Printed at the Mathematical Centre, 413 Kruislaan, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

MC SYLLABUS 47.7

NUMAL
NUMERICAL PROCEDURES IN ALGOL 60

VOLUME 6, SPECIAL FUNCTIONS AND CONSTANTS
VOLUME 7, INTERPOLATION AND APPROXIMATION

P.W. HEMKER (ed.)

MATHEMATISCH CENTRUM AMSTERDAM 1981

1980 Mathematics subject classification: 65XX04, 68B99

ISBN 90 6196 217 X

AUTHOR: D.T.WINTER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 751208.

BRIEF DESCRIPTION:
THIS SECTION CONTAINS TWO PROCEDURES:
1) PI: DELIVERS THE VALUE OF PI;
2) E: DELIVERS THE VALUE OF E.

KEYWORDS:

MATHEMATICAL CONSTANTS
PI
E

SUBSECTION: PI

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:

```
"REAL" "PROCEDURE" PI;
"CODE" 30006;
```

PI:= THE CONSTANT PI IN 48 BITS PRECISION.

LANGUAGE: COMPASS

SUBSECTION: E

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:

```
"REAL" "PROCEDURE" E;
"CODE" 30007;
```

E:= THE CONSTANT E IN 48 BITS PRECISION.

LANGUAGE: COMPASS

SOURCE TEXTS:

THE SOURCE TEXTS GIVEN HERE ARE NOT THE ACTUAL SOURCE TEXTS, AS THESE PROCEDURES ARE WRITTEN IN COMPASS. EVEN, THE TEXTS GIVEN HERE DO NOT GIVE THE SAME RESULTS ON THE CDC CYBER SYSTEM, SINCE THE ALGOL-60 COMPILER CANNOT READ THE CONSTANTS IN 48 BIT PECISION.

```
"CODE" 30006;
"REAL" "PROCEDURE" PI;
PI:= 3.14159265358979;
```

```
"CODE" 30007;
"REAL" "PROCEDURE" E;
E:= 2.71828182845905;
```

AUTHOR: D.T.WINTER.

INSTITUTE: MATHEMATICAL CENTRE,AMSTERDAM.

RECEIVED: 751208.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SEVEN PROCEDURES:
A) MBASE: DELIVERS THE BASE OF THE ARITHMETIC OF THE COMPUTER;
B) ARREB: DELIVERS THE ARITHMETIC ERROR BOUND OF THE COMPUTER;
C) DWARF: DELIVERS THE SMALLEST (IN ABSOLUTE VALUE) REPRESENTABLE REAL NUMBER;
D) GIANT: DELIVERS THE LARGEST REPRESENTABLE REAL NUMBER;
E) INTCAP: DELIVERS THE INTEGER CAPACITY;
F) OVERFLOW: TESTS WHETHER A VALUE IS AN OVERFLOW VALUE;
G) UNDERFLOW: TESTS WHETHER A VALUE IS AN UNDERFLOW VALUE;
FOR A DETAILED DESCRIPTION SEE METHOD AND PERFORMANCE.

KEYWORDS:

ARITHMETIC CONSTANTS
MACHINE CONSTANTS
OVERFLOW
UNDERFLOW

SUBSECTION: MBASE

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"INTEGER" "PROCEDURE" MBASE;
"CODE" 30001;
```

MBASE:= 2, THE BASE OF THE ARITHMETIC OF THE CYBER.

LANGUAGE: COMPASS

SUBSECTION: ARREB

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"REAL" "PROCEDURE" ARREB;
"CODE" 30002;
```

ARREB:= 2 ** (-47), THE ARITHMETIC RELATIVE ERROR BOUND.

LANGUAGE: COMPASS

SUBSECTION: DWARF

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"REAL" "PROCEDURE" DWARF;
"CODE" 30003;
```

DWARF:= THE SMALLEST (IN ABSOLUTE VALUE) REPRESENTABLE REAL NUMBER.

LANGUAGE: COMPASS

SUBSECTION: GIANT

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"REAL" "PROCEDURE" GIANT;
"CODE" 30004;
```

GIANT:= THE LARGEST REPRESENTABLE REAL NUMBER.

LANGUAGE: COMPASS

SUBSECTION: INTCAP

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"INTEGER" "PROCEDURE" INTCAP;
"CODE" 30005;
```

INTCAP:= THE INTEGER CAPACITY.

LANGUAGE: COMPASS

SUBSECTION: OVERFLOW

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
```
"BOOLEAN" "PROCEDURE" OVERFLOW(X); "VALUE" X; "REAL" X;
"CODE" 30008;
```

THE MEANING OF THE FORMAL PARAMETER IS:
X: <REAL VARIABLE>;
CONTAINS THE VALUE TO BE TESTED.

OVERFLOW DELIVERS "TRUE" IF X CONTAINS AN OVERFLOW VALUE, AND "FALSE" OTHERWISE.

LANGUAGE: COMPASS

SUBSECTION: UNDERFLOW

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"BOOLEAN" "PROCEDURE" UNDERFLOW(X); "VALUE" X; "REAL" X;
"CODE" 30009;

THE MEANING OF THE FORMAL PARAMETER IS:
X: <REAL VARIABLE>;
CONTAINS THE VALUE TO BE TESTED.

UNDERFLOW DELIVERS "TRUE" IF X CONTAINS AN UNDERFLOW VALUE, AND "FALSE" OTHERWISE.

LANGUAGE: COMPASS

METHOD AND PERFORMANCE:

THE PROCEDURES DELIVER THE FOLLOWING VALUES, THAT ARE ESSENTIALLY MACHINE DEPENDENT:
1) MBASE: 2;
2) ARREB: 2**(-47);
3) DWARF: 2**48*2**(-1022);
4) GIANT: (2**48-1)*2**1022;
5) INTCAP: 2**48-2.

FOR MBASE, DWARF AND GIANT THE VALUES ARE CLEAR, WE EXPLAIN THE OTHERS HERE:
ARREB: THIS IS THE SMALLEST POSITIVE NUMBER SO THAT 1+ARREB^=1;
INTCAP: THIS IS THE LARGEST POSITIVE NUMBER SO THAT THE FOLLOWING BOOLEAN EXPRESSION DELIVERS "TRUE" FOR EVERY INTEGER I:
"IF" I<O "OR" I>INTCAP "THEN" "TRUE" "ELSE" I-1^=I;
THE CORRECT VALUE IS NOT 2**48-1, AS IN THE CYBER ARITHMETIC I=J IF I=2**48 AND J=2**48-1.
WARNING: DWARF IS NOT VERY USEFUL WHEN TRAPPING UNDERFLOW VALUES: ABS(X) >= DWARF NEARLY ALWAYS DELIVERS TRUE EVEN IF ABS(X) IS SMALLER THEN DWARF DUE TO THE ARITHMETIC. ONE SHOULD USE: ABS(X) > DWARF (AND ONE TRAPS NON-UNDERFLOW VALUES TOO) OR THE PROCEDURE UNDERFLOW.
NOTE: AS THE ALGOL 60 ERRORMESSAGE "ARITHMETIC OVERFLOW" IS NOT ISSUED AT THE MOMENT THE OVERFLOW VALUE IS CREATED BUT WHEN SUCH A VALUE IS USED, THE PROCEDURE OVERFLOW IS WELL-DEFINED.

EXAMPLE OF USE:

HERE WE GIVE AN EXAMPLE OF USE OF THE PROCEDURES OVERFLOW AND UNDERFLOW:

```
"BEGIN"
    "BOOLEAN" "PROCEDURE" OVERFLOW(X); "CODE" 30009;
    "BOOLEAN" "PROCEDURE" UNDERFLOW(X); "CODE" 30008;
    "REAL" "PROCEDURE" DWARF; "CODE" 30003;
    "REAL" X, Y;
    Y:= 0; X:= 1 / Y;
    "IF" OVERFLOW(X) "THEN" OUTPUT(61, "(""("OVERFLOW")", /")");
    X:= DWARF;
    "IF" "NOT" UNDERFLOW(X) "THEN"
    OUTPUT(61, "(""("NO UNDERFLOW WITH DWARF")", /")");
    X:= X / 2;
    "IF" X ^= 0 "THEN"
    OUTPUT(61, "(""("DWARF / 2 ^= 0")", /")");
    "IF" UNDERFLOW(X) "THEN"
    OUTPUT(61, "(""("DWARF / 2 IS UNDERFLOW")", /")");
    "IF" X * 2 = 0 "THEN"
    OUTPUT(61, "(""("BECAUSE (DWARF / 2) * 2 = 0")", /")")
"END"
```

```
RESULTS:
OVERFLOW
NO UNDERFLOW WITH DWARF
DWARF / 2 ^= 0
DWARF / 2 IS UNDERFLOW
BECAUSE (DWARF / 2) * 2 = 0
```

SOURCE TEXTS:

THESE ARE NOT THE ACTUAL SOURCE TEXTS, AS THESE PROCEDURES ARE WRITTEN IN COMPASS, MOREOVER, THE RESULTS NEED NOT BE THAT OF THE ACTUAL PROCEDURES.

```
"CODE" 30001;
"INTEGER" "PROCEDURE" MBASE;
MBASE:= 2;

"CODE" 30002;
"REAL" "PROCEDURE" ARREB;
ARREB:= 2**(-47);

"CODE" 30003;
"REAL" "PROCEDURE" DWARF;
DWARF:= 2**48*2**(-1022);

"CODE" 30004;
"REAL" "PROCEDURE" GIANT;
GIANT:= (2**48-1)*2**1022;

"CODE" 30005;
"INTEGER" "PROCEDURE" INTCAP;
INTCAP:= 2**48-2;
```

THE SOURCE TEXT OF OVERFLOW AND UNDERFLOW ARE NOT GIVEN HERE, AS THESE EVEN CANNOT BE SIMULATED IN ALGOL-60.

AUTHOR: P.W.HEMKER.

CONTRIBUTOR: F.GROEN.

INSTITUTE: MATHEMATICAL CENTRE, AMSTERDAM.

RECEIVED: 740620.

REVISED: 781101 BY N.M.TEMME AND R.MONTIJN.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THREE PROCEDURES: TAN, ARCSIN, ARCCOS.

TAN COMPUTES THE TANGENT FOR A REAL ARGUMENT X.
ARCSIN COMPUTES THE ARCSINE FOR A REAL ARGUMENT X.
ARCCOS COMPUTES THE ARCCOSINE FOR A REAL ARGUMENT X.

KEYWORDS:

TANGENT,
ARCSINE,
ARCCOSINE.

SUBSECTION: TAN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" TAN(X); "VALUE" X; "REAL" X;
"CODE" 35120;

TAN : DELIVERS THE TANGENT OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS:
X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF TAN(X).

PROCEDURES USED : OVERFLOW = CP 30009,
GIANT = CP 30004.

METHOD AND PERFORMANCE :

THE FORMULA TAN(X) = SIN(X) / COS(X) IS USED. IF COS(X) = 0 THEN THE VALUE OF GIANT (SEE SECTION 6.2) IS DELIVERED.

EXAMPLE OF USE:

```
"BEGIN" "REAL""PROCEDURE" TAN(X); "CODE" 35120;
    OUTPUT(61,"("/"("ARCTAN(TAN(1))= ")",+D.14D")",ARCTAN(TAN(1)));
    OUTPUT(61,"("/"("ARCTAN(TAN(0))= ")",+D.14D")",ARCTAN(TAN(0)));
    OUTPUT(61,"("/"("TAN(ARCTAN(0))= ")",+D.14D")",TAN(ARCTAN(0)));
    OUTPUT(61,"("/"("TAN(ARCTAN(1))= ")",+D.14D")",TAN(ARCTAN(1)));
"END"
```

DELIVERS :

```
ARCTAN(TAN(1))= +1.00000000000000
ARCTAN(TAN(0))= +0.00000000000000
TAN(ARCTAN(0))= +0.00000000000000
TAN(ARCTAN(1))= +1.00000000000000
```

SUBSECTION : ARCSIN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" ARCSIN(X); "VALUE" X; "REAL" X;
"CODE" 35121;

ARCSIN : DELIVERS THE ARCSINE OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS:
X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF ARCSIN(X), ABS(X)<=1.

PROCEDURES USED : NONE.

METHOD AND PERFORMANCE :

FOR ABS(X) < 0.8 WE USE THE FORMULA :
ARCSIN(X) = ARCTAN(X / SQRT (1 - X * X)).
FOR 0.8 <= ABS(X) < 1 WE USE THE FORMULA :
ARCSIN(X) = SIGN(X) * (PI/2 - ARCTAN(SQRT(1/(X * X) - 1))).
FOR ABS(X) = 1 THE VALUE SIGN(X) * PI/2 IS DELIVERED.
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.

EXAMPLE OF USE :

```
"BEGIN" "REAL""PROCEDURE" ARCSIN(X); "CODE" 35121;
    OUTPUT(61,"("/"("ARCSIN(SIN(1))= ")",+D.14D")",ARCSIN(SIN(1)));
    OUTPUT(61,"("/"("ARCSIN(SIN(0))= ")",+D.14D")",ARCSIN(SIN(0)));
    OUTPUT(61,"("/"("SIN(ARCSIN(0))= ")",+D.14D")",SIN(ARCSIN(0)));
    OUTPUT(61,"("/"("SIN(ARCSIN(1))= ")",+D.14D")",SIN(ARCSIN(1)));
"END"
```

DELIVERS :

```
ARCSIN(SIN(1))= +0.99999999999990
ARCSIN(SIN(0))= +0.00000000000000
SIN(ARCSIN(0))= +0.00000000000000
SIN(ARCSIN(1))= +1.00000000000000
```

SUBSECTION: ARCCOS.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"REAL" "PROCEDURE" ARCCOS(X); "VALUE" X; "REAL" X;
"CODE" 35122;
```

ARCCOS : DELIVERS THE ARCCOSINE OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS:

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF ARCCOS(X), ABS(X)<=1.

PROCEDURES USED: NONE.

METHOD AND PERFORMANCE:

FOR 0 < X < 1 WE USE THE FORMULA:
ARCCOS(X) = 2 * ARCTAN(SQRT((1 - X) / (1 + X))).
FOR -1 < X <= 0 WE USE THE FORMULA:
ARCCOS(X) = PI - ARCCOS(-X).
FOR X = 1 THE VALUE 0 IS DELIVERED.
FOR X = -1 THE VALUE PI IS DELIVERED.
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF "-13.

EXAMPLE OF USE:

```
"BEGIN" "REAL""PROCEDURE" ARCCOS(X); "CODE" 35122;
    OUTPUT(61,"("/"("ARCCOS(COS(1))= ")",+D.14D")",ARCCOS(COS(1)));
    OUTPUT(61,"("/"("ARCCOS(COS(0))= ")",+D.14D")",ARCCOS(COS(0)));
    OUTPUT(61,"("/"("COS(ARCCOS(0))= ")",+D.14D")",COS(ARCCOS(0)));
    OUTPUT(61,"("/"("COS(ARCCOS(1))= ")",+D.14D")",COS(ARCCOS(1)));
"END"
```

DELIVERS :

```
ARCCOS(COS(1))= +1.00000000000000
ARCCOS(COS(0))= +0.00000000000000
COS(ARCCOS(0))= +0.00000000000001
COS(ARCCOS(1))= +1.00000000000000
```

SOURCE TEXTS:

```
"CODE" 35120;
"REAL" "PROCEDURE" TAN(X); "VALUE" X; "REAL" X;
"BEGIN" "REAL" U;
    "BOOLEAN" "PROCEDURE" OVERFLOW(X); "CODE" 30009;
    "REAL"    "PROCEDURE" GIANT;       "CODE" 30004;
    U:= SIN(X)/COS(X);
    TAN:= "IF" OVERFLOW(U) "THEN" GIANT "ELSE" U
"END" TAN;
        "EOP"

"CODE" 35121;
"REAL" "PROCEDURE" ARCSIN(X); "VALUE" X; "REAL" X;
"BEGIN" "REAL" U; U:= ABS(X);
    ARCSIN:= "IF" U<0.8 "THEN" ARCTAN(X/SQRT(1-X*X)) "ELSE"
             SIGN(X) * ( "IF" U=1  "THEN" 1.57079632679489 "ELSE"
             ( 1.57079632679489 - ARCTAN(SQRT(1/(X*X)-1))))
"END" ARCSIN;
        "EOP"

"CODE" 35122;
"REAL" "PROCEDURE" ARCCOS(X); "VALUE" X; "REAL" X;
"BEGIN" "REAL" U,V; U:= ABS(X); V:= (1-U)/(1+U);
    V:= "IF" V  =0 "THEN" 0 "ELSE"
        "IF" U+1=1 "THEN" 1.57079632679489 "ELSE"
                          2*ARCTAN(SQRT(V));
    ARCCOS:= "IF" X>0 "THEN" V "ELSE" 3.14159265358979 - V
"END" ARCCOS;
        "EOP"
```

AUTHOR: P.W.HEMKER.

CONTRIBUTOR: F.GROEN.

INSTITUTE: MATHEMATICAL CENTRE, AMSTERDAM.

RECEIVED: 730921.

REVISED: 781101 BY N.M.TEMME AND R.MONTIJN.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SIX PROCEDURES FOR THE COMPUTATION OF HYPERBOLIC FUNCTIONS.

SINH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF SINH(X).
COSH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF COSH(X).
TANH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF TANH(X).
ARCSINH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF ARCSINH(X).
ARCCOSH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF ARCCOSH(X).
ARCTANH COMPUTES FOR A REAL ARGUMENT X THE VALUE OF ARCTANH(X).

KEYWORDS:

HYPERBOLIC SINE,
HYPERBOLIC COSINE,
HYPERBOLIC TANGENT,
HYPERBOLIC ARCSINE,
HYPERBOLIC ARCCOSINE,
HYPERBOLIC ARCTANGENT.

SUBSECTION : SINH.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" SINH(X); "VALUE" X; "REAL" X;
"CODE" 35111;
```

SINH : DELIVERS THE HYPERBOLIC SINE OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF SINH(X).

PROCEDURES USED : OVERFLOW = CP 30009,
GIANT = CP 30004.

METHOD AND PERFORMANCE :

IF ABS(X) < 0.1 THEN SINH(X) IS CALCULATED BY MEANS OF AN ECONOMIZED TAYLOR SERIES.
IF 0.1 <= ABS(X) < 0.3 WE USE THE FORMULA :
SINH(X) = 3 * SINH (X/3) + 4 * SINH (X/3) ** 3
IF 0.3 <= ABS(X) < 17.5 THEN WE USE THE FORMULA :
SINH(X) = 0.5 * (EXP(X) - EXP(-X)).
IF X >= 17.5 THEN WE TAKE SINH(X) = SIGN(X) * EXP(X-LN(2)).
IN THE CASE OF OVERFLOW (I.E., ABS(X) > 741.6 (APPROXIMATELY)) THEN THE VALUE SINH = SIGN(X) * GIANT (SEE SUBSECTION 6.2) IS DELIVERED.
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.

EXAMPLE OF USE :

SEE EXAMPLE OF USE OF THE PROCEDURE COSH (THIS SECTION).

SUBSECTION : COSH.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" COSH(X); "VALUE" X; "REAL" X;
"CODE" 35112;
```

COSH : DELIVERS THE HYPERBOLIC COSINE OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF COSH(X).

PROCEDURES USED : SINH = CP 35111.

METHOD AND PERFORMANCE :

IF ABS(X) < 17.5 THE FORMULA COSH(X) = 0.5 * (EXP(X) + EXP(-X)) IS USED ELSE COSH(X) = SINH(ABS(X)).
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.

EXAMPLE OF USE :

THE FOLLOWING PROGRAM TESTS FOR X = -20, -2, -1, 0.1, 0.3 THE RELATION : SINH(2 * X) - 2 * SINH(X) * COSH(X) = 0.

```
"BEGIN""REAL" X;
    "REAL""PROCEDURE" SINH(X); "CODE" 35111;
    "REAL""PROCEDURE" COSH(X); "CODE" 35112;

    "FOR" X := -20, -2, -1, 0.1, 0.3  "DO"
    OUTPUT(61,"("/,+2ZD.D,3B,+D.D"+3D")",X,SINH(2 * X)
    - 2 * SINH(X) * COSH(X) );
"END"
```

OUTPUT :

```
-20.00   +6.1"+003
 -2.00   -1.1"-013
 -1.00   -1.4"-014
 +0.10   +0.0"+000
 +0.30   +0.0"+000
```

SUBSECTION : TANH.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" TANH(X); "VALUE" X; "REAL" X;
"CODE" 35113;
```

TANH : DELIVERS THE HYPERBOLIC TANGENT OF TH ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF TANH(X).

PROCEDURES USED : SINH = CP 35111.

METHOD AND PERFORMANCE :

IF ABS(X) < 0.005 THE TANH(X) IS CALCULATED BY A TRUNCATED POWER SERIES (TAYLOR'S FORMULA).
IF 0.005 <= ABS(X) < 0.3 WE USE THE FORMULA :
TANH(X) = SINH(X) / COSH(X).
IF 0.3 <= ABS(X) <= 17.5 WE USE THE FORMULA :
TANH(X) = (1 - EXP(-2 * X)) / (1 + EXP(-2 * X)).
IF ABS(X) > 17.5 THE VALUE SIGN(X) IS DELIVERED.
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.

EXAMPLE OF USE :

THE FOLLOWING PROGRAM CHECKS FOR X = -100, -10, 0, 2, 5 THE RELATION : 1 - TANH(X) ** 2 - 1 / COSH(X) ** 2 = 0.

```
"BEGIN" "REAL" X ;
  "REAL" "PROCEDURE" COSH(X); "CODE" 35112;
  "REAL" "PROCEDURE" TANH(X); "CODE" 35113;
  "FOR" X := -100, -10, 0, 2, 5 "DO"
  OUTPUT(61,"("/,+2ZD,3B,+D.D"+3D")",X,1-TANH(X)**2-1/COSH(X)**2);
"END"
```

RESULTS :

```
-100   -5.5"-087
 -10   +1.2"-014
  +0   +0.0"+000
  +2   +9.8"-015
  +5   -3.4"-015
```

SUBSECTION : ARCSINH.

CALLING SEQUENCE :

```
    THE HEADING OF THE PROCEDURE READS :
    "REAL" "PROCEDURE" ARCSINH(X); "VALUE" X; "REAL" X;
    "CODE" 35114;

    ARCSINH : DELIVERS THE INVERSE HYPERBOLIC SINE OF THE ARGUMENT X.

