \( B = (4.0, 4.25) \)
\( A = \$\text{ABS}(B) = (4.0, 4.25) \)

(2) \( B = (-2.0, -1.0) \)
\( A = \$\text{ABS}(B) = (1.0, 2.0) \)

(3) \( B = (-1.5, 4.0) \)
\( A = \$\text{ABS}(B) = (0, \text{MAX}(-1.5, 4.0)) \)
\( = (0, 4.0) \)

12. \( A = \$\text{SCH}(B, C) \quad A = (\text{MAX}(CL, BL), \text{MIN}(CR, BR)) \quad \text{Intersection} \)

Ex. \( B = (4.0, 5.0) \quad \$\text{SCH} \text{ can produce illegal intervals. This can be} \)
\( C = (-1.5, 2.0) \quad \text{proved with ICOR.} \)
\( A = \$\text{SCH}(B, C) = (\text{MAX}(4.0, -1.5), \text{MIN}(5.0, 2.0)) \)
\( = (4.0, 2.0) \quad (-\text{illegal interval}) \)

\( B = (1.0, 2.0) \)
\( C = (-1.5, 5.0) \)
\( A = \$\text{SCH}(B, C) = (1.0, 2.0) \)

13. \( A = \$\text{VER}(B, C) \quad A = (\text{MIN}(CL, BL), \text{MAX}(CR, BR)) \quad \text{Union} \)

\( B = (1.5, 2.5) \)
\( C = (1.0, 5.0) \)
\( A = \$\text{VER}(B, C) = (\text{MIN}(1.0, 1.5), \text{MAX}(5.0, 2.5)) \)
\( = (1.0, 5.0) \)

14. \( \text{IF (IZER (A))} \quad ^\wedge, ^\wedge, ^\wedge \quad \text{Test to see if zero (0) is contained in } A \)

If \( \text{IZER(A)} \leq 0 \quad \text{GO TO}^\wedge \)
If \( \text{IZER(A)} = 0 \quad \text{GO TO}^{^\wedge} \)
If \( \text{IZER(A)} > 0 \quad \text{GO TO}^{^\wedge\wedge} \)
Test to see:

\[ ^{\wedge} \text{ if } AR < 0 \text{ (if } AR < 0 \text{, } 0 \not\in A) \]
\[ ^{\wedge\wedge} \text{ if } AL > 0 \text{ (if } AL > 0 \text{, } 0 \not\in A) \]
\[ ^{\wedge\wedge\wedge} \text{ else } 0 \in A \text{ (zero contained in } A) \text{ null interval} \]

15. IF (ICOR(A)) \[ ^{\wedge} , ^{\wedge\wedge} , ^{\wedge\wedge\wedge} \] Interval Test
   IF ICOR(A) < 0 GO TO \[ ^{\wedge} \]
   IF ICOR(A) = 0 GO TO \[ ^{\wedge\wedge} \]
   IF ICOR(A) > 0 GO TO \[ ^{\wedge\wedge\wedge} \]

Test to see:

\[ ^{\wedge} \text{ if } AL > AR \text{ (illegal interval)} \]
\[ ^{\wedge\wedge} \text{ if } AL = AR \]
\[ ^{\wedge\wedge\wedge} \text{ if } AL < AR \]

16. $\text{ENT} - \text{Test on being contained in}$
   CALL $\text{ENTC} A, B, ^{\wedge1}, ^{\wedge2}, ^{\wedge3}, ^{\wedge4}$
   \[ ^{\wedge1} \text{ AL > BL and AR < BR (A is contained in } B) \]
   \[ ^{\wedge2} \text{ AL = BL and AR = BR (A and } B \text{ are equal)} \]
   \[ ^{\wedge3} \text{ AL < BL and AR > BR (B is contained in } A) \]
   \[ ^{\wedge4} \text{ A and } B \text{ are not fully contained in one another or are not contained in each other at all)} \]

17. $\text{SUM} - \text{Build a sum with a scalar product.}$
   CALL $\text{SUM(A, I, J, K)}$ where $I = J, K$

$\text{SUM}$ store $A, I, J, K$ and sets up a loop made to perform addition.
Along with the call to $\text{SUM}$ a multiplication statement has to follow immediately.