    THE MEANING OF THE FORMAL PARAMETER IS :
    X:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE (REAL) ARGUMENT OF ARCSINH(X).
```

PROCEDURES USED : LOG ONE PLUS X = CP 35130.

METHOD AND PERFORMANCE :

```
    IF ABS(X) <= "10 WE USE THE PROCEDURE LOG ONE PLUS X (SEE SECTION
        6.4.3.) BY WRITING :
        ARCSINH(X) = LN ( X + SQRT ( X * X + 1 ) ) =
        LN(1+X+X**2/(1+SQRT(1+X**2))).
    IF ABS(X) > "10 WE USE THE FORMULA :
        ARCSINH(X) = SIGN(X) * ( LN(2) + LN ( ABS(X) ) ).
    THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.
```

EXAMPLE OF USE :

```
    "BEGIN"
        "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
        "REAL" "PROCEDURE" ARCSINH(X); "CODE" 35114;
        OUTPUT(61,"("/,D.14D")",ARCSINH(SINH(0.01)));
        OUTPUT(61,"("/,D.14D")",ARCSINH(SINH(0.05)));
        OUTPUT(61,"("/,D.14D")",SINH(ARCSINH(0.05)));
        OUTPUT(61,"("/,D.14D")",SINH(ARCSINH(0.01)));
    "END"

    DELIVERS :

    +0.01000000000000
    +0.05000000000000
    +0.05000000000000
    +0.01000000000000
```

SUBSECTION : ARCCOSH.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" ARCCOSH(X); "VALUE" X; "REAL" X;
"CODE" 35115;
```

ARCCOSH : DELIVERS THE INVERSE HYPERBOLIC COSINE OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF ARCCOSH(X), X >= 1.

PROCEDURES USED : NONE.

METHOD AND PERFORMANCE :

IF X = 1 THE VALUE 0 IS DELIVERED.
IF 1 < X <= "10 WE USE THE FORMULA :
ARCCOSH(X) = LN (X + SQRT (X * X - 1)).
IF X > "10 WE USE THE FORMULA :
ARCCOSH(X) = LN(2) + LN (X).
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.
IF X IS CLOSE TO 1, SAY X = 1+Y, Y>0, AND Y IS KNOWN IN GOOD RELATIVE PRECISION, THEN IT IS ADVISED TO USE THE PROCEDURE LOG ONE PLUS X (SEE SUBSECTION 6.4.3) BY WRITING
ARCCOSH(X) = LN(1 + Y + SQRT(Y*(Y+2))).
EXAMPLE : X = EXP(T), T > 0, T IS SMALL. THEN Y = EXP(T)-1 IS AVAILABLE IN GOOD RELATIVE ACCURACY, Y = 2*EXP(T/2)*SINH(T/2).

EXAMPLE OF USE :

```
"BEGIN"
    "REAL" "PROCEDURE" COSH(X); "CODE" 35112;
    "REAL" "PROCEDURE" ARCCOSH(X); "CODE" 35115;
    OUTPUT(61,"("/,D.14D")",ARCCOSH(COSH(0.01)));
    OUTPUT(61,"("/,D.14D")",ARCCOSH(COSH(0.05)));
    OUTPUT(61,"("/,D.14D")",COSH(ARCCOSH(1.01)));
    OUTPUT(61,"("/,D.14D")",COSH(ARCCOSH(1.05)));
"END"
```

DELIVERS :

```
+0.00999999999958
+0.04999999999999
+1.01000000000000
+1.05000000000000
```

SUBSECTION : ARCTANH.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"REAL" "PROCEDURE" ARCTANH(X); "VALUE" X; "REAL" X;
"CODE" 35116;

ARCTANH : DELIVERS THE INVERSE HYPERBOLIC TANGENT OF THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETER IS :
X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF ARCTANH(X).

PROCEDURES USED : LOG ONE PLUS X = CP 35130,
GIANT = CP 30004.

METHOD AND PERFORMANCE :

IF ABS(X) < 1 WE USE THE PROCEDURE LOG ONE PLUS X (SEE SECTION 6.4.3) BY WRITING ARCTANH(X) = 0.5 * LN((1 + X)/(1 - X))= 0.5 * LN(1+2*X/(1-X)).
IF ABS(X) = 1 THE VALUE IS SIGN(X) * GIANT (SEE SECTION 6.2).
THE VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.

EXAMPLE OF USE :

```
"BEGIN"
    "REAL" "PROCEDURE" TANH(X); "CODE" 35113;
    "REAL" "PROCEDURE" ARCTANH(X); "CODE" 35116;
    OUTPUT(61,"("/,D.14D")",ARCTANH(TANH(0.01)));
    OUTPUT(61,"("/,D.14D")",ARCTANH(TANH(0.05)));
    OUTPUT(61,"("/,D.14D")",TANH(ARCTANH(0.05)));
    OUTPUT(61,"("/,D.14D")",TANH(ARCTANH(0.01)));
"END"
```

DELIVERS :

```
+0.01000000000000
+0.05000000000000
+0.05000000000000
+0.01000000000000
```

SOURCE TEXTS :

```
"CODE" 35111;
 "REAL" "PROCEDURE" SINH(X); "VALUE" X; "REAL" X;
 "BEGIN" "REAL" AX,Y;
    AX:= ABS(X);
    "IF" AX < 0.3 "THEN"
    "BEGIN" Y:= "IF" AX < 0.1 "THEN" X*X "ELSE" X*X/9;
        X:=  ((( 0.0001984540      * Y +
                 0.0083333331783  )* Y +
                 0.16666666666675)* Y +
                 1.0              )* X ;
       SINH:= "IF" AX < 0.1 "THEN" X "ELSE"
              X * ( 1.0 + 0.148148148148l5 * X * X )
    "END" "ELSE" "IF" AX < 17.5 "THEN"
    "BEGIN" AX:= EXP( AX ); SINH:= SIGN(X) * .5 * ( AX -1/AX ) "END"
    "ELSE" "IF" AX > 742.36063037970 "THEN"
    "BEGIN" "REAL" "PROCEDURE" GIANT; "CODE" 30004;
        SINH:= SIGN(X)*GIANT
    "END" "ELSE"
    SINH:= SIGN(X)*EXP(AX- .69314 71805 59945)
 "END" SINH;
        "EOP"

"CODE" 35112;
 "REAL" "PROCEDURE" COSH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X) < 17.5 "THEN"
 "BEGIN" X:= EXP(X); COSH:= 0.5 * ( X + 1/X ) "END" "ELSE"
 "BEGIN" "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
     COSH:= SINH(ABS(X))
 "END" COSH;
        "EOP"

"CODE" 35113;
 "REAL" "PROCEDURE" TANH(X); "VALUE" X; "REAL" X;
 "BEGIN" "REAL"AX;"REAL" "PROCEDURE"SINH(X); "CODE" 35111; AX:= ABS(X);
  "IF" AX < 0.005 "THEN"
  "BEGIN" "REAL" Y; Y:= X*X; TANH:= X * ( 1 - Y *
     (.33333333333333 - Y *
     (.13333333333333 - Y *
      .05396825396825    )))
  "END" "ELSE" "IF" AX < 0.3 "THEN"
  "BEGIN" "REAL" SH;
     SH:= SINH(X);
     TANH:= SH/SQRT(1+SH*SH)
  "END" "ELSE"
  "IF" AX > 17.5 "THEN" TANH:= SIGN(X) "ELSE"
  "BEGIN" AX:= EXP(-2*AX); TANH:= SIGN(X)*(1-AX)/(1+AX) "END"
 "END";
        "EOP"
```

```
"CODE" 35114;
 "REAL" "PROCEDURE" ARCSINH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X) > "10 "THEN" ARCSINH:= SIGN(X)*(0.69314 71805 5995+
     LN(ABS(X))) "ELSE"
 "BEGIN" "REAL" Y; "REAL" "PROCEDURE" LOG ONE PLUS X(X); "CODE" 35130;
     Y:= X*X; ARCSINH:= SIGN(X)*LOG ONE PLUS X(ABS(X)+Y/(1+SQRT(1+Y)))
 "END" ARCSINH;
        "EOP"

"CODE" 35115;
 "REAL" "PROCEDURE" ARCCOSH(X); "VALUE" X; "REAL" X;
 ARCCOSH:= "IF" X <= 1 "THEN" 0 "ELSE"
           "IF" X > "10 "THEN" 0.69314718055995 + LN(X) "ELSE"
           LN(X+SQRT((X-1)*(X+1)));
        "EOP"

"CODE" 35116;
 "REAL" "PROCEDURE" ARCTANH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X) >= 1 "THEN"
 "BEGIN" "REAL" "PROCEDURE" GIANT; "CODE" 30004;
     ARCTANH:= SIGN(X)*GIANT
 "END" "ELSE"
 "BEGIN" "REAL" AX; "REAL" "PROCEDURE" LOG ONE PLUS X(X); "CODE" 35130;
     AX:= ABS(X); ARCTANH:= SIGN(X)*.5*LOG ONE PLUS X(2*AX/(1-AX))
 "END" ARCTANH;
        "EOP"
```

AUTHOR : N.M. TEMME.

CONTRIBUTOR : R. MONTIJN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED : 780801.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURE LOG ONE PLUS X FOR THE COMPUTATION OF LN(1+X) FOR X > -1.

KEYWORDS : LOGARITHMIC FUNCTION.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" LOG ONE PLUS X(X); "VALUE" X; "REAL" X;
"CODE" 35130;
LOG ONE PLUS X : DELIVERS THE VALUE OF LN(1+X);
```

THE MEANING OF THE FORMAL PARAMETER IS :

```
X:      <ARITHMETIC  EXPRESSION>;
        ENTRY :  THE ARGUMENT OF LN(1+X), X > -1.
```

PROCEDURES USED : NONE.

RUNNING TIME : THE ALGORITHM NEEDS 9 MULTIPLICATIONS.

METHOD AND PERFORMANCE :

FOR X < -0.2928 OR X > 0.4142 THE PROCEDURE USES THE STANDARD FUNCTION LN, FOR -0.2928 <= X <= 0.4142 A POLYNOMIAL APPROXIMATION IS USED.
WE USE AN APPROXIMATION BASED ON THE BEST APPROXIMATON FOR THE INTERVAL 1/SQRT(2)-1 <= X <= SQRT(2)-1, OF WHICH THE COEFFICIENTS ARE GIVEN IN HART (1968); CF. P. 111, INDEX 2665.
THE PROCEDURE LOG ONE PLUS X COMPUTES LN(1+X) WITH RELATIVE ACCURACY COMPARABLE WITH THE MACHINE ACCURACY.

AS IS WELL KNOWN, FOR SMALL ABS(X) RELATIVE ACCURACY IS LOST WHEN COMPUTING LN(1+X) BY USING THE STANDARD FUNCTION LN. THE PROCEDURE IS USED IN THE PROCEDURES ARCSINH AND ARCTANH, SECTION 6.4.2.

REFERENCES : HART, J.F. CS. (1968), COMPUTER APPROXIMATIONS, WILEY, NEW YORK.

EXAMPLE OF USE :

WE COMPUTE LN(EXP(X)) FOR SMALL POSITIVE X. IN ORDER TO PRESERVE RELATIVE ACCURACY WE WRITE

```
LN ( EXP(X) )   =   LN ( 1+ EXP(X)-1 )
                =   LN ( 1+ 2* EXP(X/2)* SINH(X/2) ).
```

THE FOLOWING PROGRAM

```
"BEGIN" "REAL" X,Y;
    "REAL" "PROCEDURE" SINH(X)          ; "CODE" 35111;
    "REAL" "PROCEDURE" LOG ONE PLUS X(X); "CODE" 35130;
    "FOR" X:= "-1, "-10, "-50, "-100, "-250 "DO"
    "BEGIN" Y:= LOG ONE PLUS X( 2*EXP(X/2)*SINH(X/2) );
        OUTPUT(61,"("N,/")",Y)
    "END"
"END";
```

PRINTS THE FOLOWING RESULTS :

```
+1.0000000000000"-001
+1.0000000000000"-010
+1.0000000000000"-050
+1.0000000000000"-100
+1.0000000000000"-250
```

SOURCE TEXT :

```
"CODE" 35130;
"REAL" "PROCEDURE" LOG ONE PLUS X(X); "VALUE" X; "REAL" X;
"COMMENT" COMPUTES LN(1+X) FOR X > -1;
"IF" X = 0 "THEN" LOG ONE PLUS X:= 0 "ELSE"
"IF" X < -0.2928 "OR" X > 0.4142 "THEN" LOG ONE PLUS X:= LN(1+X) "ELSE"
"BEGIN" "REAL" Y,Z;
    Z:= X/(X+2); Y:= Z*Z;
    LOG ONE PLUS X:= Z*(2+ Y*
    ( .66666 66666 63366 + Y*
    ( .40000 00012 06045 + Y*
    ( .28571 40915 90488 + Y*
    ( .22223 82333 2791  + Y*
    ( .18111 36267 967   + Y*
      .16948 21248 8))))))
"END" LOG ONE PLUS X;
        "EOP"
```

AUTHOR(S) : H.FIOLET, N.TEMME.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED: 740628.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS FOUR PROCEDURES :

A.
EI CALCULATES THE EXPONENTIAL INTEGRAL DEFINED AS FOLLOWS (SEE ALSO REF[1] , EQ. (5.1.1)) : EI(X) = INTEGRAL (EXP(T)/T DT) FROM T=-INFINITY TO T=X , WHERE THE INTEGRAL IS TO BE INTERPRETED AS THE CAUCHY PRINCIPAL VALUE. ALSO THE RELATED FUNCTION E1(X), DEFINED BY THE INTEGRAL (EXP(-T)/T DT) FROM T= X TO T= INFINITY, FOR POSITIVE X (REF[1], EQ.(5.1.2)) CAN EASILY BE OBTAINED BY THE RELATION E1(X) = - EI(-X). FOR X=0 THE INTEGRAL IS UNDEFINED AND THE PROCEDURE WILL CAUSE OVERFLOW.

B.
EIALPHA CALCULATES A SEQUENCE OF INTEGRALS OF THE FORM
  INTEGRAL(EXP(-X*T)*T**I DT)
FROM T=1 TO T= INFINITY,
WHERE X IS POSITIVE AND I = 0,...,N.
(SEE ALSO REF[1], EQ. (5.1.5)).

C.
ENX COMPUTES A SEQUENCE OF INTEGRALS E(N,X), N=N1, N1+1,...,N2, WHERE X>0 AND N1, N2 ARE POSITIVE INTEGERS WITH N2>=N1; E(N,X) IS DEFINED AS FOLLOWS:
  E(N,X)= THE INTEGRAL FROM 1 TO INFINITY OF EXP(-X * T)/ T**N DT;
(SEE ALSO REF[1], EQ.(5.1.4));

D.
NONEXPENX COMPUTES A SEQUENCE OF INTEGRALS EXP(X)*E(N,X), N=N1, N1+1,...,N2, WHERE X>0 AND N1, N2 ARE POSITIVE INTEGERS WITH N2<=N1; E(N,X) IS DEFINED UNDER C).

KEYWORDS :

EXPONENTIAL INTEGRAL,
SPECIAL FUNCTIONS.

SUBSECTION : EI.

CALLING SEQUENCE :

```
THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" EI(X);
"VALUE" X;"REAL" X;

EI:      DELIVERS THE VALUE OF THE EXPONENTIAL INTEGRAL;

THE MEANING OF THE FORMAL PARAMETER IS :
X:       <ARITHMETIC EXPRESSION>;
         THE ARGUMENT OF THE INTEGRAL.
```

PROCEDURES USED :

```
CHEPOLSER = CP31046 ,
POL       = CP31040 ,
JFRAC     = CP35083 .
```

RUNNING TIME : CIRCA 3.2"-3 SEC.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

THE INTEGRAL IS CALCULATED BY MEANS OF THE RATIONAL CHEBYSHEV APPROXIMATIONS GIVEN IN REFERENCES [1] AND [2]. ONLY RATIOS OF POLYNOMIALS WITH EQUAL DEGREE L ARE CONSIDERED. BELOW,THE DIFFERENT INTERVALS ARE LISTED, TOGETHER WITH THE CORRESPONDING DEGREE L AND THE NUMBER OF CORRECT DIGITS OF THE APPROXIMATIONS :

[-INFINITY,-4]	6	15.1
[-4,-1]	7	16.9
[-1, 0]	5	18.5
[0, 6]	7	15.2
[6,12]	7	15.1
[12,24]]	7	15.0
[24,+INFINITY]	7	15.9 .

VARIOUS TESTS SHOWED A RELATIVE ACCURACY OF AT LEAST "-13, EXEPT IN THE NEIGHBOURHOOD OF X=.37250 , THE ZERO OF THE INTEGRAL, WHERE ONLY AN ABSOLUTE ACCURACY OF .3"-13 IS REACHED . IN SOME OF THE INTERVALS , THE RATIONAL FUNCTIONS ARE EXPRESSED EITHER AS RATIOS OF FINITE SUMS OF CHEBYSHEV POLYNOMIALS OR AS J-FRACTIONS, SINCE THE ORIGINAL FORMS ARE POORLY CONDITIONED.

REFERENCES : SEE REFERENCES [1], [2] AND [3] OF THE PROCEDURE NONEXPENX (THIS SECTION).

EXAMPLE OF USE :

```
"BEGIN"
    "COMMENT" THE COMPUTATION OF E1(.5);
    "REAL" "PROCEDURE" EI(X);"CODE" 35080;
    OUTPUT(61,"("N")",-EI(-.5))
"END"

DELIVERS :
+5.5977359477616"-001
```

SUBSECTION : EIALPHA.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" EIALPHA(X,N,ALPHA);
"VALUE" N,X;"INTEGER" N;"REAL" X;"ARRAY" ALPHA;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

X: <ARITHMETIC EXPRESSION>;
THE REAL X OCCURING IN THE INTEGRAND.

N: <ARITHMETIC EXPRESSION>;
THE INTEGER N OCCURING IN THE INTEGRAND;

ALPHA: <ARRAY IDENTIFIER>;
"ARRAY" ALPHA[0:N];
THE VALUE OF THE INTEGRAL(EXP(-X*T)*T**I DT) FROM T=1 TO T=INFINITY IS STORED IN ALPHA[I].

PROCEDURES USED : NONE.

RUNNING TIME : CIRCA (6 + N * .8) * "-4 SEC.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

THE INTEGRAL IS CALCULATED BY MEANS OF THE RECURSION FORMULA A[N]:=A[0] + N * A[N-1] / X, WITH A[0]:= EXP(-X)/X. FOR X CLOSE TO ZERO, EIALPHA MIGHT CAUSE OVERFLOW, SINCE THE VALUE OF THE INTEGRAL IS INFINITE FOR X=0. THE PROCEDURE IS NOT PROTECTED AGAINST THIS TYPE OF OVERFLOW. THE MINIMAL VALUE FOR THE ARGUMENT X DEPENDS ON THE PARAMETER N :

```
N=20    X CIRCA "-14
N=15    X CIRCA "-18
N=10    X CIRCA "-28
N= 5    X CIRCA "-53
```

THE RECURSION FORMULA IS STABLE AND VARIOUS TESTS EXECUTED ON THE CD CYBER 7228 SHOWED A RELATIVE ACCURACY OF AT LEAST .2"-12.

EXAMPLE OF USE :

```
"BEGIN"
    "PROCEDURE" EIALPHA(X,N,ALPHA);"CODE" 35081;
    "INTEGER" K;"REAL" "ARRAY" A[0:5];
    EIALPHA(.25,5,A);
    "FOR" K:=0 "STEP" 1 "UNTIL" 5 "DO"
    OUTPUT(61,"("DBBB,N,/")",K,A[K]);
"END"
```

DELIVERS :

```
0    +3.1152031322856"+000
1    +1.5576015661428"+001
2    +1.2772332842371"+002
3    +1.5357951442168"+003
4    +2.4575837510601"+004
5    +4.9151986541516"+005   .
```

REFERENCES: SEE REFERENCE [1] OF THE PROCEDURE NONEXPENX(THIS SECTION).

SUBSECTION: ENX.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" ENX(X, N1, N2, A);
"VALUE" X, N1, N2; "REAL" X; "INTEGER" N1, N2; "ARRAY" A;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X :      <ARITHMETIC EXPRESSION>;
         ENTRY: THE (REAL) POSITIVE X OCCURING IN THE INTEGRAND;
N1, N2:  <ARITHMETIC EXPRESSION>;
         ENTRY: LOWER AND UPPER BOUND, RESPECTIVELY, OF THE  INTEGER
                N OCCURING IN THE INTEGRAND;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[N1:N2];
         EXIT:  THE VALUE OF THE INTEGRAL(EXP(-X * T)/T**I DT)  FROM
                T=1 TO T= INFINITY IS STORED IN A[I].
```

PROCEDURES USED:

```
EI        = CP35080,
NONEXPENX = CP35087.
```

RUNNING TIME:
DEPENDS STRONGLY ON THE VALUES OF X, N1, AND N2, WITH A MAXIMUM OF ROUGHLY (5 + .1 * NUMBER OF NECESSARY ITERATIONS) MSEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
SEE METHOD AND PERFORMANCE OF THE PROCEDURE NONEXPENX(THIS SECTION)

SUBSECTION: NONEXPENX.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS :
    "PROCEDURE" NONEXPENX(X, N1, N2, A);
    "VALUE" X, N1, N2; "REAL" X; "INTEGER" N1, N2; "ARRAY" A;

    THE MEANING OF THE FORMAL PARAMETERS IS :
    X:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE (REAL) POSITIVE X OCCURING IN THE INTEGRAND;
    N1, N2: <ARITHMETIC EXPRESSION>;
            ENTRY: LOWER  AND UPPER BOUND, RESPECTIVELY, OF THE INTEGER
                   N OCCURING IN THE INTEGRAND;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[N1:N2];
            EXIT:  THE  VALUE  OF  EXP(X) * INTEGRAL(EXP(-X*T)/T**I DT)
                   FROM T=1 TO T=INFINITY IS STORED IN A[I].
```

PROCEDURES USED:
    ENX = CP35086.

RUNNING TIME:
    DEPENDS STRONGLY ON THE VALUES OF X, N1, AND N2, WITH A MAXIMUM
    OF ROUGHLY (5 + .1 * NUMBER OF NECESSARY ITERATIONS) MSEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

```
    THE SEQUENCE OF INTEGRALS IS GENERATED BY MEANS  OF THE  RECCURENCE
    RELATION:
          E(N+1,X) = (EXP(-X) - X * E(N,X))/N.
    FOR REASONS OF STABILITY THE RECURSION STARTS  WITH  E(NO,X), WHERE
    NO=ENTIER(X+.5), (SEE ALSO REF[5]). THE INTEGRALS ARE THEN COMPUTED
    BY BACKWARD RECURRENCE IF N<NO AND BY FORWARD  RECURRENCE IF  N>NO.
    TO  OBTAIN   THE  STARTING  VALUES   E(NO,X) OF  THE RECURSION  THE
    FOLLOWING CASES ARE DISTINGUISHED:
    A) NO = 1: THE PROCEDURE EI IS USED (SECTION 6.5);
    B) NO<=10: A TAYLOR  EXPANSION   ABOUT  X=NO IS USED, WHICH MADE IT
               NECESSARY TO STORE THE VALUES OF E(N,N) IN THE PROCEDURE
               FOR N= 2, 3,...,10;
    C) NO >10: THE FOLLOWING CONTINUED FRACTION IS USED:
               EXP(X)*E(N,X) = 1/(X+N/(1+1/(X+(N+1)/(1+...)))),
               (SEE ALSO REF[4], EQ.(2.3));
    THE CASES A) AND B) ARE TREATED IN ENX, WHILE  NONEXPENX  EVALUATES
    THE CONTINUED FRACTION IN CASE C).
    ENX CALLS FOR NONEXPENX IN CASE C).
    NONEXPENX CALLS FOR ENX IN THE CASES A) AND B).
    VARIOUS TESTS SHOWED A RELATIVE ACCURACY OF AT LEAST 5"-14.
```

REFERENCES:

[1].M.ABRAMOWITZ AND I.A.STEGUN.
HANDBOOK OF MATHEMATICAL FUNCTIONS.
DOVER PUBLICATIONS, INC. NEW YORK (1965).

[2] W.J.CODY AND H.C.THACHER, JR.
RATIONAL CHEBYSHEV APPROXIMATIONS FOR THE EXPONENTIAL INTEGRAL E1(X).
MATH. COMP. 22 (JULY 1968), 641-649.

[3] W.J.CODY AND H.C.THACHER, JR.
CHEBYSHEV APPROXIMATIONS FOR THE EXPONENTIAL INTEGRAL EI(X).
MATH. COMP. 23 (APRIL 1969), 289-303.

[4].W.GAUTSCHI.
EXPONENTIAL INTEGRALS.
CACM, DECEMBER 1973, P.761-763.

[5].W.GAUTSCHI.
RECURSIVE COMPUTATION OF CERTAIN INTEGRALS.
JACM, VOL.8, 1961, P.21-40.

EXAMPLE OF USE:

IN THE FOLLOWING PROGRAM WE COMPUTE THE VALUES OF E(40,1.1), E(41,1.1), E(42,1.1) AND EXP(X)*E(1,50.1).