**Example:**

\[
\text{CALL } \text{SUM (A, I, J, K)} \quad A = (0.0, 0.0) \\
D = \text{MUL(B, C)} \\
\quad B(1) = (1.0, 2.0) \quad C(1) = (2.0, 3.0) \\
\quad B(2) = (3.0, 4.0) \quad C(2) = (4.0, 5.0) \\
\quad B(3) = (5.0, 6.0) \quad C(3) = (6.0, 7.0) \\
\]

**Example:**

\[
\text{CALL } \text{SUM (A, I, J, K)} \quad J = 1, K = 3 \\
D = \text{MUL(B(I), C(I))} \\
\quad \text{Correspondent} \\
\quad \text{AKKU} = A \\
\quad \text{DO 1 I = 1, 3} \\
\quad D = (0.0, 0.0) + (2.0, 6.0) \\
\quad = (2.0, 6.0) + (12.0, 20.0) \\
\quad = (14.0, 26.0) + (30.0, 42.0) \\
\quad = (44.0, 68.0) \\
\]

1. AKKU = AKKU + $MUL (B(I), C(I))
   D = AKKU

D, AKKU, and A are intervals; B, C are interval arrays. The sequence of $SUM and $MUL cannot contain interval functions.

18. $INP A = $INP ('<INPUT>') conversion routine

   <INPUT> is an interval number, that is produced from an alphanumeric field of input values surrounded by quotation marks. The routine creates an interval from the set of numbers in the alphanumeric field by choosing the smallest number for the left endpoint and the largest number for the right endpoint. The interval number is greater than or equal to 1 (≥ 1) in absolute value. The real or integer numbers are separated by commas but can contain no blanks. The beginning and end of the interval number must contain at least one blank. The left side of the output is the minimum of the real or integer numbers. The right side of the output contains the maximum of the real or integer numbers.

Example:

   A = $INP ('1.3, 1.56') → A = (1.3, 1.56)
   A = $INP ('-2., 4.') → A = (-2., 4.)
   A = $INP ('10') → No answer: a blank doesn't precede "1" on left side
   A = $INP ('1E45, 23, 2.5') → A = (2.5, 1E45)

   The arithmetic functions can produce exponent overflow and divide check interrupts. Certain interval functions can produce illegal intervals and only two of the subroutines in this package can determine if the output is illegal, IZER and ICOR.

   In all the arithmetic interval functions the input is tested to determine if it is illegal. No error messages are generated just a return to the calling program.

DEFINITION OF <INPUT>:

<INPUT> := <INTER>^<REAL>
<INTER> := <REAL>
   := <INTER>, <REAL>
REAL := <FLOAT>
   := <FLOAT> <EX> <INTEG>
V. MATHEMATICS USED IN PACKAGE

If * is one of the symbols +, -, ·, /, the arithmetic operations of intervals are defined by:

\[ [a, b] \times [c, d] = \{ x \times y \mid a \leq x \leq b, c \leq y \leq d \} \]

\[ [a, b] + [c, d] = [a + c, b + d] \]

\[ [a, b] - [c, d] = [a - d, b - c] \]

\[ [a, b] \cdot [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)] \]

and if \( d \neq 0 \) then

\[ [a, b] / [c, d] = [a, b] \cdot \left[ \frac{1}{d}, \frac{1}{c} \right] \]

Normalized floating point interval arithmetic is used in this package. Under IBM O/S 360 a floating point number is represented as

<table>
<thead>
<tr>
<th>bit</th>
<th>0</th>
<th>1</th>
<th>-</th>
<th>7</th>
<th>8</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>sign</td>
<td>characteristic exponent</td>
<td>fraction (binary)</td>
<td>mantissa</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

where bit = 0 sign of number (0-positive, 1 negative)
$1 - 7$ = the exponent indicating the power of 16 by which the fraction is multiplied. A zero in bit 1 indicates negative exponent, 1 indicates positive exponent. It is coded in excess 64-notation (decimal). That is, subtract decimal equivalent of characteristic by 64 to obtain the actual characteristic.

\[ \text{e.g. } 0 \left| 1000000 \right| = 64 \]

\[ 64 - 64 = 0 = 16 \]

$8 - 63$ = fraction or mantissa. The binary points of the fraction is just before bit position 8. That is to say that a 1 in bit position 8 represents $2^{-1}$, a 1 in bit position 9 represents $2^{-2}$, etc.

The magnitudes of a floating point number is $16^{-65} \times (5.4 \times 10^{-79})$

\[ (1 - 16^{-14}) \cdot 16^{63} (7.2 \times 10^{75}) \]

Example:

\[
\begin{array}{cccccccccc}
\text{Sign} & \text{Characteristic} \\
\hline
(-1) & 65 \\
& 65 - 64 = 1 \\
& -1 \times 16^{1} \times (2^{-2} + 2^{-4} + 2^{-6}) = -2^{4} (2^{-2} + 2^{-4} + 2^{-6}) \\
& = - (2^{2} + 2^{0} + 2^{-2}) \\
& = - (4 + 1 + 0.25) = -5.25 \\
\end{array}
\]

Normalized floating point numbers have a non-zero high order hexadecimal fraction digit. One or more high order hexadecimal fraction digits which are zero makes the number unnormalized. Normalization consists of shifting the fraction left until the high order hexadecimal digit is non-zero and reducing the characteristic by the number of hexadecimal digits shifted. For multiplication and division the operands are normalized before the operations are performed. For the remaining operands (+, -) the result is normalized.

A = 436D1000 00000000 = normalized number

B = 4306D1000 00000000 = unnormalized

= 426D1000 000000 after normalization.
VI. RESTRICTIONS

1. Only basic arithmetic operations and basic set theory applications are performed. No trigonometric operations are performed.

2. Uses long (64 bits) floating point interval arithmetic.

3. The arithmetic performed can produce illegal intervals.

4. The arithmetic performed can cause arithmetic overflow and divide check interrupts.
APPENDIX

LEVEL 20.1 (AUG 71)  OS/360 FORTRAN H

DATE 72.108/09.51.35

COMPILER OPTIONS - NAME= MAIN,OPT=01,LINECNT=58,SIZE=000K,
SOURCE=EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,TD,XREF