```
"BEGIN"
    "PROCEDURE"        ENX(X, N1, N2, A); "CODE" 35086;
    "PROCEDURE" NONEXPENX(X, N1, N2, A); "CODE" 35087;

    "INTEGER" I;
    "REAL" "ARRAY" A[40:42], B[1:1];

    ENX(1.1, 40, 42, A);
    "FOR" I:= 40, 41, 42 "DO"
    OUTPUT(61,"("4B,"("E(")",DD,"(",1.1)=  ")",N/")",I,A[I]);
    NONEXPENX(50.1, 1, 1, B);
    OUTPUT(61,"("/,4B,"("EXP(50.1)*E(1,50.1)=  ")",N")",B[1]);
"END"
```

THIS PROGRAM DELIVERS:

```
E(40,1.1)=  +8.2952134128634"-003
E(41,1.1)=  +8.0936587235982"-003
E(42,1.1)=  +7.9016599781006"-003

EXP(50.1)*E(1,50.1)=  +1.9576696324723"-002
```

SOURCE TEXT(S):

```
"CODE" 35080;
"REAL" "PROCEDURE" EI(X);"VALUE" X;"REAL" X;
"BEGIN" "REAL" "ARRAY" P,Q[0:7];
    "REAL" "PROCEDURE" CHEPOLSER(N,X,A);"CODE" 31046;
    "REAL" "PROCEDURE" POL(N,X,A);"CODE" 31040;
    "REAL" "PROCEDURE" JFRAC(N,A,B);"CODE" 35083;

    "IF" X>24 "THEN"
    "BEGIN" P[0]:= +1.00000000000058    ;Q[1]:= 1.99999999924131    ;
            P[1]:=X-3.00000016782085    ;Q[2]:=-2.99996432944446    ;
            P[2]:=X-5.00140345515924    ;Q[3]:=-7.90404992298926    ;
            P[3]:=X-7.49289167792884    ;Q[4]:=-4.31325836146628    ;
            P[4]:=X-3.08336269051763"+1;Q[5]:= 2.95999399486831"+2;
            P[5]:=X-1.39381360364405    ;Q[6]:=-6.74704580465832    ;
            P[6]:=X+8.91263822573708    ;Q[7]:= 1.04745362652468"+3;
            P[7]:=X-5.31686623494482"+1;
       EI:=EXP(X)*(1+JFRAC(7,Q,P)/X)/X
    "END" "ELSE" "IF" X>12 "THEN"
    "BEGIN" P[0]:= +9.99994296074708"-1;Q[1]:= 1.00083867402639    ;
            P[1]:=X-1.95022321289660    ;Q[2]:=-3.43942266899870    ;
            P[2]:=X+1.75656315469614    ;Q[3]:= 2.89516727925135"+1;
            P[3]:=X+1.79601688769252"+1;Q[4]:= 7.60761148007735"+2;
            P[4]:=X-3.23467330305403"+1;Q[5]:= 2.57776384238440"+1;
            P[5]:=X-8.28561994140641    ;Q[6]:= 5.72837193837324"+1;
            P[6]:=X-1.86545454883399"+1;Q[7]:= 6.95000655887434"+1;
            P[7]:=X-3.48334653602853    ;
       EI:=EXP(X)*JFRAC(7,Q,P)/X
    "END" "ELSE" "IF" X>6 "THEN"
    "BEGIN" P[0]:= +1.00443109228078    ;Q[1]:= 5.27468851962908"-1;
            P[1]:=X-4.32531132878135"+1;Q[2]:= 2.73624119889328"+3;
            P[2]:=X+6.01217990830080"+1;Q[3]:= 1.43256738121938"+1;
            P[3]:=X-3.31842531997221"+1;Q[4]:= 1.00367439516726"+3;
            P[4]:=X+2.50762811293560"+1;Q[5]:=-6.25041161671876    ;
            P[5]:=X+9.30816385662165    ;Q[6]:= 3.00892648372915"+2;
            P[6]:=X-2.19010233854880"+1;Q[7]:= 3.93707701852715    ;
            P[7]:=X-2.18086381520724    ;
       EI:=EXP(X)*JFRAC(7,Q,P)/X
    "END" "ELSE" "IF" X>0 "THEN"
    "BEGIN" "REAL" T,R,X0,XMX0;
            P[0]:=-1.95773036904548"+8;Q[0]:=-8.26271498626055"+7;
            P[1]:= 3.89280421311201"+6;Q[1]:= 8.91925767575612"+7;
            P[2]:=-2.21744627758845"+7;Q[2]:=-2.49033375740540"+7;
            P[3]:=-1.19623669349247"+5;Q[3]:= 4.28559624611749"+6;
            P[4]:=-2.49301393458648"+5;Q[4]:=-4.83547436162164"+5;
            P[5]:=-4.21001615357070"+3;Q[5]:= 3.57300298058508"+4;
            P[6]:=-5.49142265521085"+2;Q[6]:=-1.60708926587221"+3;
            P[7]:=-8.66937339951070    ;Q[7]:= 3.41718750000000"+1;
       X0:=.372507410781367;
       T:=X/3-1;
       R:=CHEPOLSER(7,T,P)/CHEPOLSER(7,T,Q);
       XMX0:=(X-409576229586/1099511627776)-.767177250199394"-12;
                                                                "COMMENT"
```

;

```
        "IF" ABS(XMXO)>.037 "THEN" T:=LN(X/XO) "ELSE"
        "BEGIN" "REAL" Z,Z2;
             P[0]:= .837207933976075"+1;Q[0]:= .418603966988037"+1;
             P[1]:=-.652268740837103"+1;Q[1]:=-.465669026080814"+1;
             P[2]:= .569955700306720   ;Q[2]:= .1"+1;
            Z:=XMXO/(X+XO);Z2:=Z*Z;
            T:=Z*POL(2,Z2,P)/POL(2,Z2,Q)
        "END";
        EI:=T+XMXO*R
    "END" "ELSE"
    "IF" X>-1 "THEN"
    "BEGIN" "REAL" Y;
             P[0]:=-4.41785471728217"+4;Q[0]:= 7.65373323337614"+4;
             P[1]:= 5.77217247139444"+4;Q[1]:= 3.25971881290275"+4;
             P[2]:= 9.93831388962037"+3;Q[2]:= 6.10610794245759"+3;
             P[3]:= 1.84211088668000"+3;Q[3]:= 6.35419418378382"+2;
             P[4]:= 1.01093806161906"+2;Q[4]:= 3.72298352833327"+1;
             P[5]:= 5.03416184097568   ;Q[5]:= 1;
        Y:=-X;
        EI:=LN(Y)-POL(5,Y,P)/POL(5,Y,Q)
    "END" "ELSE" "IF" X>-4 "THEN"
    "BEGIN" "REAL" Y;
             P[0]:= 8.67745954838444"-8;Q[0]:= 1;
             P[1]:= 9.99995519301390"-1;Q[1]:= 1.28481935379157"+1;
             P[2]:= 1.18483105554946"+1;Q[2]:= 5.64433569561803"+1;
             P[3]:= 4.55930644253390"+1;Q[3]:= 1.06645183769914"+2;
             P[4]:= 6.99279451291003"+1;Q[4]:= 8.97311097125290"+1;
             P[5]:= 4.25202034768841"+1;Q[5]:= 3.14971849170441"+1;
             P[6]:= 8.83671808803844   ;Q[6]:= 3.79559003762122   ;
             P[7]:= 4.01377664940665"-1;Q[7]:= 9.08804569188869"-2;
        Y:=-1/X;
        EI:=-EXP(X)*POL(7,Y,P)/POL(7,Y,Q)
    "END" "ELSE"
    "BEGIN" "REAL" Y;
             P[0]:=-9.99999999998447"-1;Q[0]:= 1;
             P[1]:=-2.66271060431811"+1;Q[1]:= 2.86271060422192"+1;
             P[2]:=-2.41055827097015"+2;Q[2]:= 2.92310039388533"+2;
             P[3]:=-8.95927957772937"+2;Q[3]:= 1.33278537748257"+3;
             P[4]:=-1.29885688746484"+3;Q[4]:= 2.77761949509163"+3;
             P[5]:=-5.45374158883133"+2;Q[5]:= 2.40401713225909"+3;
             P[6]:=-5.66575206533869   ;Q[6]:= 6.31657483280800"+2;
        Y:=-1/X;
        EI:=-EXP(X)*Y*(1+Y*POL(6,Y,P)/POL(6,Y,Q))
    "END"
"END" EI;
      "EOP"
```

```
    "CODE" 35081;
    "PROCEDURE" EIALPHA(X,N,ALPHA);
    "VALUE" X,N;"REAL" X;"INTEGER" N;"ARRAY" ALPHA;
    "BEGIN" "REAL" A,B,C;"INTEGER" K;
        C:=1/X;A:=EXP(-X);
        B:=ALPHA[0]:=A*C;
        "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
        ALPHA[K]:=B:=(A+K*B)*C
    "END" EIALPHA;
        "EOP"

"CODE" 35086;
    "PROCEDURE" ENX(X, N1, N2, A);
    "VALUE" X, N1, N2;
    "REAL" X; "INTEGER" N1, N2; "ARRAY" A;
    "IF" X<= 1.5 "THEN"
    "BEGIN"
        "REAL" "PROCEDURE" EI(X); "CODE" 35080;
        "REAL" W, E; "INTEGER" I;
        W:= -EI(-X);
        "IF" N1=1 "THEN" A[1]:=W;
        "IF" N2>1 "THEN" E:= EXP(-X);
        "FOR" I:=2 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN"
            W:= (E - X * W)/(I - 1);
            "IF" I>= N1 "THEN" A[I]:=W
        "END"
    "END" "ELSE"
    "BEGIN" "INTEGER" I, N; "REAL" W, E, AN;
        N:=ENTIER(X+.5);
        "IF" N<=10 "THEN"
        "BEGIN" "REAL" F, W1, T, H;
            "REAL" "ARRAY" P[2:19];
            P[ 2]:=.37534261820491"-1; P[11]:=.135335283236613   ;
            P[ 3]:=.89306465560228"-2; P[12]:=.497870683678639"-1;
            P[ 4]:=.24233983686581"-2; P[13]:=.183156388887342"-1;
            P[ 5]:=.70576069342458"-3; P[14]:=.673794699908547"-2;
            P[ 6]:=.21480277819013"-3; P[15]:=.247875217666636"-2;
            P[ 7]:=.67375807781018"-4; P[16]:=.911881965554516"-3;
            P[ 8]:=.21600730159975"-4; P[17]:=.335462627902512"-3;
            P[ 9]:=.70411579854292"-5; P[18]:=.123409804086680"-3;
            P[10]:=.23253026570282"-5; P[19]:=.453999297624848"-4;
                                                                "COMMENT"
```

```
            F:= W:= P[N];
            E:= P[N+9];
            W1:= T:= 1;
            H:= X-N;
            "FOR" I:=N-1, I-1 "WHILE" ABS(W1)>"-15 * W "DO"
            "BEGIN"
                F:= (E - I * F)/N;
                T:= -H * T / (N-I);
                W1:= T * F; W:= W + W1
             "END"
        "END" "ELSE"
        "BEGIN"
            "PROCEDURE" NONEXPENX(X, N1, N2, A); "CODE" 35087;
            "ARRAY" B[N:N];
            NONEXPENX(X, N, N, B);
            W:= B[N] * EXP(-X)
        "END";
        "IF" N1=N2 & N1=N "THEN" A[N]:=W "ELSE"
        "BEGIN"
            E:= EXP(-X);
            AN:=W;
            "IF" N<=N2 & N>=N1 "THEN" A[N]:=W;
            "FOR" I:= N-1 "STEP" -1 "UNTIL" N1 "DO"
            "BEGIN"
                W:= (E - I * W)/X;
                "IF" I<= N2 "THEN" A[I]:= W
            "END";
            W:=AN;
            "FOR" I:=N+1 "STEP" 1 "UNTIL" N2 "DO"
            "BEGIN"
                W:= (E - X * W)/(I - 1);
                "IF" I>=N1 "THEN" A[I]:=W
            "END"
        "END"
    "END" ENX;
        "EOP"
```

```
"CODE" 35087;
    "PROCEDURE" NONEXPENX(X, N1, N2, A);
    "VALUE" X, N1, N2;
    "REAL" X; "INTEGER" N1, N2; "ARRAY" A;
    "BEGIN" "INTEGER" I, N; "REAL" W, AN;
        N:= "IF" X<=1.5 "THEN" 1 "ELSE" ENTIER(X+.5);
        "IF" N<=10 "THEN"
        "BEGIN"
            "PROCEDURE" ENX(X, N1, N2, A); "CODE" 35086;
            "ARRAY" B[N:N];
            ENX(X, N, N, B);
            W:= B[N] * EXP(X)
        "END" "ELSE"
        "BEGIN"
            "INTEGER" K, K1;
            "REAL" UE, VE, WE, WE1, UO, VO, WO, WO1, R, S;
            UE:=1; VE:= WE:= 1/(X+N); WE1:=0;
            UO:=1; VO:= -N/(X * (X + N + 1)); WO1:= 1/X; WO:= VO + WO1;
            W:= (WE + WO)/2;
            K1:=1;
            "FOR" K:=K1 "WHILE" WO-WE>"-15 * W & WE>WE1 & WO<WO1 "DO"
            "BEGIN"
                WE1:= WE; WO1:=WO;
                R:= N+K; S:= R + X + K;
                UE:= 1/(1-K*(R-1)*UE/((S-2)*S));
                UO:= 1/(1-K*  R  *UO/( S * S-1));
                VE:= VE * (UE-1);
                VO:= VO * (UO-1);
                WE:= WE + VE;
                WO:= WO + VO;
                W:= (WE + WO)/2;
                K1:= K1 + 1
            "END"
        "END";
        AN:=W;
        "IF" N<=N2 & N>=N1 "THEN" A[N]:=W;
        "FOR" I:= N-1 "STEP" -1 "UNTIL" N1 "DO"
        "BEGIN"
            W:= (1 - I * W)/X;
            "IF" I<= N2 "THEN" A[I]:=W
        "END";
        W:=AN;
        "FOR" I:= N+1 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN"
            W:= (1 - X * W)/(I - 1);
            "IF" I>=N1 "THEN" A[I]:=W
        "END"
    "END" EXPENX;
        "EOP"
```

AUTHOR(S): H.FIOLET, N.TEMME.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 740317.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES:

THE PROCEDURE SINCOSINT CALCULATES THE SINE INTEGRAL SI(X) AND
THE COSINE INTEGRAL CI(X) DEFINED BY
    SI(X) = INTEGRAL FROM 0 TO X OF SIN(T)/T DT
AND
    CI(X) = GAMMA + LN(ABS(X)) +
            INTEGRAL FROM 0 TO X OF (COS(T)-1)/T DT,
WHERE GAMMA DENOTES EULER'S CONSTANT
(SEE [1] EQ. 5.2.1 AND 5.2.2);

THE AUXILIARY PROCEDURE SINCOSFG CALCULATES F(X) AND G(X)
DEFINED BY
    F(X) = CI(X) * SIN(X) - (SI(X) - PI / 2) * COS(X)
AND
    G(X) =-CI(X) * COS(X) - (SI(X) - PI / 2) * SIN(X);

FOR X=0 THE VALUES OF CI(X), F(X) AND G(X) ARE UNDEFINED;
THE FOLLOWING RELATIONS CONCERNING NEGATIVE X ARE VALID:
SI(-X) = -SI(X), CI(-X) = CI(X), F(-X) = -F(X), G(-X) = G(X).

KEYWORDS: SINE INTEGRAL,
COSINE INTEGRAL.

SUBSECTION: SINCOSINT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" SINCOSINT(X,SI,CI); "VALUE" X; "REAL" X, SI, CI;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

X : <ARITHMETIC EXPRESSION>;
    ENTRY: THE (REAL) ARGUMENT OF SI(X) AND CI(X);
SI: <VARIABLE>;
    EXIT: THE VALUE OF SI(X);
CI: <VARIABLE>;
    EXIT: THE VALUE OF CI(X).

PROCEDURES USED:

SINCOSFG = CP35385,
CHEPOLSER = CP31046.

RUNNING TIME:

"IF" ABS(X) <= 4 "THEN" ABOUT 3.8 MSEC
"ELSE" ABOUT 7.5 MSEC .

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE METHOD AND PERFORMANCE OF THE PROCEDURE SINCOSFG
(THIS SECTION).

SUBSECTION: SINCOSFG.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" SINCOSFG(X,F,G); "VALUE" X; "REAL" X, F, G;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF F(X) AND G(X);

F: <VARIABLE>;
EXIT: THE VALUE OF F(X);

G: <VARIABLE>;
EXIT: THE VALUE OF G(X).

PROCEDURES USED:

SINCOSINT = CP35084,
CHEPOLSER = CP31046.

RUNNING TIME:

"IF" ABS(X) <= 4 "THEN" ABOUT 4.7 MSEC
"ELSE" ABOUT 6.5 MSEC .

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

IF ABS(X) <= 4 THE SINE AND COSINE INTEGRALS ARE REPRESENTED BY TRUNCATED CHEBYSHEV SERIES. ON THIS INTERVAL THE FUNCTIONS F AND G ARE CALCULATED BY MEANS OF THE EQUATIONS GIVEN IN THE BRIEF DESCRIPTION.
IF ABS(X) > 4 THE FUNCTIONS F AND G ARE REPRESENTED BY TRUNCATED CHEBYSHEV SERIES. IN THIS CASE THE SINE AND COSINE INTEGRALS ARE COMPUTED BY MEANS OF THE FOLLOWING RELATIONS:

    SI(X) = PI / 2 - F(X) * COS(X) - G(X) * SIN(X)

AND

    CI(X) =   F(X) * SIN(X) - G(X) * COS(X).

THE FUNCTION VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-13.
WHEN USING THE PROCEDURE SINCOSINT FOR LARGE VALUES OF X , THE RELATIVE ACCURACY MAINLY DEPENDS ON THE ACCURACY OF THE FUNCTIONS SIN(X) AND COS(X).

REFERENCES:

[1].M.ABRAMOWITZ AND I.STEGUN (EDS.),1964.
HANDBOOK OF MATHEMATICAL FUNCTIONS WITH FORMULAS, GRAPHS AND MATHEMATICAL TABLES.
APPL. MATH. SER. 55, U.S.GOVT. PRINTING OFFICE,WASHINGTON, D.C.

[2].R.BULIRSCH.
NUMERICAL CALCULATION OF THE SINE, COSINE AND FRESNEL INTEGRALS HANDBOOK SERIES SPECIAL FUNCTIONS.
NUM. MATH. 9, 1967, PP380-385.

EXAMPLE OF USE:

IN THE FOLLOWING PROGRAM WE COMPUTE THE VALUES OF SI(X), CI(X), F(X) AND G(X) FOR X = 1;

```
"BEGIN"
    "PROCEDURE" SINCOSINT(X, SI, CI); "CODE" 35084;
    "PROCEDURE" SINCOSFG(X, F, G); "CODE" 35085;

    "REAL" SI, CI, F, G;

    SINCOSINT(1, SI, CI);
    SINCOSFG(1, F, G);

    OUTPUT(61,"("4B,"("SI(1)= ")",N,2B,"("CI(1)= ")",N/")",SI,CI);
    OUTPUT(61,"("4B,"(" F(1)= ")",N,2B,"(" G(1)= ")",N ")", F, G);
"END"
```

THIS PROGRAM DELIVERS:

```
SI(1)= +9.4608307036717"-001    CI(1)= +3.3740392290097"-001
 F(1)= -2.2725525318067"-001     G(1)= -9.7840157048430"-001
```

SOURCE TEXT(S):

```
"CODE" 35084;
    "PROCEDURE" SINCOSINT(X,SI,CI); "VALUE" X; "REAL" X,SI,CI;
    "BEGIN" "REAL" ABSX,Z,F,G;
        "PROCEDURE" SINCOSFG(X,F,G); "CODE" 35085;
        "REAL" "PROCEDURE" CHEPOLSER(N,X,A); "CODE" 31046;

        ABSX:= ABS(X);
        "IF" ABSX <= 4 "THEN"
        "BEGIN" "REAL" "ARRAY" A[0:10]; "REAL" Z2;
            A[0] :=+2.7368706803630"+00; A[1]:=-1.1106314107894"+00;
            A[2] :=+1.4176562194666"-01; A[3]:=-1.0252652579174"-02;
            A[4] :=+4.6494615619880"-04; A[5]:=-1.4361730896642"-05;
            A[6] :=+3.2093684948229"-07; A[7]:=-5.4251990770162"-09;
            A[8] :=+7.1776288639895"-11; A[9]:=-7.6335493723482"-13;
            A[10]:=+6.6679958346983"-15;
            Z:= X / 4; Z2:= Z * Z; G:= Z2 +Z2 - 1;
            SI:= Z * CHEPOLSER(10,G,A);
            A[0] :=+2.9659601400727"+00; A[1]:=-9.4297198341830"-01;
            A[2] :=+8.6110342738169"-02; A[3]:=-4.7776084547139"-03;
            A[4] :=+1.7529161205146"-04; A[5]:=-4.5448727803752"-06;
            A[6] :=+8.7515839180060"-08; A[7]:=-1.2998699938109"-09;
            A[8] :=+1.5338974898831"-11; A[9]:=-1.4724256070277"-13;
            A[10]:=+1.1721420798429"-15;
            CI:= .577215664901533 + LN(ABSX) - Z2 * CHEPOLSER(10,G,A)
        "END" "ELSE"
        "BEGIN" "REAL" CX,SX;
            SINCOSFG(X,F,G);
            CX:= COS(X); SX:= SIN(X);
            SI:= 1.570796326794897; "IF" X<0 "THEN" SI:= -SI;
            SI:= SI - F * CX - G * SX;
            CI:= F * SX - G * CX
        "END"
    "END" SINCOSINT;
        "EOP"
```

```
"CODE" 35085;
    "PROCEDURE" SINCOSFG(X,F,G); "VALUE" X; "REAL" X,F,G;
    "BEGIN" "REAL" ABSX,SI,CI;
        "PROCEDURE" SINCOSINT(X,SI,CI); "CODE" 35084;
        "REAL" "PROCEDURE" CHEPOLSER(N,X,A); "CODE" 31046;

        ABSX:= ABS(X);
        "IF" ABSX <= 4 "THEN"
        "BEGIN" "REAL" CX,SX;
           SINCOSINT(X,SI,CI);
           CX:= COS(X); SX:= SIN(X); SI:= SI - 1.570796326794897;
           F:= CI * SX - SI * CX;
           G:=-CI * CX - SI * SX
        "END" "ELSE"
        "BEGIN" "REAL" "ARRAY" A[0:23];
            A[0] :=+9.657882803518 5"-01; A[1] :=-4.306083777859 7"-02;
            A[2] :=-7.3143711748104"-03; A[3] :=+1.4705235789868"-03;
            A[4] :=-9.8657685732702"-05; A[5] :=-2.2743202204655"-05;
            A[6] :=+9.8240257322526"-06; A[7] :=-1.8973430148713"-06;
            A[8] :=+1.0063435941558"-07; A[9] :=+8.0819364822241"-08;
            A[10]:=-3.8976282875288"-08; A[11]:=+1.0335650325497"-08;
            A[12]:=-1.4104344875897"-09; A[13]:=-2.5232078399683"-10;
            A[14]:=+2.5699831325961"-10; A[15]:=-1.0597889253948"-10;
            A[16]:=+2.8970031570214"-11; A[17]:=-4.1023142563083"-12;
            A[18]:=-1.0437693730018"-12; A[19]:=+1.0994184520547"-12;
            A[20]:=-5.2214239401679"-13; A[21]:=+1.7469920787829"-13;
            A[22]:=-3.8470012979279"-14;
            F:= CHEPOLSER(22, 8/ABSX-1, A) / X;
            A[0] :=+2.2801220638241"-01; A[1] :=-2.6869727411097"-02;
            A[2] :=-3.5107157280958"-03; A[3] :=+1.2398008635186"-03;
            A[4] :=-1.5672945116862"-04; A[5] :=-1.0664141798094"-05;
            A[6] :=+1.1170629343574"-05; A[7] :=-3.1754011655614"-06;
            A[8] :=+4.4317473520398"-07; A[9] :=+5.5108696874463"-08;
            A[10]:=-5.9243078711743"-08; A[11]:=+2.2102573381555"-08;
            A[12]:=-5.0256827540623"-09; A[13]:=+3.1519168259424"-10;
            A[14]:=+3.6306990848979"-10; A[15]:=-2.2974764234591"-10;
            A[16]:=+8.5530309424048"-11; A[17]:=-2.1183067724443"-11;
            A[18]:=+1.7133662645092"-12; A[19]:=+1.7238877517248"-12;
            A[20]:=-1.2930281366811"-12; A[21]:=+5.7472339223731"-13;
            A[22]:=-1.8415468268314"-13; A[23]:=+3.5937256571434"-14;
            G:= 4 * CHEPOLSER(23, 8/ABSX-1, A) / ABSX /ABSX
        "END"
    "END" SINCOSFG;
        "EOP"
```

AUTHOR(S) : D. T. WINTER,N.M.TEMME.

INSTITUTE: MATHEMATICAL CENTRE

RECEIVED: 730727

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE FOLLOWING PROCEDURES:

RECIP GAMMA: THIS PROCEDURE CALCULATES THE RECIPROCAL OF THE GAMMA FUNCTION FOR ARGUMENTS IN THE RANGE [.5,1.5]; MOREOVER ODD AND EVEN PARTS ARE DELIVERED;

GAMMA: THIS PROCEDURE CALCULATES THE GAMMA FUNCTION;

LOG GAMMA: THIS PROCEDURE CALCULATES THE NATURAL LOGARITHM OF THE GAMMA FUNCTION FOR POSITIVE ARGUMENTS.

INCOMGAM : COMPUTES THE INCOMPLETE GAMMA FUNCTIONS CORRESPONDING TO THE DEFINITIONS 6.5.2 AND 6.5.3 IN REFERENCE [1].
THE COMPUTATIONS ARE BASED ON PADE-APPROXIMATIONS.

LET B(X,P,Q) = INTEGRAL FROM 0 TO X OF T**(P-1)*(1-T)**(Q-1)*DT,
P>0, Q>0, 0<=X<=1; B IS CALLED THE INCOMPLETE BETA FUNCTION.
LET I(X,P,Q) = B(X,P,Q)/B(1,P,Q); I IS CALLED THE INCOMPLETE BETA FUNCTION RATIO.

INCBETA : COMPUTES I(X,P,Q); 0<=X<=1, P>0, Q>0;
IBPPLUSN: COMPUTES I(X,P+N,Q) FOR N=0(1)NMAX, 0<=X<=1, P>0, Q>0;
IBQPLUSN: COMPUTES I(X,P,Q+N) FOR N=0(1)NMAX, 0<=X<=1, P>0, Q>0.
THE REMAINING FOUR PROCEDURES ARE AUXILIARY PROCEDURES
FOR INCBETA, IBPPLUSN AND IBQPLUSN.

KEYWORDS:

GAMMA-FUNCTION,
INCOMPLETE GAMMA-FUNCTION,
PADE-APPROXIMATION,
CONTINUED FRACTION,
INCOMPLETE BETA-FUNCTION,
INCOMPLETE BETA-FUNCTION RATIO.

SUBSECTION : RECIP GAMMA.

CALLING SEQUENCE:

```
    THE HEADING OF THIS PROCEDURE IS:
    "REAL" "PROCEDURE" RECIP GAMMA(X, ODD, EVEN);
    "VALUE" X; "REAL" X, ODD, EVEN;

    RECIP GAMMA:= 1/GAMMA(1-X).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    X:   <ARITHMETIC EXPRESSION>;
         THE ARGUMENT.  THIS ARGUMENT SHOULD SATISFY  -.5  <=  X  < = .5
         (ACTUALLY THE GAMMA FUNCTION IS CALCULATED FOR  1 - X,  I.E. IF
         ONE WANTS TO CALCULATE 1/GAMMA(1), ONE HAS TO SET X TO 0);
    ODD: <IDENTIFIER>;
         EXIT: THE ODD PART OF 1 / GAMMA(1 - X) DIVIDED BY (2 * X); I.E.
             (1 / GAMMA(1 - X) - 1 / GAMMA(1 + X)) / (2 * X);
    EVEN: <IDENTIFIER>;
         EXIT:  THE EVEN PART OF  1 / GAMMA(1 - X)  DIVIDED  BY  2; I.E.
             (1 / GAMMA(1 - X) + 1 / GAMMA(1 + X)) / 2;
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: AN ARRAY OF 12 ELEMENTS IS USED.

LANGUAGE: ALGOL-60.

METHOD AND PERFORMANCE:

THE RECIPROCAL OF THE GAMMA FUNCTION IS APPROXIMATED BY A TRUNCATED CHEBYSHEV SERIES. ODD AND EVEN PART ARE CALCULATED SEPARATELY. THE COEFFICIENTS OF THE CHEBYSHEV SERIES AS GIVEN IN THE PROCEDURE TEXT SHOULD GUARANTEE A PRECISION OF 14 DECIMAL DIGITS, HOWEVER AS THESE COEFFICIENTS CAN NOT BE READ IN FULL PRECISION UNDER CD-ALGOL VERSION 3, THIS PRECISION CAN NOT BE GUARANTEED. A PRECISION OF 13 DECIMAL DIGITS HOWEVER WILL BE OBTAINED. MOREOVER FOR THE ARGUMENT 1 (I.E. X = 0) EVEN AND RECIP GAMMA BOTH YIELD THE CORRECT VALUE.

EXAMPLE OF USE:

THE FOLLOWING PROGRAM:

```
"BEGIN" "REAL" X, ODD, EVEN;
    "REAL" "PROCEDURE" RECIP GAMMA(X, ODD, EVEN); "CODE" 35060;
    X:= RECIP GAMMA(.4, ODD, EVEN);
    OUTPUT(61, "(""("0.4")", 3(N), /")", X, ODD, EVEN);
    X:= RECIP GAMMA(0, ODD, EVEN);
    OUTPUT(61, "(""("0.0")", 3(N)")", X, ODD, EVEN)
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
0.4 +6.7150497244208"-001  -5.6944440692994"-001  +8.9928273521406"-001
0.0 +1.0000000000000"+000  -5.7721566490154"-001  +1.0000000000000"+000
```

SUBSECTION : GAMMA.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:

```
"REAL" "PROCEDURE" GAMMA(X); "VALUE" X; "REAL" X;
```

GAMMA:= THE VALUE OF THE GAMMA-FUNCTION AT X.

THE MEANING OF THE FORMAL PARAMETER IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. IF ONE OF THE FOLLOWING THREE CONDITIONS IS FULFILLED OVERFLOW WILL OCCUR:
1: THE ARGUMENT IS TOO LARGE (> 177);
2: THE ARGUMENT IS A NON-POSITIVE INTEGER;
3: THE ARGUMENT IS TOO 'CLOSE' TO A LARGE (IN ABSOLUTE VALUE) NON-POSITIVE INTEGER.

PROCEDURES USED:

RECIP GAMMA = CP35060
LOG GAMMA = CP35062.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: NO AUXILIARY ARRAY'S ARE DECLARED.

LANGUAGE: ALGOL-60.

METHOD AND PERFORMANCE:

WE DISTINGUISH BETWEEN THE FOLLOWING CASES FOR THE ARGUMENT X:

X < .5:
IN THIS CASE THE FORMULA GAMMA(X) * GAMMA(1-X) = PI / SIN(PI*X) IS USED. HOWEVER THE SINE FUNCTION IS NOT CALCULATED DIRECTLY ON THE ARGUMENT PI*X BUT ON THE ARGUMENT PI*(X MOD .5), IN THIS WAY A BIG DECREASE OF PRECISION IS AVOIDED. THE PRECISION HERE DEPENDS STRONGLY ON THE PRECISION OF THE SINE FUNCTION; HOWEVER A PRECISION BETTER THAN 12 DECIMAL DIGITS CAN BE EXPECTED IN THE GAMMA FUNCTION.

.5 <= X <= 1.5:
HERE THE PROCEDURE RECIP GAMMA IS CALLED. A PRECISION OF MORE THAN 13 DECIMAL DIGITS IS OBTAINED; MOREOVER: GAMMA(1) = 1.

1.5 < X <= 22:
THE RECURSION FORMULA GAMMA(1 + X) = X * GAMMA(X) IS USED. THE PRECISION DEPENDS ON THE NUMBER OF RECURSIONS NEEDED, A PRECISION BETTER THAN 10 DECIMAL DIGITS IS ALWAYS OBTAINED. THE UPPERBOUND OF 22 HAS BEEN CHOSEN, BECAUSE NOW IT IS ASSURED THAT FOR ALL INTEGER ARGUMENTS FOR WHICH THE VALUE OF THE GAMMA FUNCTION IS REPRESENTABLE (AND THIS IS THE CASE FOR ALL INTEGER ARGUMENTS IN THE RANGE [1,22]), THIS VALUE IS OBTAINED, I.E. GAMMA(I) = 1 * 2 * ... * (I - 1).

X > 22:
NOW THE PROCEDURES LOG GAMMA AND EXP ARE USED. THE PRECISION STRONGLY DEPENDS ON THE PRECISION OF THE EXPONENTIAL FUNCTION, AND NO BOUND FOR THE ERROR CAN BE GIVEN.

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" GAMMA(X); "CODE" 35061;
    "FOR" X:= -8.5, .25, 1.5, 22, 50 "DO"
    OUTPUT(61, "("+2Z.2D3B, N, /")", X, GAMMA(X))
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
 -8.50   -2.6335215159963"-005
  +.25   +3.6256099082219"+000
 +1.50   +8.8622692545276"-001
+22.00   +5.1090942171709"+019
+50.00   +6.0828186403422"+062
```

SUBSECTION : LOG GAMMA.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"REAL" "PROCEDURE" LOG GAMMA(X); "VALUE" X; "REAL" X;

LOG GAMMA:= THE NATURAL LOGARITHM OF THE GAMMA FUNCTION AT X.

THE MEANING OF THE FORMAL PARAMETER IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. THIS ARGUMENT MUST BE POSITIVE.

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: AN ARRAY OF 18 ELEMENTS IS USED.

LANGUAGE: ALGOL-60.

METHOD AND PERFORMANCE:
WE DISTIGUISH BETWEEN THE FOLLOWING CASES FOR THE ARGUMENT X (IN MOST CASES NOTHING IS SAID ABOUT PRECISION, AS THIS HIGHLY DEPENDS ON THE PRECISION OF THE NATURAL LOGARITHM; HOWEVER, A PRECISION BETTER THAN 11 DECIMAL DIGITS IS ALWAYS OBTAINED):

0 < X < 1:
HERE THE RECURSION FORMULA (LOG GAMMA(X)=LOG GAMMA(1+X)-LN(X)) IS USED.

1 <= X <= 2:
ON THIS INTERVAL THE TRUNCATED CHEBYSHEV SERIES FOR THE FUNCTION LOG GAMMA(X) / ((X-1)*(X-2)) IS USED. IN THIS WAY A PRECISION BETTER THAN 13 DECIMAL DIGITS IS ASSURED.

2 < X <= 13:
THE RECURSION FORMULA LOG GAMMA(X) = LOG GAMMA(1-X) + LN(X) IS USED.

13 < X <= 22:
AS FOR X < 1 THE FORMULA LOG GAMMA(X) = LOG GAMMA(1+X)-LN(X) IS USED.

X < 22:
IN THIS CASE LOG GAMMA IS CALCULATED BY USE OF THE ASYMPTOTIC EXPANSION FOR LOG GAMMA(X) - (X - .5) * LN(X) .

EXAMPLE OF USE:

THE FOLLOWING PROGRAM:

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" LOG GAMMA(X); "CODE" 35062;
    "FOR" X:= .25, 1.5, 12, 15, 80 "DO"
    OUTPUT(61, "("+2Z.2D3B, N, /")", X, LOG GAMMA(X))
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
  +.25   +1.2880225246981"+000
 +1.50   -1.2078223763524"-001
+12.00   +1.7502307845874"+001
+15.00   +2.5191221182739"+001
+80.00   +2.6929109765102"+002
```

SUBSECTION : INCOMGAM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" INCOMGAM(X,A,KLGAM,GRGAM,GAM,EPS);
"VALUE" X,A,EPS; "REAL" X,A,KLGAM,GRGAM,GAM,EPS;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE INDEPENDENT ARGUMENT X, X>=0;

A: <ARITHMETIC EXPRESSION>;
THE INDEPENDENT PARAMETER A, A>0;

KLGAM: <VARIABLE>;
EXIT: THE INTEGRAL FROM 0 TO X OF EXP(-T)*T**(A-1)*DT IS DELIVERED IN KLGAM;

GRGAM: <VARIABLE>;
EXIT: THE INTEGRAL FROM X TO INFINITY OF EXP(-T)* T**(A-1)*DT IS DELIVERED IN GRGAM;

GAM: <ARITHMETIC EXPRESSION>;
ENTRY: THE VALUE OF THE GAMMAFUNCTION WITH ARGUMENT A. FOR THIS EXPRESSION THE "REAL" "PROCEDURE" GAMMA(X); "CODE" 35061 MAY BE USED;

EPS: <ARITHMETIC EXPRESSION>;
ENTRY: THE DESIRED RELATIVE ACCURACY. THE VALUE OF EPS SHOULD NOT BE SMALLER THAN THE MACHINE ACCURACY, WHICH IS ABOUT "-14.

PROCEDURES USED: NONE.

RUNNING TIME: DEPENDS ON THE VALUES OF X,A,EPS.
FOR THE EXAMPLE BELOW THE EXECUTION TIME IS 0.003 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

FOR THE METHOD SEE REFERENCE [4]. THE RELATIVE ACCURACY OF THE RESULTS DEPENDS NOT ONLY ON THE QUANTITY EPS, BUT ALSO ON THE ACCURACY OF THE FUNCTIONS EXP AND GAMMA. ESPECIALLY FOR LARGE VALUES OF X AND A THE DESIRED ACCURACY CANNOT BE GUARANTEED.

REFERENCES:
SEE REFERENCES [1] AND [4] OF THE PROCEDURE IBQPLUSN(THIS SECTION).

EXAMPLE OF USE:

```
"BEGIN" "REAL" P,Q;
    "PROCEDURE" INCOMGAM(X,A,KLGAM,GRGAM,GAM,EPS);

"CODE" 35030;

    INCOMGAM(3,4,P,Q,1*2*3,2**(-48));
    "COMMENT" 1*2*3 = GAMMA(4);
    OUTPUT(61,"("/,"("KLGAM AND GRGAM ARE")",
    /,2(N)")",P,Q);
"END"
```

DELIVERS:

```
KLGAM AND GRGAM ARE
+2.1166086673066"+000   +3.8833913326934"+000.
```

SUBSECTION : INCBETA.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" INCBETA(X,P,Q,EPS);
 "VALUE" X,P,Q,EPS; "REAL" X,P,Q,EPS;
```

INCBETA DELIVERS THE VALUE OF I(X,P,Q);

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY 0<=X<=1;
P:      <ARITHMETIC EXPRESSION>;
        PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; P>0;
Q:      <ARITHMETIC EXPRESSION>;
        PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; Q>0;
EPS:    <ARITHMETIC EXPRESSION>;
        ENTRY: THE DESIRED RELATIVE ACCURACY; EPS SHOULD NOT BE
        SMALLER THAN THE MACHINE ACCURACY.
```

PROCEDURES USED: GAMMA = CP 35061.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: NO AUXILIARY ARRAYS ARE USED.

METHOD AND PERFORMANCE:

THE INCOMPLETE BETA FUNCTION I(X,P,Q) IS APPROXIMATED BY THE CONTINUED FRACTION CORRESPONDING TO FORMULA 26.5.8 IN REFERENCE[1]. IF X > .5 THE RELATION I(X,P,Q) = 1 - I(1-X,Q,P) IS USED. IT IS ADVISED TO USE IN INCBETA ONLY SMALL VALUES OF P AND Q, SAY 0 < P <= 5, 0 < Q <= 5. FOR OTHER RANGES OF THE PARAMETERS P AND Q THE PROCEDURES IBPPLUSN AND IBQPLUSN CAN BE USED.
INCBETA SATISFIES INCBETA = X IF X = 0 OR X = 1, WHATEVER P AND Q.
THERE IS NO CONTROL ON THE PARAMETERS X,P,Q FOR THEIR INTENDED RANGES.

REFERENCES: SEE REFERENCES [1], [2] AND [3] OF THE PROCEDURE IBQPLUSN (THIS SECTION).