<table>
<thead>
<tr>
<th>ISN 0002</th>
<th>IMPLICIT COMPLEX<em>16(A-H,S),REAL</em>8(D-Z),INTEGER(I-N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISN 0003</td>
<td>DIMENSION C(20),D(20),E(20)</td>
</tr>
<tr>
<td>ISN 0004</td>
<td>WRITE(6,101)</td>
</tr>
<tr>
<td>ISN 0005</td>
<td>I=1</td>
</tr>
<tr>
<td>ISN 0006</td>
<td>P=0.5DO</td>
</tr>
<tr>
<td>ISN 0007</td>
<td>A=(2.0DD, 5.0DD)</td>
</tr>
<tr>
<td>ISN 0008</td>
<td>B=(1.0DD, 8.0DD)</td>
</tr>
<tr>
<td>ISN 0009</td>
<td>N=3</td>
</tr>
<tr>
<td>ISN 0010</td>
<td>V=DLI(A)</td>
</tr>
<tr>
<td>ISN 0011</td>
<td>S=DLI(B)</td>
</tr>
<tr>
<td>ISN 0012</td>
<td></td>
</tr>
<tr>
<td>ISN 0013</td>
<td>U=DLI(B)</td>
</tr>
<tr>
<td>ISN 0014</td>
<td>WRITE(6,203) V,S,T,U,N,P</td>
</tr>
<tr>
<td>ISN 0015</td>
<td>WRITE(6,222)</td>
</tr>
<tr>
<td>ISN 0016</td>
<td>D(1)=ITS(N)</td>
</tr>
<tr>
<td>ISN 0017</td>
<td>WRITE(6,108)</td>
</tr>
<tr>
<td>ISN 0018</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0019</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0020</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0021</td>
<td>I=I+1</td>
</tr>
<tr>
<td>ISN 0022</td>
<td>D(2)=SDDT8(P)</td>
</tr>
<tr>
<td>ISN 0023</td>
<td>WRITE(6,109)</td>
</tr>
<tr>
<td>ISN 0024</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0025</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0026</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0027</td>
<td>I=I+1</td>
</tr>
<tr>
<td>ISN 0028</td>
<td>D(3)=SADD(A,B)</td>
</tr>
<tr>
<td>ISN 0029</td>
<td>WRITE(6,102)</td>
</tr>
<tr>
<td>ISN 0030</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0031</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0032</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0033</td>
<td>I=I+1</td>
</tr>
<tr>
<td>ISN 0034</td>
<td>D(4)=SUN(A,B)</td>
</tr>
<tr>
<td>ISN 0035</td>
<td>WRITE(6,103)</td>
</tr>
<tr>
<td>ISN 0036</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0037</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0038</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0039</td>
<td>I=I+1</td>
</tr>
<tr>
<td>ISN 0040</td>
<td>D(5)=SMUL(A,B)</td>
</tr>
<tr>
<td>ISN 0041</td>
<td>WRITE(6,104)</td>
</tr>
<tr>
<td>ISN 0042</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0043</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0044</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0045</td>
<td>I=I+1</td>
</tr>
<tr>
<td>ISN 0046</td>
<td>D(6)=SDIV(A,B)</td>
</tr>
<tr>
<td>ISN 0047</td>
<td>WRITE(6,105)</td>
</tr>
<tr>
<td>ISN 0048</td>
<td>R=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0049</td>
<td>S=DLI(D(1))</td>
</tr>
<tr>
<td>ISN 0050</td>
<td>WRITE(6,202) I,R,S</td>
</tr>
<tr>
<td>ISN 0051</td>
<td>WRITE(6,223)</td>
</tr>
<tr>
<td>ISN 0052</td>
<td>I=1</td>
</tr>
<tr>
<td>ISN 0053</td>
<td>C(1)=SP0T(A,N)</td>
</tr>
<tr>
<td>ISN 0054</td>
<td>WRITE(6,110)</td>
</tr>
<tr>
<td>ISN 0055</td>
<td>R=DLI(C(I))</td>
</tr>
</tbody>
</table>
ISN 0056   S=DRF(C(I))
ISN 0057   WRITE(6,2002) I,R,S
ISN 0058   I=I+1
ISN 0059   C(2)=SPTO(B,N)
ISN 0060   WRITE(6,111)
ISN 0061   R=DLI(C(I))
ISN 0062   S=DRF(C(I))
ISN 0063   WRITE(6,2002) I,R,S
ISN 0064   I=I+1
ISN 0065   C(3)=SSTOREA(I)
ISN 0066   WRITE(6,112)
ISN 0067   R=DLI(C(I))
ISN 0068   S=DRF(C(I))
ISN 0069   WRITE(6,2002) I,R,S
ISN 0070   I=I+1
ISN 0071   C(4)=SSTOREA(I)
ISN 0072   WRITE(6,113)
ISN 0073   R=DLI(C(I))
ISN 0074   S=DRF(C(I))
ISN 0075   WRITE(6,2002) I,R,S
ISN 0076   I=I+1
ISN 0077   C(5)=SABS(A)
ISN 0078   WRITE(6,114)
ISN 0079   R=DLI(C(I))
ISN 0080   S=DRF(C(I))
ISN 0081   WRITE(6,2002) I,R,S
ISN 0082   I=I+1