EXAMPLE OF USE:

THE FOLLOWING PROGRAM:

```
"BEGIN" "REAL" "PROCEDURE" INCBETA(X,P,Q,EPS); "CODE" 35050;
OUTPUT(61,"("N")",INCBETA(.3,1.4,1.5,2**(-46)))
"END"
```

YIELDS THE FOLLOWING RESULT:

```
+2.7911593308577"-001.
```

SUBSECTION : IBPPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" IBPPLUSN(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"INTEGER" NMAX; "REAL" X,P,Q,EPS; "ARRAY" I;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY 0<=X<=1;
P:      <ARITHMETIC EXPRESSION>;
        PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; P>0.
        IT IS ADVISED TO TAKE 0<P<=1;
Q:      <ARITHMETIC EXPRESSION>;
        PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; Q>0;
NMAX:   <ARITHMETIC EXPRESSION>;
        NMAX INDICATES THE MAXIMUM NUMBER OF FUNCTION VALUES
        I(X,P+N,Q) TO BE GENERATED;
EPS:    <ARITHMETIC EXPRESSION>;
        ENTRY: THE DESIRED RELATIVE ACCURACY; EPS SHOULD NOT BE
        SMALLER THAN THE MACHINE ACCURACY;
I:      <ARRAY IDENTIFIER>;
        "ARRAY" I[0:NMAX]; NMAX>=0;
        EXIT: I[N] = I(X,P+N,Q) FOR N=0(1)NMAX.
```

PROCEDURES USED:

```
IXQFIX   = CP 35053;
IXPFIX   = CP 35054.
BOTH PROCEDURES IXQFIX AND IXPFIX CALL FOR
INCBETA  = CP 35050;
FORWARD  = CP 35055;
BACKWARD = CP 35056.
```

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: AN ARRAY OF NMAX + 1 ELEMENTS IS TO BE INSERTED BY THE USER. AN AUXILIARY ARRAY OF ENTIER(Q) + 1 ELEMENTS IS DECLARED IN THE AUXILIARY PROCEDURES.

METHOD AND PERFORMANCE:

SEE REFERENCE [2] AND [3]. IN [2] THE PROCEDURE IBPPLUSN IS CALLED INCOMPLETE BETA Q FIXED. THERE IS NO CONTROL ON THE PARAMETERS X,P,Q,NMAX FOR THEIR INTENDED RANGES.

REFERENCES: SEE REFERENCES [1], [2] AND [3] OF THE PROCEDURE IBQPLUSN (THIS SECTION).

EXAMPLE OF USE:

THE FOLLOWING PROGRAM:

```
"BEGIN" "REAL" "ARRAY" ISUBX[0:2];
    "PROCEDURE" IBPPLUSN(X,P,Q,NMAX,EPS,I); "CODE" 35051;
    IBPPLUSN(.3,.4,1.5,2,2**(-46),ISUBX);
    OUTPUT(61,"("3(N)")",ISUBX[0],ISUBX[1],ISUBX[2])
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
+7.2167087410147"-001 +2.7911593308576"-001 +9.8932849957944"-002.
```

SUBSECTION : IBQPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" IBQPLUSN(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"INTEGER" NMAX; "REAL" X,P,Q,EPS; "ARRAY" I;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY 0<=X<=1;

P: <ARITHMETIC EXPRESSION>;
PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; P>0;

Q: <ARITHMETIC EXPRESSION>;
PARAMETER: SEE DEFINITION IN BRIEF DESCRIPTION; Q>0;
IT IS ADVISED TO TAKE 0<Q<=1;

NMAX: <ARITHMETIC EXPRESSION>;
NMAX INDICATES THE MAXIMUM NUMBER OF FUNCTION VALUES
I(X,P,Q+N) TO BE GENERATED;

EPS: <ARITHMETIC EXPRESSION>;
ENTRY: THE DESIRED RELATIVE ACCURACY; EPS SHOULD NOT BE
SMALLER THAN THE MACHINE ACCURACY;

I: <ARRAY IDENTIFIER>;
"ARRAY" I[0:NMAX]; NMAX>=0;
EXIT: I[N] = I(X,P,Q+N) FOR N=0(1)NMAX.

PROCEDURES USED:

IXQFIX = CP 35053;
IXPFIX = CP 35054.
BOTH PROCEDURES IXQFIX AND IXPFIX CALL FOR
INCBETA = CP 35050;
FORWARD = CP 35055;
BACKWARD = CP 35056.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: AN ARRAY OF NMAX + 1 ELEMENTS IS TO BE INSERTED BY THE USER. AN AUXILIARY ARRAY OF ENTIER(P) + 1 ELEMENTS IS DECLARED IN THE AUXILIARY PROCEDURES.

METHOD AND PERFORMANCE:

SEE REFERENCE [2] AND [3]. IN [2] THE PROCEDURE IBQPLUSN IS CALLED INCOMPLETE BETA P FIXED. THERE IS NO CONTROL ON THE PARAMETERS X,P,Q,NMAX FOR THEIR INTENDED RANGES.

REFERENCES:

[1].M.ABRAMOWITZ AND I.A.STEGUN (ED.).
HANDBOOK OF MATHEMATICAL FUNCTIONS.
DOVER PUBLICATIONS, INC., NEW YORK, 1965.

[2].W.GAUTSCHI. COMM.A.C.M. 7, 1964, ALGORITHM 222, P 143.

[3].W.GAUTSCHI. SIAM REV. 9, 1967, PP 24-82.

[4].Y.L.LUKE. SIAM J. MATH. ANAL. VOL.1, 1971, PP. 266-281.

EXAMPLE OF USE:

THE FOLLOWING PROGRAM:

```
"BEGIN" "REAL" "ARRAY" ISUBX[0:2];
    "PROCEDURE" IBQPLUSN(X,P,Q,NMAX,EPS,I); "CODE" 35052;
    IBQPLUSN(.3,1.4,.5,2,2**(-46),ISUBX);
    OUTPUT(61,"("3(N)")",ISUBX[0],ISUBX[1],ISUBX[2])
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
+8.9449529793325"-002 +2.7911593308576"-001 +4.4728681067173"-001.
```

THE REMAINING PROCEDURES AND SUBSECTIONS ARE:

SUBSECTION : IXQFIX.
SUBSECTION : IXPFIX.
SUBSECTION : FORWARD.
SUBSECTION : BACKWARD.

THESE AUXILIARY PROCEDURES ARE NOT DESCRIBED HERE. MORE INFORMATION CAN BE FOUND IN REFERENCE [2], WHERE THE PROCEDURES FORWARD AND BACKWARD HAVE THE SAME NAME, WHILE IXQFIX AND IXPFIX ARE CALLED ISUBXQFIXED AND ISUBXPFIXED RESPECTIVELY. IN THE PROCEDURE BACKWARD WE CHANGED THE STARTING VALUE NU FOR THE BACKWARD RECURRENCE ALGORITHM. THE NEW VALUE OF NU IS MORE REALISTIC. ITS COMPUTATION IS BASED ON SOME ASYMPTOTIC ESTIMATIONS. ALSO THE INITIAL VALUE R=0 IS CHANGED INTO R=X.

SOURCE TEXT(S) :

```
"CODE" 35060;
"REAL" "PROCEDURE" RECIP GAMMA(X, ODD, EVEN);
"VALUE" X; "REAL" X, ODD, EVEN;
"BEGIN" "INTEGER" I;
    "REAL" ALFA, BETA, X2;
    "ARRAY" B[1:12];
    B[ 1]:= -.28387 65422 76024; B[ 2]:= -.07685 28408 44786;
    B[ 3]:= +.00170 63050 71096; B[ 4]:= +.00127 19271 36655;
    B[ 5]:= +.00007 63095 97586; B[ 6]:= -.00000 49717 36704;
    B[ 7]:= -.00000 08659 20800; B[ 8]:= -.00000 00331 26120;
    B[ 9]:= +.00000 00017 45136; B[10]:= +.00000 00002 42310;
    B[11]:= +.00000 00000 09161; B[12]:= -.00000 00000 00170;
    X2:= X * X * 8;
    ALFA:= -.00000 00000 00001; BETA:= 0;
    "FOR" I:= 12 "STEP" - 2 "UNTIL" 2 "DO"
    "BEGIN" BETA:= -(ALFA * 2 + BETA); ALFA:= - BETA * X2 - ALFA + B[I]
    "END";
    EVEN:= (BETA / 2 + ALFA) * X2 - ALFA + .92187 02936 50453;
    ALFA:= -.00000 00000 00034; BETA:= 0;
    "FOR" I:= 11 "STEP" - 2 "UNTIL" 1 "DO"
    "BEGIN" BETA:= -(ALFA * 2 + BETA); ALFA:= - BETA * X2 - ALFA + B[I]
    "END";
    ODD:= (ALFA + BETA) * 2;
    RECIP GAMMA:= ODD * X + EVEN
"END" RECIP GAMMA;
        "EOP"

"CODE" 35061;
"REAL" "PROCEDURE" GAMMA(X); "VALUE" X; "REAL" X;
"BEGIN" "REAL" Y, S, F, G, ODD, EVEN;
    "BOOLEAN" INV;
    "REAL" "PROCEDURE" RECIP GAMMA(X, ODD, EVEN);
    "VALUE" X; "REAL" X, ODD, EVEN;
 "CODE" 35060;
    "REAL" "PROCEDURE" LOG GAMMA(X); "VALUE" X; "REAL" X;
 "CODE" 35062;
    "IF" X < .5 "THEN"
    "BEGIN" Y:= X - ENTIER(X / 2) * 2; S:= 3.14159 26535 8979;
        "IF" Y >= 1 "THEN" "BEGIN" S:= - S; Y:= 2 - Y "END";
        "IF" Y >= .5 "THEN" Y:= 1 - Y; INV:= "TRUE"; X:= 1 - X;
        F:= S / SIN(3.14159 26535 8979 * Y)
    "END"
    "ELSE" INV:= "FALSE";
    "IF" X > 22 "THEN" G:= EXP(LOG GAMMA(X)) "ELSE"
    "BEGIN" S:= 1;
    NEXT: "IF" X > 1.5 "THEN"
        "BEGIN" X:= X - 1; S:= S * X; "GOTO" NEXT "END";
        G:= S / RECIP GAMMA(1 - X, ODD, EVEN)
    "END";
    GAMMA:= "IF" INV "THEN" F / G "ELSE" G
"END" GAMMA;
        "EOP"
```

```
"CODE" 35062;
"REAL" "PROCEDURE" LOG GAMMA(X); "VALUE" X; "REAL" X;
"IF" X > 13 "THEN"
"BEGIN" "REAL" R, X2;
    R:= 1;
NEXT: "IF" X <= 22 "THEN"
    "BEGIN" R:= R / X; X:= X + 1; "GOTO" NEXT "END";
    X2:= - 1 / (X * X); R:= LN(R);
    LOG GAMMA:= LN(X) * (X - .5) - X + R + .91893 85332 04672 +
    (((.59523 80952 38095"-3  * X2 + .79365 07936 50794"-3) * X2 +
       .27777 77777 77778"-2) * X2 + .83333 33333 33333"-1) / X
"END"
"ELSE"
"BEGIN" "REAL" Y, F, U0, U1, U, Z;
    "INTEGER" I;
    "ARRAY" B[1:18];
    F:= 1; U0:= U1:= 0;
    B[ 1]:= -.07611 41616 704358; B[ 2]:= +.00843 23249 659328;
    B[ 3]:= -.00107 94937 263286; B[ 4]:= +.00014 90074 800369;
    B[ 5]:= -.00002 15123 998886; B[ 6]:= +.00000 31979 329861;
    B[ 7]:= -.00000 04851 693012; B[ 8]:= +.00000 00747 148782;
    B[ 9]:= -.00000 00116 382967; B[10]:= +.00000 00018 294004;
    B[11]:= -.00000 00002 896918; B[12]:= +.00000 00000 461570;
    B[13]:= -.00000 00000 073928; B[14]:= +.00000 00000 011894;
    B[15]:= -.00000 00000 001921; B[16]:= +.00000 00000 000311;
    B[17]:= -.00000 00000 000051; B[18]:= +.00000 00000 000008;
    "IF" X < 1 "THEN"
    "BEGIN" F:= 1 / X; X:= X + 1 "END"
    "ELSE"
NEXT: "IF" X > 2 "THEN"
    "BEGIN" X:= X - 1; F:= F * X; "GOTO" NEXT "END";
    F:= LN(F); Y:= X + X - 3; Z:= Y + Y;
    "FOR" I:= 18 "STEP" - 1 "UNTIL" 1 "DO"
    "BEGIN" U:= U0; U0:= Z * U0 + B[I] - U1; U1:= U "END";
    LOG GAMMA:= (U0 * Y + .49141 53930 29387 - U1) * (X - 1) * (X - 2)
    + F
"END" LOG GAMMA;
        "EOP"
```

```
"CODE" 35030;
    "PROCEDURE" INCOMGAM(X,A,KLGAM,GRGAM,GAM,EPS);
    "VALUE" X,A,EPS; "REAL" X,A,KLGAM,GRGAM,GAM,EPS;
    "BEGIN" "REAL" C0,C1,C2,D0,D1,D2,X2,AX,P,Q,R,S,R1,R2,SCF;
        "INTEGER" N;
        S:= EXP(-X + A * LN(X)); SCF:= "+300;
        "IF" X <= ("IF" A < 3 "THEN" 1 "ELSE" A) "THEN"
        "BEGIN" X2:= X * X; AX:= A * X; D0:= 1; P:= A; C0:= S;
            D1:=(A+1)*(A+2-X); C1:=((A+1) * (A+2)+X) * S;
             R2:= C1/D1;
             "FOR" N:= 1, N+1 "WHILE" ABS((R2-R1)/R2) > EPS "DO"
                "BEGIN" P:= 2+P; Q:= (P+1) * (P*(P+2)-AX);
                R:= N * (N+A) * (P+2) * X2;
                C2:= (Q*C1 + R*C0)/P; D2:= (Q*D1 + R*D0)/P;
                 R1:=R2; R2:=C2/D2;
                C0:=C1; C1:=C2; D0:=D1; D1:=D2;
                "IF" ABS(C1) > SCF "OR" ABS(D1) > SCF "THEN"
               "BEGIN" C0:= C0/SCF; C1:= C1/SCF;
                       D0:= D0/SCF; D1:= D1/SCF
               "END"
            "END"; KLGAM:= R2/A; GRGAM:= GAM - KLGAM
        "END" "ELSE"
        "BEGIN" C0:=A*S; C1:=(1+X)* C0; Q:= X +2  - A;
            D0:= X; D1:= X * Q; R2:= C1/D1;
            "FOR" N:=1, N+1 "WHILE" ABS((R2-R1)/R2)>EPS "DO"
            "BEGIN"     Q:= 2 + Q; R:= N * (N+1-A);
                C2:= Q*C1-R*C0; D2:= Q*D1-R*D0;
                R1:=R2; R2:=C2/D2;
                C0:=C1; C1:=C2; D0:=D1; D1:=D2;
                "IF" ABS(C1) > SCF "OR" ABS(D1) > SCF "THEN"
               "BEGIN" C0:= C0/SCF; C1:= C1/SCF;
                       D0:= D0/SCF; D1:= D1/SCF
               "END"
            "END"; GRGAM:= R2/A; KLGAM:= GAM - GRGAM
        "END"
    "END" INCOMGAM;
        "EOP"
```

```
"CODE" 35050;
"REAL" "PROCEDURE" INCBETA(X,P,Q,EPS);
"VALUE" X,P,Q,EPS; "REAL" X,P,Q,EPS;
"BEGIN" "INTEGER" M,N; "REAL" G,F,FN,FN1,FN2,GN,GN1,GN2,DN,PQ;
    "BOOLEAN" N EVEN,RECUR;

    "REAL" "PROCEDURE" GAMMA(X); "VALUE" X; "REAL" X;
    "CODE"  35061;

    "IF" X=0 "OR" X=1 "THEN" INCBETA:= X "ELSE"
    "BEGIN" "IF" X>.5 "THEN"
        "BEGIN" F:= P; P:= Q; Q:= F; X:= 1-X; RECUR:= "TRUE""END"
        "ELSE" RECUR:= "FALSE";
        G:= FN2:= 0; M:= 0; PQ:= P+Q; F:= FN1:= GN1:= GN2:= 1;
        N EVEN:= "FALSE";
        "FOR" N:= 1,N+1 "WHILE" ABS((F-G)/F) > EPS "DO"
        "BEGIN" "IF" N EVEN "THEN"
            "BEGIN" M:= M+1; DN:= M*X*(Q-M)/(P+N-1)/(P+N) "END"
            "ELSE" DN:= -X*(P+M)*(PQ+M)/(P+N-1)/(P+N);
            G:= F; FN:= FN1+DN*FN2; GN:= GN1+DN*GN2;
            N EVEN:= ^ N EVEN; F:= FN/GN;
            FN2:= FN1; FN1:= FN; GN2:= GN1; GN1:= GN
        "END";
        F:= F*X**P*(1-X)**Q*GAMMA(P+Q)/GAMMA(P+1)/GAMMA(Q);
        "IF" RECUR "THEN" F:= 1-F;
        INCBETA:= F
    "END"
"END" INCBETA;
        "EOP"

"CODE" 35051;
"PROCEDURE" IBPPLUSN(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"INTEGER" NMAX; "REAL" X,P,Q,EPS; "ARRAY" I;
"BEGIN" "INTEGER" N;

    "PROCEDURE" IXQFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
    "REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
    "CODE" 35053;

    "PROCEDURE" IXPFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
    "REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
    "CODE" 35054;

    "IF" X=0 "OR" X=1 "THEN"
    "BEGIN" "FOR" N:= 0 "STEP" 1 "UNTIL" NMAX "DO" I[N]:= X "END"
    "ELSE"
    "BEGIN" "IF" X <=.5 "THEN" IXQFIX(X,P,Q,NMAX,EPS,I) "ELSE"
        "BEGIN" IXPFIX(1-X,Q,P,NMAX,EPS,I);
            "FOR" N:= 0 "STEP" 1 "UNTIL" NMAX "DO" I[N]:= 1-I[N]
        "END"
    "END"
"END" IBPPLUSN;
        "EOP"
```

```
"CODE" 35052;
"PROCEDURE" IBQPLUSN(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"INTEGER" NMAX; "REAL" X,P,Q,EPS; "ARRAY" I;
"BEGIN" "INTEGER" N;

    "PROCEDURE" IXQFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
    "REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
    "CODE" 35053;

    "PROCEDURE" IXPFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
    "REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
    "CODE" 35054;

    "IF" X=0 "OR" X=1 "THEN"
    "BEGIN" "FOR" N:= 0 "STEP" 1 "UNTIL" NMAX "DO" I[N]:= X "END"
    "ELSE"
    "BEGIN" "IF" X <=.5 "THEN" IXPFIX(X,P,Q,NMAX,EPS,I) "ELSE"
        "BEGIN" IXQFIX(1-X,Q,P,NMAX,EPS,I);
            "FOR" N:= 0 "STEP" 1 "UNTIL" NMAX "DO" I[N]:= 1-I[N]
        "END"
    "END"
"END" IBQPLUSN;
        "EOP"

"CODE" 35053;
"PROCEDURE" IXQFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
"BEGIN" "INTEGER" M,MMAX; "REAL" S,IQ0,IQ1,Q0;

    "REAL" "PROCEDURE" INCBETA(X,P,Q,EPS);
     "VALUE" X,P,Q,EPS; "REAL" X,P,Q,EPS;
    "CODE" 35050;

    "PROCEDURE" FORWARD(X,P,Q,I0,I1,NMAX,I);
    "VALUE" X,P,Q,I0,I1,NMAX; "INTEGER" NMAX; "REAL" X,P,Q,I0,I1;
    "ARRAY" I;
    "CODE" 35055;

    "PROCEDURE" BACKWARD(X,P,Q,I0,NMAX,EPS,I);
    "VALUE" X,P,Q,I0,NMAX,EPS; "INTEGER" NMAX; "REAL" X,P,Q,I0,EPS;
    "ARRAY" I;
    "CODE" 35056;

    M:= ENTIER(Q); S:= Q-M; Q0:= "IF" S>0 "THEN" S "ELSE" S+1;
    MMAX:= "IF" S>0 "THEN" M "ELSE" M-1;
    IQ0:= INCBETA(X,P,Q0,EPS);
    "IF" MMAX>0 "THEN" IQ1:= INCBETA(X,P,Q0+1,EPS);
    "BEGIN" "ARRAY" IQ[0:MMAX];
        FORWARD(X,P,Q0,IQ0,IQ1,MMAX,IQ);
        BACKWARD(X,P,Q,IQ[MMAX],NMAX,EPS,I)
    "END"
"END" IXQFIX;
        "EOP"
```

```
"CODE" 35054;
"PROCEDURE" IXPFIX(X,P,Q,NMAX,EPS,I); "VALUE" X,P,Q,NMAX,EPS;
"REAL" X,P,Q,EPS; "INTEGER" NMAX; "ARRAY" I;
"BEGIN" "INTEGER" M,MMAX; "REAL" S,PO,IO,I1,IQO,IQ1;

    "REAL" "PROCEDURE" INCBETA(X,P,Q,EPS);
     "VALUE" X,P,Q,EPS; "REAL" X,P,Q,EPS;
    "CODE" 35050;

    "PROCEDURE" FORWARD(X,P,Q,IO,I1,NMAX,I);
    "VALUE" X,P,Q,IO,I1,NMAX; "INTEGER" NMAX; "REAL" X,P,Q,IO,I1;
    "ARRAY" I;
    "CODE" 35055;

    "PROCEDURE" BACKWARD(X,P,Q,IO,NMAX,EPS,I);
    "VALUE" X,P,Q,IO,NMAX,EPS; "INTEGER" NMAX; "REAL" X,P,Q,IO,EPS;
    "ARRAY" I;
    "CODE" 35056;

    M:= ENTIER(P); S:= P-M; PO:= "IF" S>0 "THEN" S "ELSE" S+1;
    MMAX:= "IF" S>0 "THEN" M "ELSE" M-1;
    IO:= INCBETA(X,PO,Q,EPS);
    I1:= INCBETA(X,PO,Q+1,EPS);
    "BEGIN" "ARRAY" IP[O:MMAX];
        BACKWARD(X,PO,Q,IO,MMAX,EPS,IP); IQO:= IP[MMAX];
        BACKWARD(X,PO,Q+1,I1,MMAX,EPS,IP); IQ1:= IP[MMAX]
    "END";
    FORWARD(X,P,Q,IQO,IQ1,NMAX,I)
"END" IXPFIX;
        "EOP"

"CODE" 35055;
"PROCEDURE" FORWARD(X,P,Q,IO,I1,NMAX,I);
"VALUE" X,P,Q,IO,I1,NMAX; "INTEGER" NMAX; "REAL" X,P,Q,IO,I1;
"ARRAY" I;
"BEGIN" "INTEGER" M,N; "REAL" Y,R,S;
    I[O]:= IO; "IF" NMAX > 0 "THEN" I[1]:= I1;
    M:= NMAX-1; R:= P+Q-1; Y:= 1-X;
    "FOR" N:= 1 "STEP" 1 "UNTIL" M "DO"
    "BEGIN" S:= (N+R)*Y;
        I[N+1]:= ((N+Q+S)*I[N]-S*I[N-1])/(N+Q)
    "END"
"END" FORWARD;
        "EOP"
```

```
"CODE" 35036;
"PROCEDURE" BACKWARD(X,P,Q,I0,NMAX,EPS,I);
"VALUE" X,P,Q,I0,NMAX,EPS; "INTEGER" NMAX; "REAL" X,P,Q,I0,EPS;
"ARRAY" I;
"BEGIN" "INTEGER" M,N,NU; "REAL" R,PQ,Y,LOGX;
     "ARRAY" IAPPROX[0:NMAX];
     I[0]:= I0; "IF" NMAX>0 "THEN"
     "BEGIN""FOR" N:= 1 "STEP" 1 "UNTIL" NMAX "DO" IAPPROX[N]:= 0;
          PQ:= P+Q-1; LOGX:= LN(X);
          R:= NMAX+(LN(EPS)+Q*LN(NMAX))/LOGX;
          NU:= ENTIER(R-Q*LN(R)/LOGX);
L1:       N:= NU; R:= X;
L2:       Y:= (N+PQ)*X; R:= Y/(Y+(N+P)*(1-R));
          "IF" N<= NMAX "THEN" I[N]:= R; N:= N-1;
          "IF" N >= 1 "THEN" "GOTO" L2; R:= I0;
          "FOR" N:= 1 "STEP" 1 "UNTIL" NMAX "DO" R:= I[N]:= I[N]*R;
          "FOR" N:= 1 "STEP" 1 "UNTIL" NMAX "DO"
          "IF" ABS((I[N]-IAPPROX[N])/I[N]) > EPS "THEN"
          "BEGIN" "FOR" M:= 1 "STEP" 1 "UNTIL" NMAX "DO"
               IAPPROX[M]:= I[M]; NU:= NU+5; "GOTO" L1
          "END"
     "END"
"END" BACKWARD;
          "EOP"
```

AUTHOR: S.P.N. VAN KAMPEN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 740410.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES:

```
A) THE PROCEDURE ERRORFUNCTION COMPUTES THE ERROR FUNCTION AND
COMPLEMENTARY ERROR FUNCTION FOR A REAL ARGUMENT, I.E.
    ERF(X)  = 2 / SQRT(PI) * INTEGRAL FROM 0 TO X OF EXP(-T ** 2)DT
AND
    ERFC(X) = 2 / SQRT(PI) * INTEGRAL FROM X TO INFINITY OF
              EXP(-T ** 2)DT
            = 1 - ERF(X),
(SEE E.G. [1] EQ. 7.1.1 AND 7.1.2);
THESE FORMULAS ARE RELATED TO THE NORMAL OR GAUSSIAN PROBABILITY
FUNCTION:
    P(X)    = 1 / SQRT(2 * PI) * INTEGRAL FROM - INFINITY TO X OF
              EXP(-T ** 2 / 2)DT
            = (1 + ERF(X / SQRT(2))) / 2
AND
    Q(X)    = 1 / SQRT(2 * PI) * INTEGRAL FROM X TO INFINITY OF
              EXP(-T ** 2 / 2)DT
            = ERFC(X / SQRT(2)) / 2,
(SEE E.G. [1] EQ. 26.2.2, 26.2.3 AND 26.2.29).

B) THE AUXILIARY PROCEDURE NONEXPERFC COMPUTES
    EXP(X * X) * ERFC(X).

C) THE PROCEDURE INVERSE ERROR FUNCTION CALCULATES THE INVERSE OF
THE ERROR FUNCTION DEFINED BY:
    Y = INVERF(X),
WHERE
    X = ERF(Y) =
      = 2 / SQRT(PI) * INTEGRAL FROM 0 TO Y OF EXP(-T ** 2) DT,
(SEE THE PROCEDURE ERRORFUNCTION (THIS SECTION) ).

D) THE PROCEDURE FRESNEL CALCULATES THE FRESNEL INTEGRALS C(X) AND
S(X) DEFINED BY
    C(X) = INTEGRAL FROM 0 TO X OF COS(PI / 2 * T * T)DT
AND
    S(X) = INTEGRAL FROM 0 TO X OF SIN(PI / 2 * T * T)DT
(SEE [1] EQ. 7.3.1 AND 7.3.2);
```

```
    E) THE AUXILIARY PROCEDURE FG  CALCULATES F(X) AND G(X)  DEFINED BY
        F(X) = (0.5 - S(X))COS(PI / 2 * X * X) -
               (0.5 - C(X))SIN(PI / 2 * X * X)
    AND
        G(X) = (0.5 - C(X))COS(PI / 2 * X * X) +
               (0.5 - S(X))SIN(PI / 2 * X * X)
    (SEE [1] EQ. 7.3.5 AND 7.3.6).
```

KEYWORDS:
ERROR FUNCTION,
COMPLEMENTARY ERROR FUNCTION,
NORMAL PROBABILITY FUNCTION,
GAUSSIAN PROBABILITY FUNCTION,
FRESNEL INTEGRALS,
INVERSE ERROR FUNCTION.

SUBSECTION: ERRORFUNCTION.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" ERRORFUNCTION(X, ERF, ERFC);
"VALUE" X; "REAL" X, ERF, ERFC;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE (REAL) ARGUMENT OF ERF(X) AND ERFC(X);
ERF:    <VARIABLE>;
        EXIT: THE VALUE OF ERF(X),
ERFC:   <VARIABLE>;
        EXIT: THE VALUE OF ERFC(X).
```

PROCEDURES USED: NONEXPERFC = CP35022.

RUNNING TIME: ABOUT 0.001 100 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE METHOD AND PERFORMANCE OF NONEXPERFC (THIS SECTION).

SUBSECTION: NONEXPERFC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :
"REAL" "PROCEDURE" NONEXPERFC(X); "VALUE" X; "REAL" X;

NONEXPERFC DELIVERS THE VALUE OF EXP(X * X) * ERFC(X);

THE MEANING OF THE FORMAL PARAMETERS IS :
X: <ARITHMETIC EXPRESSION>;
ENTRY: THE (REAL) ARGUMENT OF NONEXPERFC.

PROCEDURES USED: ERRORFUNCTION = CP35021.

RUNNING TIME: ABOUT 0.000 900 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

IF ABS(X) <= 0.5 THE VALUES OF ERF(X) AND ERFC(X) ARE COMPUTED IN THE PROCEDURE ERRORFUNCTION BY MEANS OF RATIONAL CHEBYSHEV APPROXIMATION AS GIVEN IN [2]. ON THIS INTERVAL THE VALUE OF NONEXPERFC(X) = EXP(X * X) * ERFC(X) IS COMPUTED BY CALLING THE PROCEDURE ERRORFUNCTION.
IF ABS(X) > 0.5 THE VALUES OF ERF(X) AND ERFC(X) ARE COMPUTED BY CALLING THE PROCEDURE NONEXPERFC, WHILE THE VALUE OF NONEXPERFC(X) IS COMPUTED BY MEANS OF RATIONAL CHEBYSHEV APPROXIMATIONS AS GIVEN IN [2].
THE COMPUTED VALUES OF ERF(X) AND ERFC(X) ARE COMPARED WITH HIGHER PRECISION VALUES USING 4000 PSEUDO-RANDOM ARGUMENTS. IT APPEARED THAT ERF(X) IS COMPUTED WITH AN AVERAGE RELATIVE ERROR 1.93"-15 AND A MAXIMUM RELATIVE ERROR 1.35"-14.
IF X < 6 ERFC(X) IS COMPUTED WITH AN AVERAGE RELATIVE ERROR 8.87"-15 AND A MAXIMUM RELATIVE ERROR 1.55"-13.
IF X <= 26 ERFC(X) IS COMPUTED WITH AN AVERAGE RELATIVE ERROR 5.71"-14 AND A MAXIMUM RELATIVE ERROR 2.70"-12.
IF X > 26 ERFC(X)=0, BECAUSE IN THIS CASE ERFC(X) IS LESS THAN THE SMALLEST REPRESENTABLE POSITIVE NUMBER ON THE CD CYBER 73-28.
FOR THIS REASON IT IS ADVISABLE TO COMPUTE FOR X > 26 NONEXPERFC(X) INSTEAD OF ERFC(X).
IF X < -26.2 THE PROCEDURE NONEXPERFC WILL BE TERMINATED ABNORMALLY BY CAUSE OF OVERFLOW.

REFERENCES: SEE REFERENCES [1] AND [2] OF THE PROCEDURE FG (THIS SECTION).

EXAMPLE OF USE:

```
    WE COMPUTE  THE VALUES  OF
        ERF(1)  = 0.84270 07929 49714 8693,
        ERFC(1) = 0.15729 92070 50285 1307
    AND NONEXPERFC(100) =
        EXP(100 * 100) * ERFC(100) = 0.5416 13782 89843 2905"-2;

    "BEGIN"
        "PROCEDURE" ERRORFUNCTION(X, ERF, ERFC); "CODE" 35021;
        "REAL" "PROCEDURE" NONEXPERFC(X); "CODE" 35022;

        "REAL" ERF, ERFC, P;

        ERRORFUNCTION(1, ERF, ERFC);
        P:= NONEXPERFC(100);
        OUTPUT(61, "(""("    ERF(1)  = ")", +D.5DB5DB5D, /,
                    "("    ERFC(1) = ")", +D.5DB5DB5D, /,
                    "("    NONEXPERFC(100) = ")", +.5DB5DB5D"+D")",
                 ERF, ERFC, P);
    "END"

    THIS PROGRAM DELIVERS:

    ERF(1)  = +0.84270 07929 49710
    ERFC(1) = +0.15729 92070 50280
    NONEXPERFC(100) = +.56416 13782 98940"-2.
```

SUBSECTION : INVERSE ERROR FUNCTION.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE" INVERSE ERROR FUNCTION(X, ONEMINX, INVERF);
    "VALUE" X, ONEMINX; "REAL" X, ONEMINX, INVERF;


    THE MEANING OF THE FORMAL PARAMETERS IS :
    X:       <ARITHMETIC EXPRESSION>;
             ENTRY:
                 THE ARGUMENT OF THE FUNCTION INVERF;
                 IT IS NECESSARY THAT -1 < X < 1;
                 IF  ABS(X) > 0.8  THE  VALUE  OF X IS  NOT  USED IN THE
                 PROCEDURE;
    ONEMINX: <ARITHMETIC EXPRESSION>;
             ENTRY:
                 IF  ABS(X) <= 0.8  THE VALUE OF ONEMINX  IS NOT USED IN
                 THE PROCEDURE; IF  ABS(X) > 0.8  ONEMINX HAS TO CONTAIN
                 THE VALUE OF 1 - ABS(X); IN THE CASE  THAT ABS(X) IS IN
                 THE NEIGHBOURHOOD  OF 1, CANCELLATION  OF  DIGITS  TAKE
                 PLACE  IN THE  CALCULATION  OF 1 - ABS(X); IF THE VALUE
                 1-ABS(X) IS KNOWN EXACTLY  FROM ANOTHER SOURCE, ONEMINX
                 HAS TO  CONTAIN  THIS  VALUE, WHICH  WILL  GIVE  BETTER
                 RESULTS;
    INVERF:  <VARIABLE>;
             EXIT: THE RESULT OF THE PROCEDURE.
```

PROCEDURES USED: CHEPOLSER = CP31046.

RUNNING TIME: ABOUT 0.003 800 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE FUNCTION VALUE INVERF IS CALCULATED ON DIFFERENT INTERVALS BY MEANS OF CHEBYSHEV POLYNOMIALS, OF WHICH THE COEFFICIENTS ARE GIVEN IN [1].
ON THE COMPUTED RESULTS WE USED THE TESTS:

```
EPS1:= ABS(ERF(INVERF(X)) / X - 1),
EPS2:= ABS(INVERF(ERF(Y)) / Y - 1),
EPS3:= ABS((1 - ERF(INVERF(1 - X))) / X - 1).
```

IF ABS(X) < 0.9 UPPER BOUNDS FOR EPS1 AND EPS2 ARE 7.1"-15 AND 4.1"-14 RESP.
IF 0.9 < ABS(X) < 1 CANCELLATION OF DIGITS TAKE PLACE IN THE CALCULATION OF 1 - ABS(X). THIS CANCELLED DIGITS ARE ALSO LOST IN THE RESULT. IF THE VALUE OF 1 - ABS(X) IS KNOWN EXACTLY AND GIVEN IN ONEMINX , EPS1 AND EPS2 HAVE THE SAME UPPER BOUND AS BEFORE.
IF ABS(X) <= 0.99 AND THE VALUE OF 1 - ABS(X) IS KNOWN EXACTLY EPS3 <= 3.6"-14.
FOR "-300 <= 1 - ABS(X) < "-2 WE FOUND EPS3 <= 2.2"-11.

REFERENCES:

1. ANTHONY J. STRECOK.
ON THE CALCULATION OF THE INVERSE OF THE ERROR FUNCTION.
MATH. OF COMP., V. 22, 1968, PP144 - 158.

EXAMPLE OF USE:

IN THE FOLLOWING PROGRAM WE COMPUTE THE VALUES OF INVERF(0.6) AND INVERF(1 - "-150):

```
"BEGIN"
    "PROCEDURE" INVERSE ERROR FUNCTION(X, ONEMINX, INVERF);
    "CODE" 35023;

    "REAL" INVERF1, INVERF2;

    INVERSE ERROR FUNCTION(0.6, 0, INVERF1);
    INVERSE ERROR FUNCTION(1, "-150, INVERF2);

    OUTPUT(61,"(""("     X = ")", +D.D, "(" 1 - X = ")", +D.3D"+2ZD,
              "(" INVERF = ")", +.5DB5DB5D"+D, /")",
              0.6, 0.4, INVERF1);
    OUTPUT(61,"(""("     X = ")", +D.D, "(" 1 - X = ")", +D.3D"+2ZD,
              "(" INVERF = ")", +.5DB5DB5D"+D, /")",
              1 - "-150, "-150, INVERF2)
"END"
```

THIS PROGRAM DELIVERS:

```
X = +0.6 1 - X = +4.000"  -1 INVERF = +.59511 60814 50000"+0
X = +1.0 1 - X = +1.000"-150 INVERF = +.18490 44855 00090"+2
```

SUBSECTION: FRESNEL.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS :
    "PROCEDURE" FRESNEL(X, C, S); "VALUE" X; "REAL" X, C, S;

    THE MEANING OF THE FORMAL PARAMETERS IS :
    X:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE (REAL) ARGUMENT OF C(X) AND S(X);
    C:      <VARIABLE>;
            EXIT: THE VALUE OF C(X);
    S:      <VARIABLE>;
            EXIT: THE VALUE OF S(X).
```

PROCEDURES USED: FG = CP35028.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
SEE METHOD AND PERFORMANCE OF THE PROCEDURE FG (THIS SECTION).

REFERENCES :
SEE REF. [1] AND [3] OF THE PROCEDURE FG (THIS SECTION).

SUBSECTION: FG.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" FG(X, F, G); "VALUE" X; "REAL" X, F, G;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE (REAL) ARGUMENT OF F(X) AND G(X);
F:      <VARIABLE>;
        EXIT: THE VALUE OF F(X);
G:      <VARIABLE>;
        EXIT: THE VALUE OF G(X).
```

PROCEDURES USED: FRESNEL = CP35027.

RUNNING TIME: ABOUT 0.001 400 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

IF ABS(X) <= 1.6 THE FRESNEL INTEGRALS ARE COMPUTED WITH RATIONAL CHEBYSHEV APPROXIMATIONS AS GIVEN IN [3]. ON THIS INTERVAL THE FUNCTIONS F AND G ARE CALCULATED BY MEANS OF THE EQUATIONS GIVEN IN THE BRIEF DESCRIPTION.
IF ABS(X) > 1.6 THE FUNCTIONS F AND G ARE COMPUTED WITH RATIONAL CHEBYSHEV APPROXIMATIONS AS GIVEN IN [3]. IN THIS CASE THE FRESNEL INTEGRALS ARE COMPUTED BY MEANS OF

C(X) = 0.5 + F(X)SIN(PI / 2 * X * X) - G(X)COS(PI / 2 * X * X)

AND

S(X) = 0.5 - F(X)COS(PI / 2 * X * X) - G(X)SIN(PI / 2 * X * X).

IF X < 0 WE USE THE RELATIONS

C(-X) = -C(X), S(-X) = -S(X), F(-X) = -F(X) AND G(-X) = -G(X).

THE FUNCTION VALUES ARE COMPUTED WITH A RELATIVE PRECISION OF ABOUT "-14.

REFERENCES:

[1].M.ABRAMOWITZ AND I.A.STEGUN (ED.).
HANDBOOK OF MATHEMATICAL FUNCTIONS.
DOVER PUBLICATIONS, INC., NEW YORK, 1965.

[2].W.J.CODY.
RATIONAL CHEBYSHEV APPROXIMATIONS FOR THE ERROR FUNCTION.
MATH. COMP. V. 23, 1969, PP631-637.

[3].W.J.CODY.
CHEBYSHEV APPROXIMATIONS FOR THE FRESNEL INTEGRALS.
MATH. COMP. V. 22, 1968, PP450-453.

EXAMPLE OF USE:

IN THE FOLLOWING PROGRAM WE COMPUTE THE VALUES OF C(X), S(X), F(X) AND G(X) FOR X = 1;

```
"BEGIN"
    "PROCEDURE" FRESNEL(X, C, S); "CODE" 35027;
    "PROCEDURE" FG(X, F, G); "CODE" 35028;

    "REAL" C, S, F, G;

    FRESNEL(1, C, S);
    FG(1, F, G);

    OUTPUT(61, "("""("     C(1) = ")", +.5DB5D,
                "("        S(1) = ")", +.5DB5D, /")", C, S);
    OUTPUT(61, "("""("     F(1) = ")", +.5DB5D,
                "("        G(1) = ")", +.5DB5D")", F, G)

"END"
```

THIS PROGRAM DELIVERS:

```
C(1) = +.77989 34004        S(1) = +.43825 91474
F(1) = +.27989 34004        G(1) = +.06174 08526
```

SOURCE TEXT(S) :

```
"CODE" 35021;
    "PROCEDURE" ERRORFUNCTION(X, ERF, ERFC);
    "VALUE" X; "REAL" X, ERF, ERFC;
    "IF" X > 26   "THEN" "BEGIN" ERF:=  1; ERFC:= 0 "END" "ELSE"
    "IF" X < -5.5 "THEN" "BEGIN" ERF:= -1; ERFC:= 2 "END" "ELSE"
    "BEGIN" "REAL" ABSX, C, P, Q;
        "REAL" "PROCEDURE" NONEXPERFC(X); "CODE" 35022;

        ABSX:= ABS(X);
        "IF" ABSX <= 0.5 "THEN"
        "BEGIN" C:= X * X;          P:= ((-0.35609 84370 18154"-1  * C +
            0.69963 83488 61914"+1) * C + 0.21979 26161 82942"+2) * C +
            0.24266 79552 30532"+3;
            Q:= ((C +
            0.15082 79763 04078"+2) * C + 0.91164 90540 45149"+2) * C +
            0.21505 88758 69861"+3;
            ERF:=  X * P / Q; ERFC:= 1 - ERF
        "END" "ELSE"
        "BEGIN" ERFC:= EXP(-X * X) * NONEXPERFC(ABSX);
            ERF:= 1 - ERFC;
            "IF" X < 0 "THEN"
            "BEGIN" ERF:= -ERF; ERFC:= 2 - ERFC "END"
        "END"
    "END" ERRORFUNCTION;
         "EOP"

"CODE" 35023;
    "PROCEDURE" INVERSE ERROR FUNCTION(X, ONEMINX, INVERF);
    "VALUE" X, ONEMINX; "REAL" X, ONEMINX, INVERF;
    "BEGIN" "REAL" ABSX, P, BETAX;
        "REAL" "ARRAY" A[0 : 23];
        "REAL" "PROCEDURE" CHEPOLSER(N, X, A); "CODE" 31046;

        ABSX:= ABS(X);
        "IF" ABSX > 0.8 "AND" ONEMINX > 0.2 "THEN" ONEMINX:= 0;
        "IF" ABSX <= 0.8 "THEN"
        "BEGIN"
            A[ 0]:= 0.99288 53766 18941; A[ 1]:= 0.12046 75161 43104;
            A[ 2]:= 0.01607 81993 42100; A[ 3]:= 0.00268 67044 37162;
            A[ 4]:= 0.00049 96347 30236; A[ 5]:= 0.00009 88982 18599;
            A[ 6]:= 0.00002 03918 12764; A[ 7]:= 0.00000 43272 71618;
            A[ 8]:= 0.00000 09380 81413; A[ 9]:= 0.00000 02067 34720;
            A[10]:= 0.00000 00461 59699; A[11]:= 0.00000 00104 16680;
            A[12]:= 0.00000 00023 71501; A[13]:= 0.00000 00005 43928;
            A[14]:= 0.00000 00001 25549; A[15]:= 0.00000 00000 29138;
            A[16]:= 0.00000 00000 06795; A[17]:= 0.00000 00000 01591;
            A[18]:= 0.00000 00000 00374; A[19]:= 0.00000 00000 00088;
            A[20]:= 0.00000 00000 00021; A[21]:= 0.00000 00000 00005;
            INVERF:= CHEPOLSER(21, X * X / 0.32 - 1, A) * X
        "END" "ELSE"
        "IF" ONEMINX >= 25"-4 "THEN"
        "BEGIN"                                              "COMMENT"
```