ISN 0083   C(6)=SABS(B)
ISN 0084   WRITE(6,115)
ISN 0085   R=DLI(C(I))
ISN 0086   S=DRF(C(I))
ISN 0087   WRITE(6,2002) I,R,S
ISN 0088   I=I+1
ISN 0089   C(7)=SCH(A,B)
ISN 0090   WRITE(6,107)
ISN 0091   R=DLI(C(I))
ISN 0092   S=DRF(C(I))
ISN 0093   WRITE(6,2002) I,R,S
ISN 0094   IF(1COR((C(7))=10,20,20
ISN 0095   10 IF(R.GT.S) WRITE(6,190) R,S
ISN 0097   I=I+1
ISN 0098   20 C(8)=SVER(A,B)
ISN 0099   WRITE(6,106)
ISN 0100   R=DLI(C(I))
ISN 0101   S=DRF(C(I))
ISN 0102   WRITE(6,2002) I,R,S
ISN 0103   IF(IZER(I)) 28,30,29
ISN 0104   28 IF(S.LT.0) GO TO 35
ISN 0106   29 IF(R.GT.0) GO TO 35
ISN 0108   30 WRITE(6,210)
    C.TEST ON_BEING_CONTAINED_IN
ISN 0109   35 WRITE(6,217)
ISN 0110   CALL SINC(A,0,636,637,638,639)
ISN 0111   36 WRITE(6,211)
ISN 0112   GO TO 40
ISN 0113   37 WRITE(6,212)
ISN 0114 38 GO TO 40
ISN 0115 39 WRITE(6,213)
ISN 0116 39 GO TO 40
ISN 0117 39 WRITE(6,214)
ISN 0118 40 R=0.0DO,0.0DO)
ISN 0119 WRITE(6,101)
ISN 0120 1=L[I(A)
ISN 0121 1=NRE(A)
ISN 0122 WRITE(6,203) V,S,T,U,N,P
ISN 0123 WRITE(6,217)
ISN 0124 CALL SENTS(A),E41,E42,E43,E44)
ISN 0125 41 WRITE(6,211)
ISN 0126 Go TO 44
ISN 0127 42 WRITE(6,212)
ISN 0128 Go TO 45
ISN 0129 43 WRITE(6,213)
ISN 0130 Go TO 45
ISN 0131 44 WRITE(6,214)
ISN 0132 45. CONTINUE
ISN 0133 CALL $SUM(B,I,1,5)
ISN 0134 F=$MUL(C(I),D(I))
ISN 0135 R=DLIF
ISN 0136 S=DRE(F)
ISN 0137 WRITE(6,218)
ISN 0138 WRITE(6,215),R,S.
ISN 0139 "CONVERSION ROUTINE"
ISN 0140 C(1)=SINP( 1,3,1.56 )
ISN 0141 C(2)=SINP( 1,-2,4 )
ISN 0142 C(3)=SINP(1)
ISN 0143 C(4)=SINP(1,1E45,23,2.5 )
ISN 0144 C(5)=SIT$5(5.500)
ISN 0145 WRITE(6,216)
ISN 0146 DO 9 I=1,6
ISN 0147 R=DLI(C(I))
ISN 0148 S=DRE(C(I))
ISN 0149 9 WRITE(6,2002),I,R,S
ISN 0150 2002 FORMAT(1X,15,'LEFT= ',D24.16,' RIGHT= ',D24.16)
ISN 0151 101 FORMAT(1INTERVAL ARITHMETIC PACKAGE')
ISN 0152 102 FORMAT('OA+B')
ISN 0153 103 FORMAT('OA-A')
ISN 0154 104 FORMAT('OA=B')
ISN 0155 105 FORMAT('OA/B')
ISN 0156 106 FORMAT('OA UNION B')
ISN 0157 107 FORMAT('OA INTERSECT B')
ISN 0158 108 FORMAT('OSIT$(N)')
ISN 0159 109 FORMAT('OSIT$(P)')
ISN 0160 110 FORMAT('OA==N')
ISN 0161 111 FORMAT('0R2N')
ISN 0162 112 FORMAT('OMIN(A')
ISN 0163 113 FORMAT('OSMIN(R')
ISN 0164 114 FORMAT('0SABS(A')
ISN 0165 115 FORMAT('0SABS(R')
ISN 0166 190 FORMAT(' THIS IS AN ILLEGAL INTERVAL ('D24.16,'1',D24.16,')')
ISN 0167 203 FORMAT(1X,'A= ',D24.16,'1',D24.16,1X,'B= ',D24.16,'1',D24.16,1X,'1X','15/1X','P= ',D24.16)
<table>
<thead>
<tr>
<th>ISN 0168</th>
<th>210</th>
<th>FORMAT(' ZERO IS CONTAINED IN THE FOLLOWING INTERVAL (' ,G24.16, ,')')</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISN 0169</td>
<td>211</td>
<td>FORMAT(' A IS CONTAINED IN B')</td>
</tr>
<tr>
<td>ISN 0170</td>
<td>212</td>
<td>FORMAT(' A AND B ARE EQUAL')</td>
</tr>
<tr>
<td>ISN 0171</td>
<td>213</td>
<td>FORMAT(' B IS CONTAINED IN A')</td>
</tr>
<tr>
<td>ISN 0172</td>
<td>214</td>
<td>FORMAT(' A AND B ARE NOT FULLY CONTAINED IN EACH OTHER')</td>
</tr>
<tr>
<td>ISN 0173</td>
<td>215</td>
<td>FORMAT(' LEFT= ',D24.16,2X,'RIGHT= ',D24.16)</td>
</tr>
<tr>
<td>ISN 0174</td>
<td>216</td>
<td>FORMAT('O$INP CONVERSION ROUTINE')</td>
</tr>
<tr>
<td>ISN 0175</td>
<td>217</td>
<td>FORMAT('O$ENT-TEST ON BEING CONTAINED IN')</td>
</tr>
<tr>
<td>ISN 0176</td>
<td>218</td>
<td>FORMAT('O$UM PRODUCED WITH SCALAR PRODUCT')</td>
</tr>
<tr>
<td>ISN 0177</td>
<td>222</td>
<td>FORMAT('OVALUES FOR ARRAY D(I)')</td>
</tr>
<tr>
<td>ISN 0178</td>
<td>223</td>
<td>FORMAT('OVALUES FOR ARRAY C(I)')</td>
</tr>
<tr>
<td>ISN 0179</td>
<td>STOP</td>
<td></td>
</tr>
<tr>
<td>ISN 0180</td>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>
INTERVAL ARITHMETIC PACKAGE