```
        A[ 0]:=  0.91215 88034 17554; A[ 1]:= -0.01626 62818 67664;
        A[ 2]:=  0.00043 35564 72949; A[ 3]:=  0.00021 44385 70074;
        A[ 4]:=  0.00000 26257 51076; A[ 5]:= -0.00000 30210 91050;
        A[ 6]:= -0.00000 00124 06062; A[ 7]:=  0.00000 00624 06609;
        A[ 8]:= -0.00000 00005 40125; A[ 9]:= -0.00000 00014 23208;
        A[10]:=  0.00000 00000 34384; A[11]:=  0.00000 00000 33584;
        A[12]:= -0.00000 00000 01458; A[13]:= -0.00000 00000 00810;
        A[14]:=  0.00000 00000 00053; A[15]:=  0.00000 00000 00020;
        BETAX:= SQRT(- LN((1 + ABSX) * ONEMINX));
        P:= -1.54881 30423 7326 * BETAX + 2.56549 01231 4782;
        P:= CHEPOLSER(15, P, A);
        INVERF:= "IF" X < 0 "THEN" - BETAX * P "ELSE" BETAX * P
    "END" "ELSE"
    "IF" ONEMINX >= 5"-16 "THEN"
    "BEGIN"
        A[ 0]:=  0.95667 97090 20493; A[ 1]:= -0.02310 70043 09065;
        A[ 2]:= -0.00437 42360 97508; A[ 3]:= -0.00057 65034 22651;
        A[ 4]:= -0.00001 09610 22307; A[ 5]:=  0.00002 51085 47025;
        A[ 6]:=  0.00001 05623 36068; A[ 7]:=  0.00000 27544 12330;
        A[ 8]:=  0.00000 04324 84498; A[ 9]:= -0.00000 00205 30337;
        A[10]:= -0.00000 00438 91537; A[11]:= -0.00000 00176 84010;
        A[12]:= -0.00000 00039 91289; A[13]:= -0.00000 00001 86932;
        A[14]:=  0.00000 00002 72923; A[15]:=  0.00000 00001 32817;
        A[16]:=  0.00000 00000 31834; A[17]:=  0.00000 00000 01670;
        A[18]:= -0.00000 00000 02036; A[19]:= -0.00000 00000 00965;
        A[20]:= -0.00000 00000 00220; A[21]:= -0.00000 00000 00010;
        A[22]:=  0.00000 00000 00014; A[23]:=  0.00000 00000 00006;
        BETAX:= SQRT(- LN((1 + ABSX) * ONEMINX));
        P:= -0.55945 76313 29832 * BETAX + 2.28791 57162 6336;
        P:= CHEPOLSER(23, P, A);
        INVERF:= "IF" X < 0 "THEN" - BETAX * P "ELSE" BETAX * P
    "END" "ELSE" "IF" ONEMINX >= 3.13 "-294 "THEN"
    "BEGIN"
        A[ 0]:=  0.98857 50640 66189; A[ 1]:=  0.01085 77051 84599;
        A[ 2]:= -0.00175 11651 02763; A[ 3]:=  0.00002 11969 93207;
        A[ 4]:=  0.00001 56648 71404; A[ 5]:= -0.00000 05190 41687;
        A[ 6]:= -0.00000 00371 35790; A[ 7]:=  0.00000 00012 17431;
        A[ 8]:= -0.00000 00001 76812; A[ 9]:= -0.00000 00000 11937;
        A[10]:=  0.00000 00000 00380; A[11]:= -0.00000 00000 00066;
        A[12]:= -0.00000 00000 00009;
        BETAX:= SQRT(- LN((1 + ABSX) * ONEMINX));
        P:= -9.19999 23588 3015 / SQRT(BETAX) + 2.79499 08201 2460;
        P:= CHEPOLSER(12, P, A);
        INVERF:= "IF" X < 0 "THEN" - BETAX * P "ELSE" BETAX * P
    "END" "ELSE" INVERF:= SIGN(X) * 26
"END" INVERSE ERROR FUNCTION;
    "EOP"
```

```
"CODE" 35022;
    "REAL" "PROCEDURE" NONEXPERFC(X); "VALUE" X; "REAL" X;
    "BEGIN" "REAL" ABSX, ERF, ERFC, C, P, Q;
        "PROCEDURE" ERRORFUNCTION(X, ERF, ERFC); "CODE" 35021;

        ABSX:= ABS(X);
        "IF" ABSX <= 0.5 "THEN"
        "BEGIN" ERRORFUNCTION(X, ERF, ERFC);
            NONEXPERFC:= EXP(X * X) * ERFC
        "END" "ELSE"
        "IF" ABSX < 4 "THEN"
        "BEGIN" C:= ABSX;      P:= ((((((-0.13686 48573 82717"-6  * C +
            0.56419 55174 78974"+0) * C + 0.72117 58250 88309"+1) * C +
            0.43162 22722 20567"+2) * C + 0.15298 92850 46940"+3) * C +
            0.33932 08167 34344"+3) * C + 0.45191 89537 11873"+3) * C +
            0.30045 92610 20162"+3;
            Q:= ((((((C +
            0.12782 72731 96294"+2) * C + 0.77000 15293 52295"+2) * C +
            0.27758 54447 43988"+3) * C + 0.63898 02644 65631"+3) * C +
            0.93135 40948 50610"+3) * C + 0.79095 09253 27898"+3) * C +
            0.30045 92609 56983"+3;
            NONEXPERFC:= "IF" X > 0 "THEN" P / Q "ELSE"
                         EXP(X * X) * 2 - P / Q
        "END" "ELSE"
        "BEGIN" C:= 1 / X / X;      P:= (((0.22319 24597 34185"-1  * C +
            0.27866 13086 09648"-0) * C + 0.22695 65935 39687"-0) * C +
            0.49473 09106 23251"-1) * C + 0.29961 07077 03542"-2;
            Q:= (((C +
            0.19873 32018 17135"+1) * C + 0.10516 75107 06793"+1) * C +
            0.19130 89261 07830"+0) * C + 0.10620 92305 28468"-1;
            C:= (C * (-P) / Q + 0.56418 95835 47756) / ABSX;
            NONEXPERFC:= "IF" X > 0 "THEN" C "ELSE" EXP(X * X) * 2 - C
        "END"
    "END" NONEXPERFC;
         "EOP"

"CODE" 35027;
    "PROCEDURE" FRESNEL(X, C, S); "VALUE" X; "REAL" X, C, S;
    "BEGIN" "REAL" ABSX, X3, X4, A, P, Q, F, G, C1, S1;
        "PROCEDURE" FG(X, F, G); "CODE" 35028;
        ABSX:= ABS(X);
        "IF" ABSX <= 1.2 "THEN"
        "BEGIN" A:= X * X; X3:= A * X; X4:= A * A;
            P:= (((5.47711 38568 2687"-6  * X4 - 5.28079 65137 2623"-4)
            * X4 + 1.76193 95254 3491"-2) * X4 - 1.99460 89882 6184"-1)
            * X4 + 1;
            Q:= (((1.18938 90142 2876"-7  * X4 + 1.55237 88527 6994"-5)
            * X4 + 1.09957 21502 5642"-3) * X4 + 4.72792 11201 0453"-2)
            * X4 + 1;
            C:= X * P / Q;
            P:= (((6.71748 46662 5141"-7  * X4 - 8.45557 28435 2777"-5)
            * X4 + 3.87782 12346 3683"-3) * X4 - 7.07489 91514 4523"-2)
            * X4 + 5.23598 77559 8299"-1;                        "COMMENT"
```

```
            Q:= (((5.95281 22767 8410"-8  * X4 + 9.62690 87593 9034"-6)
            * X4 + 8.17091 94215 2134"-4) * X4 + 4.11223 15114 2384"-2)
            * X4 + 1;
            S:= X3 * P / Q
        "END" "ELSE"
        "IF" ABSX <= 1.6 "THEN"
        "BEGIN" A:= X * X; X3:= A * X; X4:= A * A;
           P:=((((-5.68293 31012 1871"-8  * X4 + 1.02365 43505 6106"-5)
            * X4 - 6.71376 03469 4922"-4) * X4 + 1.91870 27943 1747"-2)
            * X4 - 2.07073 36033 5324"-1) * X4 + 1.00000 00000 0111"+0;
            Q:=((((4.41701 37406 5010"-10 * X4 + 8.77945 37789 2369"-8)
            * X4 + 1.01344 63086 6749"-5) * X4 + 7.88905 24505 2360"-4)
            * X4 + 3.96667 49695 2323"-2) * X4 + 1;
            C:= X * P / Q;
           P:=((((-5.76765 81559 3089"-9  * X4 + 1.28531 04374 2725"-6)
            * X4 - 1.09540 02391 1435"-4) * X4 + 4.30730 52650 4367"-3)
            * X4 - 7.37766 91401 0191"-2) * X4 + 5.23598 77559 8344"-1;
            Q:=((((2.05539 12445 8580"-10 * X4 + 5.03090 58124 6612"-8)
            * X4 + 6.87086 26571 8620"-6) * X4 + 6.18224 62019 5473"-4)
            * X4 + 3.53398 34276 7472"-2) * X4 + 1;
            S:= X3 * P / Q
        "END" "ELSE"
        "IF" ABSX < "15 "THEN"
        "BEGIN" FG(X, F, G);
            A:= X * X;
            A:= (A - ENTIER(A / 4) * 4) * 1.57079 63267 9490;
            C1:= COS(A); S1:= SIN(A);
            A:= "IF" X < 0 "THEN" -0.5 "ELSE" 0.5;
            C:=  F * S1 - G * C1 + A;
            S:= -F * C1 - G * S1 + A
        "END" "ELSE" C:= S:= SIGN(X) * 0.5
    "END" FRESNEL;
        "EOP"

"CODE" 35028;
    "PROCEDURE" FG(X, F, G); "VALUE" X; "REAL" X, F, G;
    "BEGIN" "REAL" ABSX, C, S, C1, S1, A, XINV, X3INV, C4, P, Q;
        "PROCEDURE" FRESNEL(X, C, S); "CODE" 35027;

        ABSX:= ABS(X);
        "IF" ABSX <= 1.6 "THEN"
        "BEGIN" FRESNEL(X, C, S);
            A:= X * X * 1.57079 63267 9490; C1:= COS(A); S1:= SIN(A);
            A:= "IF" X < 0 "THEN" -0.5 "ELSE" 0.5;
            P:= A - C; Q:= A - S;
            F:= Q * C1 - P * S1;
            G:= P * C1 + Q * S1
        "END" "ELSE"
        "IF" ABSX <= 1.9 "THEN"
        "BEGIN" XINV:= 1 / X; A:= XINV * XINV;
            X3INV:= A * XINV; C4:= A * A;                     "COMMENT"
```

```
        P:= (((1.35304 23554 0388"+1  * C4 + 6.98534 26160 1021"+1)
        * C4 + 4.80340 65557 7925"+1) * C4 + 8.03588 12280 3942"+0)
        * C4 + 3.18309 26850 4906"-1;
        Q:= (((6.55630 64008 3916"+1  * C4 + 2.49561 99380 5172"+2)
        * C4 + 1.57611 00558 0123"+2) * C4 + 2.55491 61843 5795"+1)
        * C4 + 1;
        F:= XINV * P / Q;
        P:=((((2.05421 43249 8501"+1  * C4 + 1.96232 03797 1663"+2)
        * C4 + 1.99182 81867 8903"+2) * C4 + 5.31122 81348 0989"+1)
        * C4 + 4.44533 82755 0512"+0) * C4 + 1.01320 61881 0275"-1;
        Q:=((((1.01379 48339 6003"+3  * C4 + 3.48112 14785 6545"+3)
        * C4 + 2.54473 13318 1822"+3) * C4 + 5.83590 57571 6429"+2)
        * C4 + 4.53925 01967 3689"+1) * C4 + 1;
        G:= X3INV * P / Q
    "END" "ELSE"
    "IF" ABSX <= 2.4 "THEN"
    "BEGIN" XINV:= 1 / X; A:= XINV * XINV;
        X3INV:= A * XINV; C4:= A * A;
        P:=((((7.17703 24936 5140"+2  * C4 + 3.09145 16157 4430"+3)
        * C4 + 1.93007 64078 6716"+3) * C4 + 3.39837 13492 6984"+2)
        * C4 + 1.95883 94102 1969"+1) * C4 + 3.18309 88182 2017"-1;
        Q:=((((3.36121 69918 0551"+3  * C4 + 1.09334 24898 8809"+4)
        * C4 + 6.33747 15585 1144"+3) * C4 + 1.08535 06750 0650"+3)
        * C4 + 6.18427 13817 2887"+1) * C4 + 1;
        F:= XINV * P / Q;
        P:=((((3.13330 16306 8756"+2  * C4 + 1.59268 00608 5354"+3)
        * C4 + 9.08311 74952 9594"+2) * C4 + 1.40959 61791 1316"+2)
        * C4 + 7.11205 00178 9783"+0) * C4 + 1.01321 16176 1805"-1;
        Q:=((((1.15149 83237 6261"+4  * C4 + 2.41315 56721 3370"+4)
        * C4 + 1.06729 67803 0581"+4) * C4 + 1.49051 92279 7329"+3)
        * C4 + 7.17128 59693 9302"+1) * C4 + 1;
        G:= X3INV * P / Q
    "END" "ELSE"
    "BEGIN" XINV:= 1 / X; A:= XINV * XINV;
        X3INV:= A * XINV; C4:= A * A;
        P:=((((2.61294 75322 5142"+4  * C4 + 6.13547 11361 4700"+4)
        * C4 + 1.34922 02817 1857"+4) * C4 + 8.16343 40178 4375"+2)
        * C4 + 1.64797 71284 1246"+1) * C4 + 9.67546 03296 7090"-2;
        Q:=((((1.37012 36481 7226"+6  * C4 + 1.00105 47890 0791"+6)
        * C4 + 1.65946 46262 1853"+5) * C4 + 9.01827 59623 1524"+3)
        * C4 + 1.73871 69067 3649"+2) * C4 + 1;
        F:= (C4 * (-P) / Q + 0.31830 98861 83791) * XINV;
       P:=(((((1.72590 22465 4837"+6  * C4 + 6.66907 06166 8636"+6)
        * C4 + 1.77758 95083 8030"+6) * C4 + 1.35678 86781 3756"+5)
        * C4 + 3.87754 14174 6378"+3) * C4 + 4.31710 15782 3358"+1)
        * C4 + 1.53989 73381 9769"-1;
       Q:=(((((1.40622 44112 3580"+8  * C4 + 9.38695 86253 1635"+7)
        * C4 + 1.62095 60050 0232"+7) * C4 + 1.02878 69305 6688"+6)
        * C4 + 2.69183 18039 6243"+4) * C4 + 2.86733 19497 5899"+2)
        * C4 + 1;
        G:= (C4 * (-P) / Q + 0.10132 11836 42338) * X3INV
    "END"
"END" FG;
    "EOP"
```

AUTHORS: M. BAKKER AND N.M. TEMME.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 780601.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE FOLLOWING PROCEDURES:

BESS J0;
COMPUTES THE ORDINARY BESSEL FUNCTION OF THE FIRST KIND OF ORDER ZERO WITH ARGUMENT X;

BESS J1;
COMPUTES THE ORDINARY BESSEL FUNCTION OF THE FIRST KIND OF ORDER ONE WITH ARGUMENT X;

BESS J;
GENERATES AN ARRAY OF ORDINARY BESSEL FUNCTIONS OF THE FIRST KIND OF ORDER L (L = 0,...,N) WITH ARGUMENT X;

BESS Y01;
COMPUTES THE ORDINARY BESSEL FUNCTIONS OF THE SECOND KIND OF ORDERS ZERO AND ONE WITH ARGUMENT X; X > 0;

BESS Y;
GENERATES AN ARRAY OF ORDINARY BESSEL FUNCTIONS OF THE SECOND KIND OF ORDER L (L = 0,...N) WITH ARGUMENT X; X> 0;

BESS PQ0;
THIS PROCEDURE IS AN AUXILIARY PROCEDURE FOR THE COMPUTATION OF THE ORDINARY BESSEL FUNCTIONS OF ORDER ZERO FOR LARGE VALUES OF THEIR ARGUMENT;

BESS PQ1;
THIS PROCEDURE IS AN AUXILIARY PROCEDURE FOR THE COMPUTATION OF THE ORDINARY BESSEL FUNCTIONS OF ORDER ONE FOR LARGE VALUES OF THEIR ARGUMENT.

KEYWORDS: BESSEL FUNCTION,
ORDINARY BESSEL FUNCTION OF THE FIRST KIND,
ORDINARY BESSEL FUNCTION OF THE SECOND KIND.

REFERENCES:

[1] ABRAMOWITZ, M., AND STEGUN, I. (EDS),
HANDBOOK OF MATHEMATICAL FUNCTIONS WITH FORMULAS, GRAPHS AND MATHEMATICAL TABLES. APPL. MATH. SER. 55, U.S. GOVT. PRINTING OFFICE, WASHINGTON, D.C. (1964).

[2] C.W. CLENSHAW,
CHEBYSHEV SERIES FOR MATHEMATICAL FUNCTIONS, NAT. PHYS. LAB. MATH. TABLES, VOL. 5, HER MAJESTY'S STATIONARY OFFICE, LONDON (1962).

[3] W. GAUTSCHI,
COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS, SIAM REVIEW, VOL. 9, 24-82 (1967).

SUBSECTION: BESS JO.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" BESS JO(X); "VALUE" X; "REAL" X;
"CODE" 35160;
```

BESS JO DELIVERS THE ORDINARY BESSEL FUNCTION OF THE FIRST KIND OF ORDER ZERO WITH ARGUMENT X;

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

BESS PQO = CP 35165.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR ABS(X) < 8: LESS THAN 3 MS,
FOR ABS(X) >= 8: LESS THAN 5 MS, ON THE CYBER 73/28.

METHOD AND PERFORMANCE:

CHEBYSHEV SERIES FROM [2].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" BESS JO(X); "CODE" 35160;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D "-ZD")",
    X, BESS JO(X))
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
1        .76519768655794" 0
```

SUBSECTION: BESS J1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"REAL" "PROCEDURE" BESS J1(X); "VALUE" X; "REAL" X;
"CODE" 35161;
```

BESS J1 DELIVERS THE ORDINARY BESSEL FUNCTION OF THE FIRST KIND OF ORDER ONE WITH ARGUMENT X;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

BESS PQ1 = CP 35166.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR ABS(X) < 8: LESS THAN 3 MS,
FOR ABS(X) >= 8: LESS THAN 5 MS, ON THE CYBER 73/28.

METHOD AND PERFORMANCE:

CHEBYSHEV SERIES FROM [2].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" BESS J1(X); "CODE" 35161;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D "-ZD")",
    X, BESS J1(X)))
"END"
```

DELIVERS THE FOLLOWING RESULTS:

```
1     .44005058574492" 0
```

SUBSECTION: BESS J.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" BESS J(X,N,J); "VALUE" X,N;
"INTEGER" N; "REAL" X; "ARRAY" J;
"CODE" 35162;

THE MEANING OF THE FORMAL PARAMETERS IS:
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF ARRAY J; N >= 0;
J:  <ARRAY IDENTIFIER>;
    "ARRAY" J[0:N];
    EXIT: J[L] IS THE ORDINARY BESSEL FUNCTION OF THE FIRST KIND OF
          ORDER L AND ARGUMENT X.
```

PROCEDURES USED: START = CP 35185;

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

RUNNING TIME:
ROUGHLY PROPORTIONAL TO THE MAXIMUM OF 1.359 * X + 72 AND N + 18.

METHOD AND PERFORMANCE: MILLER'S ALGORITHM, SEE [3].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X; "ARRAY" J[0:1];
    "REAL" "PROCEDURE" BESS J0(X); "CODE" 35160;
    "REAL" "PROCEDURE" BESS J1(X); "CODE" 35161;
    "PROCEDURE" BESS J(X,N,J,);     "CODE" 35162;
    "FOR" X:= 1,5,10,25 "DO"
    "BEGIN" BESS J(X,1,J);
        OUTPUT(61,"("ZZ.D, 2(BB-.D"-ZD),/")",
        X, J[0] - BESS J0(X),J[1] - BESS J1(X))
    "END"
"END"
```

DELIVERS THE FOLLOWING RESULTS:

```
 1.0   .2"-13   .2"-13
 5.0  -.8"-14  -.4"-14
10.0  -.4"-14   .4"-14
25.0  -.1"-14  -.9"-15
```

SUBSECTION: BESS Y01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" BESS Y01(X,YO,Y1); "VALUE" X; "REAL" X,YO,Y1;
"CODE" 35163;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;
YO: <VARIABLE>;
    EXIT: YO HAS THE VALUE OF THE ORDINARY BESSEL FUNCTION OF THE
          SECOND KIND OF ORDER 0 AND ARGUMENT X;
Y1: <VARIABLE>;
    EXIT: Y1 HAS THE VALUE OF THE ORDINARY BESSEL FUNCTION OF THE
          SECOND KIND OF ORDER 1 AND ARGUMENT X.
```

PROCEDURES USED:

```
BESS J0  = CP 35160,
BESS J1  = CP 35161,
BESS PQ0 = CP 35165,
BESS PQ1 = CP 35166.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

RUNNING TIME:

ABOUT 15 MS, ON THE CYBER 73/28.

METHOD AND PERFORMANCE:

CHEBYSHEV SERIES FROM [2].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X,YO,Y1;
    "PROCEDURE" BESS Y01(X,YO,Y1); "CODE" 35163;
    X:= 1; BESS Y01(X,YO,Y1);
    OUTPUT(61,"("/,4BD.D,2(B-.14D"-ZD)")",X,YO,Y1)
"END"
```

DELIVERS THE FOLLOWING RESULTS:

1.0 988256964215676" -1 -.78121282130028" 0

SUBSECTION: BESS Y.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS Y(X,N,Y); "VALUE" X,N;
"INTEGER" N; "REAL" X; "ARRAY" Y;
"CODE" 35164;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X > 0;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY Y; N >= 0;
Y:  <ARRAY IDENTIFIER>;
    "ARRAY" Y[0:N];
    EXIT: Y[I] IS THE VALUE OF THE ORDINARY BESSEL FUNCTION OF THE
          SECOND KIND OF ORDER I (I = 0,...,N) AND ARGUMENT X.
```

PROCEDURES USED:

BESS Y01 = CP 35163.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

RUNNING TIME:

DEPENDS ON N; SEE BESS Y01.

METHOD AND PERFORMANCE:

Y[0] AND Y[1] ARE COMPUTED BY USING BESS Y01 (CP 35163); THE REMAINING Y[I] ARE COMPUTED BY USING THE RECURRENCE RELATION
Y[I+1]:= Y[I] * 2 * I/X - Y[I-1], I >= 1.

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "ARRAY" Y[0:2];
    "PROCEDURE" BESS Y(X,N,Y); "CODE" 35164;
    BESS Y(1,2,Y);
    OUTPUT(61,"("3(-.14D"-ZD)")", Y[0], Y[1], Y[2])
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
8.8256964215676"- 2 -7.8121282130028"- 1 -1.6506826068162"  0
```

SUBSECTION: BESS PQ0.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS PQ0(X,P0,Q0); "VALUE" X; "REAL" X,P0,Q0;
"CODE" 35165;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X > 0;

P0: <VARIABLE>;
EXIT: P0 CORRESPONDS WITH THE FUNCTION P(X,0) DEFINED IN [1,FORMULAS 9.2.5 AND 9.2.6];

Q0: <VARIABLE>;
EXIT: Q0 CORRESPONDS WITH THE FUNCTION Q(X,0) DEFINED IN [1,FORMULAS 9.2.5 AND 9.2.6].

PROCEDURES USED:

BESS J0 = CP 35160,
BESS Y01 = CP 35163.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

ABOUT 15 MS, ON THE CYBER 73/28.

METHOD AND PERFORMANCE:

FOR X >= 8 CHEBYSHEV SERIES FROM [2],
FOR X < 8 WITH BESS J0 AND BESS Y01.