A = 0.2000000000000000D 01, 0.8000000000000000D 01
B = 0.0

N = 3
P = 0.5000000000000000D 00

SENT-TEST ON BEING CONTAINED IN
A AND B ARE NOT FULLY CONTAINED IN EACH OTHER

SUM PRODUCED WITH SCALAR PRODUCT
LEFT = -0.6850000000000000D 02
RIGHT = 0.8730000000000000D 03

$INP CONVERSION ROUTINE

1LEFT = 0.1300000000000000D 01
1RIGHT = 0.1560000000000000D 01
2LEFT = -0.2000000000000000D 01
2RIGHT = 0.4000000000000000D 01
3LEFT = -0.2000000000000000D 01
3RIGHT = 0.4000000000000000D 01
4LEFT = 0.2500000000000000D 01
4RIGHT = 0.1000000000000000D 46
5LEFT = 0.5500000000000000D 01
5RIGHT = 0.5500000000000000D 01
6LEFT = 0.9000000000000000D 01
6RIGHT = 0.9000000000000000D 01
INTERVAL ARITHMETIC PACKAGE

A = 0.2000000000000000 01, 0.5000000000000000 01
B = 0.1000000000000000 01, 0.8000000000000000 01
N = 3
P = 0.5000000000000000 00

VALUES FOR ARRAY D(I)

SITS(N)
1LEFT= 0.3000000000000000 01 RIGHT= 0.3000000000000000 01
2LEFT= 0.5000000000000000 00 RIGHT= 0.5000000000000000 00

A+B
3LEFT= 0.3000000000000000 01 RIGHT= 0.1300000000000000 02

A-R
4LEFT= -0.6000000000000000 00 RIGHT= 0.4000000000000000 01

A*B
5LEFT= 0.2000000000000000 00 RIGHT= 0.4000000000000000 02

A/B
6LEFT= 0.2500000000000000 00 RIGHT= 0.5000000000000000 01

VALUES FOR ARRAY C(I)

A**N
1LEFT= 0.8000000000000000 00 RIGHT= 0.1250000000000000 03

B**N
2LEFT= 0.1000000000000000 00 RIGHT= 0.5120000000000000 03

$MIN(A)
3LEFT= -0.5000000000000000 00 RIGHT= -0.2000000000000000 01

$MIN(B)
4LEFT= -0.8000000000000000 00 RIGHT= -0.1000000000000000 01

$ARS(A)
5LEFT= 0.2000000000000000 00 RIGHT= 0.5000000000000000 01

$ARS(B)
6LEFT= 0.1000000000000000 00 RIGHT= 0.8000000000000000 01

A INTERSECT B
7LEFT= 0.2000000000000000 00 RIGHT= 0.5000000000000000 01

A UNION B
8LEFT= 0.1000000000000000 00 RIGHT= 0.8000000000000000 01

$ENT-TEST ON BEING CONTAINED IN
A IS CONTAINED IN B
References