EXAMPLE OF USE:

SEE SUBSECTION BESS PQ1.

SUBSECTION: BESS PQ1.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" BESS PQ1(X,P1,Q1); "VALUE" X; "REAL" X,P1,Q1;
"CODE" 35166;

THE MEANING OF THE FORMAL PARAMETERS IS:
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X > 0;
P1: <VARIABLE>;
    EXIT: P1  CORRESPONDS WITH THE FUNCTION P(X,1) DEFINED
          IN [1,FORMULAS 9.2.5 AND 9.2.6];
Q1: <VARIABLE>;
    EXIT: Q1  CORRESPONDS WITH THE FUNCTION Q(X,1) DEFINED
          IN [1,FORMULAS 9.2.5 AND 9.2.6].
```

PROCEDURES USED:

BESS J1 = CP 35161,
BESS Y01 = CP 35163.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

ABOUT 15 MS, ON THE CYBER 73/28.

METHOD AND PERFORMANCE:

FOR X >= 8 CHEBYSHEV SERIES FROM [2],
FOR X < 8 WITH BESS J1 AND BESS Y01.

EXAMPLE OF USE:

FROM THE WRONSKIAN RELATION [1,9.1.16] IT CAN BE SHOWN THAT PO * P1 + QO * Q1 = 1, WHATEVER X. IN THE FOLLOWING PROGRAM WE VERIFY THIS RELATION.

```
"BEGIN" "REAL" X,P,Q,R,S;
    "PROCEDURE" BESS PQO(X,PO,QO); "CODE" 35165;
    "PROCEDURE" BESS PQ1(X,P1,Q1); "CODE" 35166;
    "FOR" X:= 1,3,5,10 "DO"
    "BEGIN" BESSPQO(X,P,Q); BESSPQ1(X,R,S);
        OUTPUT(61,"("BB,D.2D"+3D")", ABS(P*R + Q*S -1))
    "END"
"END"
```

THE RESULTS ARE:

4.97"-014 4.26"-014 5.68"-014 7.11"-015

SOURCE TEXT(S):

```
"CODE" 35160;
    "REAL" "PROCEDURE" BESS JO(X); "VALUE" X; "REAL" X;
    "IF" X=O "THEN" BESS JO:= 1 "ELSE"
    "IF" ABS(X) <  8 "THEN"
    "BEGIN" "REAL" Z, Z2, AR, BO, B1, B2;
        X:= X/8; Z:= 2*X*X - 1; Z2:= Z + Z;
        B1:= B2:= O;
        "FOR" AR:=-.75885"-15, +.4125321         "-13,
        -.194383469        "-11, +.7848696314     "-10,
        -.267925353056     "- 8, +.7608163592419  "- 7,
        -.176194690776215"- 5, +.324603288210051"- 4,
        -.46062616620628 "- 3, +.48191800694676 "- 2,
        -.34893769411409 "- 1, +.158067102332097    ,
        -.37009499387265 "- 0, +.265178613203337    ,
        -.8723442352852 22"- 2 "DO"
        "BEGIN" BO:= Z2*B1-B2+AR;
          B2:= B1; B1:= BO
        "END";
      BESS JO:= Z*B1 - B2 + .15772 79714 7489
    "END" "ELSE"
    "BEGIN" "REAL" C, COSX, SINX, PO, QO;
      "PROCEDURE" BESS PQO(X, PO, QO); "CODE" 35165;
      X:= ABS(X); C:= .79788 45608 02865 / SQRT(X);
      COSX:= COS(X-.70685 83470 57703" 1);
      SINX:= SIN(X-.70685 83470 57703" 1);
      BESS PQO(X, PO, QO);
      BESSJO:= C * (PO * COSX - QO * SINX)
    "END" BESS JO;
        "EOP"
```

```
"CODE" 35161;
    "REAL" "PROCEDURE" BESS J1(X); "VALUE" X; "REAL" X;
    "IF" X=0 "THEN" BESS J1:= 0 "ELSE"
    "IF" ABS(X) < 8 "THEN"
    "BEGIN" "REAL" Z, Z2, AR, B0, B1, B2;
      X:= X/8; Z:= 2*X*X - 1; Z2:= Z + Z;
        "COMMENT" COMPUTATION OF J1;
        B1:= B2:= 0;
        "FOR" AR:=
        -.19554            "-15, +.1138572          "-13,
        -.57774042         "-12, +.2528123664       "-10,
        -.94242129816      "- 9, +.29497070072 78   "- 7,
        -.76175878054003 "- 6, +.1588 70192399321"- 4,
        -.260444389348581"- 3, +.324027018268386"- 2,
        -.291755248061542"- 1, +.177709117239728"- 0,
        -.661443934134543"- 0, +.128799409885768"+ 1,
        -.119180116054122"+ 1 "DO"
        "BEGIN" B0:= Z2*B1-B2+AR;
          B2:= B1; B1:= B0
        "END";
      BESS J1:= X * (Z * B1 - B2 + .64835 87706 05265)
    "END" "ELSE"
    "BEGIN" "REAL" C, COSX, SINX, P1, Q1; "INTEGER" SGNX;
      "PROCEDURE" BESS PQ1(X, P1, Q1); "CODE" 35166;
      SGNX:= SIGN(X); X:= ABS(X);
      C := .79788 45608 02865 / SQRT(X);
      COSX:= COS(X-.70685 83470 57703"+1);
      SINX:= SIN(X-.70685 83470 57703"+1);
      BESS PQ1(X, P1, Q1);
      BESS J1:= SGNX * C * (P1*SINX + Q1*COSX)
    "END" BESS J1;
          "EOP"

"CODE" 35162;
    "PROCEDURE" BESS J(X, N, J); "VALUE" X, N;
    "REAL"X; "INTEGER" N; "ARRAY" J;
    "IF" X=0 "THEN"
    "BEGIN" J[0]:= 1;
        "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" J[N]:= 0
    "END" "ELSE"
```

"COMMENT"

;

```
    "BEGIN""REAL" X2, R, S; "INTEGER" L, M, NU, SIGNX;
        "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
        SIGNX:= SIGN(X); X:= ABS(X);
        R:= S:= 0; X2:= 2/X; L:= 0; NU:= START(X,N,0);
        "FOR" M:= NU    "STEP" -1 "UNTIL" 1 "DO"
        "BEGIN" R:= 1/(X2*M-R);
            L:= 2-L; S:= R*(L+S);
            "IF" M<=N "THEN" J[M]:= R
        "END";
        J[0]:= R:= 1/(1+S);
        "FOR" M:= 1 "STEP" 1 "UNTIL" N "DO"
        J[M]:= R:= R*J[M];
        "IF" SIGNX < 0 "THEN"
        "FOR" M:= 1 "STEP" 2 "UNTIL" N "DO"
        J[M]:= -J[M];
    "END" BESSELJ;
         "EOP"


"CODE" 35163;
    "PROCEDURE" BESS YO1(X, YO, Y1); "VALUE" X; "REAL" X, YO, Y1;
    "IF" X< 8 "THEN"
    "BEGIN" "REAL" Z, Z2, C, LNX, AR, BO, B1, B2;
      "REAL" "PROCEDURE" JO(X); "CODE" 35160;
      "REAL" "PROCEDURE" J1(X); "CODE" 35161;
      C := .63661 97723 67581; LNX := C * LN(X);
      C := C/X; X:= X/8; Z:= 2*X*X - 1; Z2:= Z + Z;
        "COMMENT" COMPUTATION OF YO;
        B1:= B2:= 0;
        "FOR" AR:= +.164349 "-14,
        -.3747341          "-13, +.402633082        "-11,
        -.15837552542      "- 9, +.524879478733     "- 8,
        -.14407233274019 "- 6, +.32065325376548 "- 5,
        -.56320791410569 9"- 4, +.753113593257774"- 3,
        -.72879624795521 "- 2, +.471966895957634"- 1,
        -.177302012781143"- 0, +.261567346255047"- 0,
        +.179034314077182"- 0, -.274474305529745"DO"
        "BEGIN" BO:= Z2*B1-B2+AR;
          B2:= B1; B1:= BO
        "END";
      YO:= LNX * JO(8*X)+Z*B1-B2-.33146 11320 3285"-1;
        "COMMENT" COMPUTATION OF Y1;
        B1:= B2:= 0;
```

"COMMENT"

```
        "FOR" AR:=
        +.42773           "-15, -.2440949          "-13,
        +.121143321       "-11, -.5172121473       "-10,
        +.187547032473    "- 8, -.5688440039919    "- 7,
        +.141662436449235"- 5, -.283046401495148"- 4,
        +.440478629867099"- 3, -.51316411610611 "- 2,
        +.423191803533369"- 1, -.226624991556755"- 0,
        +.675615780772188"- 0, -.767296362886646"- 0,
        -.128697384381350"- 0"DO"
        "BEGIN" B0:= Z2*B1-B2+AR;
          B2:= B1; B1:= B0
        "END";
      Y1:= LNX * J1(X*8)-C + X * (Z*B1-B2+.20304 10588 593425"-1)
    "END" "ELSE"
    "BEGIN" "REAL" C, COSX, SINX, P0, Q0, P1, Q1;
      "PROCEDURE" BESS PQ0(X, P0, Q0); "CODE" 35165;
      "PROCEDURE" BESS PQ1(X, P1, Q1); "CODE" 35166;
      C:= .79788 45608 02865 / SQRT(X);
      BESS PQ0(X, P0, Q0); BESS PQ1(X, P1, Q1);
      X:= X-.70685 83470 57703"1; COSX:= COS(X); SINX:= SIN(X);
      Y0:= C * (P0*SINX + Q0*COSX);
      Y1:= C * (Q1*SINX - P1*COSX)
    "END" BESS Y01;
        "EOP"


"CODE" 35164;
    "PROCEDURE" BESS Y(X, N, Y); "VALUE" X, N;
    "REAL" X; "INTEGER" N; "ARRAY" Y;
    "BEGIN" "INTEGER" I; "REAL" Y0, Y1, Y2;
      "PROCEDURE" BESS Y01(X, Y0, Y1); "CODE" 35163;
      BESS Y01(X, Y0, Y1); Y[0]:= Y0;
      "IF" N > 0 "THEN" Y[1]:= Y1 ;
      X:= 2/X;
      "FOR" I:=2 "STEP" 1 "UNTIL" N "DO"
      "BEGIN" Y[I]:= Y2:= (I-1)*X*Y1 - Y0;
        Y0:= Y1; Y1:= Y2
      "END"
    "END" BESS Y;
        "EOP"
```

```
"CODE" 35165;
    "PROCEDURE" BESS PQ0(X, P0, Q0);
    "VALUE" X; "REAL" X, P0, Q0;
    "IF" X < 8 "THEN"
    "BEGIN" "REAL" B, COSX, SINX, J0X, Y0;
      "REAL" "PROCEDURE" J0(X); "CODE" 35160;
      "PROCEDURE" BESS Y01(X, Y0, Y1); "CODE" 35163;
      B:= SQRT(X) * 1.2533 14137 31550;
      BESS Y01(X, Y0, J0X); J0X:= J0(X);
      X:= X-.78539 81633 97448; COSX:= COS(X); SINX:= SIN(X);
      P0:= B * (Y0 * SINX + J0X * COSX);
      Q0:= B * (Y0 * COSX - J0X * SINX)
    "END" "ELSE"
    "BEGIN" "REAL" X2, AR, B0, B1, B2, Y;
            Y:= 8/X; X:= 2*Y*Y-1; X2:= X+X; B1:= B2:= 0;
            "FOR" AR:=
            -.10012             "-15, +.67481            "-15,
            -.506903            "-14, +.4326596          "-13,
            -.43045789          "-12, +.516826239        "-11,
            -.7864091377        "-10, +.163064646352     "- 8,
            -.51705945376 06    "- 7, +.3075184787 5195  "- 5,
            -.536522046813212"- 3 "DO"
            "BEGIN" B0:= X2 * B1 - B2 + AR;
               B2:= B1; B1:= B0
            "END";
            P0:= X * B1 - B2 + .99946034934752;
            "COMMENT" COMPUTATION OF Q0;
            B1:= B2:= 0;
            "FOR" AR:=
            -.60999             "-15, +.425523           "-14,
            -.3336328           "-13, +.30061451         "-12,
            -.320674742         "-11, +.4220121905       "-10,
            -.72719159369       "- 9, +.1797245724797    "- 7,
            -.74144984110606 "- 6, +.683851994261165"- 4
            "DO"
            "BEGIN" B0:= X2 * B1 - B2 + AR;
                B2:= B1; B1:= B0
            "END";
            Q0:=(X * B1 - B2 -.015555854605337) * Y
    "END" BESS PQ0;
        "EOP"
```

```
"CODE" 35166;
    "PROCEDURE" BESS PQ1(X, P1, Q1);
    "VALUE" X; "REAL" X, P1, Q1;
    "IF" X < 8 "THEN"
    "BEGIN" "REAL" B, COSX, SINX, J1X, Y1;
      "REAL" "PROCEDURE" J1(X); "CODE" 35161;
      "PROCEDURE" BESS Y01(X, Y0, Y1); "CODE" 35163;
      B:= SQRT(X) * 1.253314137 31550;
      BESS Y01(X, J1X, Y1); J1X:= J1(X);
      X:= X-.78539 81633 97448; COSX:= COS(X); SINX:= SIN(X);
      P1:= B * (J1X * SINX - Y1 * COSX);
      Q1:= B * (J1X * COSX + Y1 * SINX)
    "END" "ELSE"
    "BEGIN" "REAL" X2, AR, B0, B1, B2, Y;
        Y:= 8 / X; X:= 2 * Y * Y - 1; X2 := X + X;
            "COMMENT" COMPUTATION OF P1;
            B1:= B2:= 0;
            "FOR" AR:= +.10668"-15,
            -.72212           "-15, +.545267          "-14,
            -.4684224         "-13, +.46991955        "-12,
            -.570486364       "-11, +.881689866       "-10,
            -.187189074911    "- 8, +.6177633960644   "- 7,
            -.398728430048 89 "- 5, +.89898983308594  "- 3
            "DO"
            "BEGIN" B0:= B1 * X2 - B2 + AR;
                B2:= B1; B1:= B0
            "END";
            P1:= X * B1 - B2 + 1.0009030408600137;
            "COMMENT" COMPUTATION OF Q1;
            B1:= B2:= 0;
            "FOR" AR:=
            -.10269           "-15, +.65083           "-15,
            -.456125          "-14, +.3596777         "-13,
            -.32643157        "-12, +.351521879       "-11,
            -.4686363688      "-10, +.82291933277     "- 9,
            -.2095978138408   "- 7, +.91386152579555  "- 6,
            -.96277235491571  "- 4 "DO"
            "BEGIN" B0:= X2 * B1 - B2 + AR;
                B2:= B1; B1:= B0
            "END";
            Q1:=(X * B1 - B2 + .46777787069535" -1) * Y
    "END" BESS PQ1;
        "EOP"
```

AUTHORS: M. BAKKER AND N.M. TEMME.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 750201.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE FOLLOWING PROCEDURES:

BESS I0;
COMPUTES THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF ORDER ZERO WITH ARGUMENT X;

BESS I1;
COMPUTES THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF ORDER ONE WITH ARGUMENT X;

BESS I;
GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS OF THE FIRST KIND OF ORDER L (L = 0, ..., N) WITH ARGUMENT X;

BESS K01;
COMPUTES THE MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDERS ZERO AND ONE WITH ARGUMENT X; X > 0;

BESS K;
GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDER L (L = 0, ..., N) WITH ARGUMENT X; X > 0;

NONEXP BESS I0;
DOES THE SAME AS BESS I0, BUT THE RESULT IS MULTIPLIED BY EXP(-ABS(X));

NONEXP BESS I1;
DOES THE SAME AS BESS I1, BUT THE RESULT IS MULTIPLIED BY EXP(-ABS(X));

NONEXP BESS I;
DOES THE SAME AS BESS I, BUT THE ARRAY ELEMENTS ARE MULTIPLIED BY EXP(-ABS(X));

NONEXP BESS K01;
DOES THE SAME AS BESS K01, BUT THE RESULTS ARE MULTIPLIED BY EXP(X);

NONEXP BESS K;
DOES THE SAME AS BESS K, BUT THE ARRAY ELEMENTS ARE MULTIPLIED BY EXP(X).

KEYWORDS: BESSEL FUNCTIONS,
MODIFIED BESSEL FUNCTIONS,
INTEGER ORDER.

REFERENCES:

[1] M.ABRAMOWITZ AND I.A. STEGUN,
HANDBOOK OF MATHEMATICAL FUNCTIONS,
DOVER PUBLICATIONS, INC., NEW YORK, 1968.

[2] D.B.HUNTER,
THE CALCULATION OF SOME BESSEL FUNCTIONS,
MATHEMATICS OF COMPUTATION (1964), P. 123.

[3] YUDELL LUKE,
THE SPECIAL FUNCTIONS AND THEIR APPROXIMATIONS, VOLUME 2,
ACADEMIC PRESS, NEW YORK AND LONDON (1969).

[4] C.W.CLENSHAW,
CHEBYSHEV SERIES FOR MATHEMATICAL FUNCTIONS,
NAT. PHYS. LAB. MATH. TABLES, VOLUME 5,
HER MAJESTY,S STATIONARY OFFICE, LONDON (1962).

[5] W.GAUTSCHI,
COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS,
SIAM REVIEWS, VOLUME 9 (1967), P. 24.

[6] J.M.BLAIR,
RATIONAL CHEBYSHEV APPROXIMATIONS FOR THE MODIFIED
BESSEL FUNCTIONS I0(X) AND I1(X);
MATHEMATICS OF COMPUTATIONS,VOLUME 28,
NR 126, APRIL 1974, P. 581-583.

SUBSECTION: BESS I0.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" BESS I0(X); "VALUE" X; "REAL" X;
"CODE" 35170;

BESS I0 DELIVERS THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF ORDER ZERO WITH ARGUMENT X;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

NONEXP BESS I0 = CP35175.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR X = 0 BESS I0 IS ASSIGNED ITS VALUE IMMEDIATELY;
FOR 0 < ABS(X) <= 15.0 17 MULTIPLICATIONS AND ONE DIVISION ARE REQUIRED;
FOR ABS(X) > 15.0 11 MULTIPLICATIONS, 3 DIVISIONS, ONE EVALUATION OF THE SQUARE ROOT AND ONE EVALUATION OF THE EXPONENNTIAL FUNCTION ARE REQUIRED.

METHOD AND PERFORMANCE: RATIONAL APPROXIMATION, SEE [6].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" BESS I0(X); "CODE" 35170;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D"-ZD")",
    X, BESS I0(X))
"END"
```

PRINTS THE FOLLOWING RESULTS:

1 .12660658777520" 1

SUBSECTION: BESS I1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" BESS I1(X); "VALUE" X; "REAL" X;
"CODE" 35171;

BESS I1 DELIVERS THE MODIFIED BESSEL FUNCTION OF THE
FIRST KIND OF ORDER ONE WITH ARGUMENT X;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

NONEXP BESS I1 = CP35176.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR X = 0 BESS I1 IS ASSIGNED ITS VALUE IMMEDIATELY;
FOR 0 < ABS(X) <= 15.0 17 MULTIPLICATIONS AND ONE DIVISION
ARE REQUIRED;
FOR ABS(X) > 15.0 12 MULTIPLICATIONS, 3 DIVISIONS, ONE EVALUATION
OF THE SQUARE ROOT AND ONE EVALUATION OF THE EXPONENTIAL FUNCTION
ARE REQUIRED.

METHOD AND PERFORMANCE: RATIONAL APPROXIMATION, SEE [6].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" BESS I1(X); "CODE" 35171;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D"-ZD")",
    X, BESS I1(X))
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
1      .56515910399252"  0
```

SUBSECTION: BESS I.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS I(X, N, I); "VALUE" X, N;
"INTEGER" N; "REAL" X; "ARRAY" I;
"CODE" 35172;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY I;
I:  <ARRAY IDENTIFIER>;
    "ARRAY" I[0 : N];
    EXIT: I[L] POSSESSES THE VALUE OF THE MODIFIED BESSEL FUNCTION
    OF THE FIRST KIND OF ORDER L (0 <= L <= N).
```

METHOD AND PERFORMANCE: SEE NON EXP BESS I (THIS SECTION).

PROCEDURES USED :

NONEXP BESS I = CP 35177.

REQUIRED CENTRAL MEMORY:

NO AUXILIARY ARRAYS ARE USED.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO THE MAXIMUM OF 1.359 * X + 72 AND N + 18.

EXAMPLE OF USE : THE FOLLOWING PROGRAM CHECKS FOR X = 1 (1) 20 THE WRONSKIAN RELATION

X * (I[N - 1] * K[N] + I[N] * K[N - 1]) - 1 = 0

FOR N = 1 (1) 5; THE PROGRAM READS:

```
"BEGIN" "REAL" X; "INTEGER" N; "ARRAY" I, K[0:5];
    "PROCEDURE" BESS I(X, N, I); "CODE" 35172;
    "PROCEDURE" BESS K(X, N, K); "CODE" 35174;

    "FOR" X:= 1 "STEP" 1 "UNTIL" 20 "DO"
    "BEGIN" OUTPUT(61,"("/ZD")", X);
        BESS I(X, 5, I); BESS K(X, 5, K);
        "FOR" N:= 1, 2, 3, 4, 5 "DO"
        OUTPUT(61,"("BB-.D"-ZD")",
        X * (I[N] * K[N - 1] + I[N - 1] * K[N]) - 1)
    "END"
"END"
```

THE RESULTS ARE:

1	.0" 0	.0" 0	-.7"-14	-.7"-14	-.7"-14
2	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
3	.7"-14	.7"-14	.0" 0	.0" 0	.0" 0
4	.7"-14	.0" 0	.0" 0	.0" 0	.0" 0
5	.0" 0	.7"-14	.7"-14	.0" 0	.0" 0
6	.0" 0	.0" 0	.0" 0	.0" 0	-.7"-14
7	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
8	-.1"-13	-.1"-13	-.1"-13	-.1"-13	-.1"-13
9	.0" 0	.0" 0	.0" 0	-.7"-14	-.7"-14
10	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
11	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
12	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
13	.7"-14	.7"-14	.0" 0	.7"-14	.0" 0
14	.0" 0	.7"-14	.0" 0	.0" 0	.0" 0
15	.0" 0	.0" 0	.0" 0	.0" 0	.0" 0
16	.0" 0	.0" 0	.0" 0	.0" 0	-.7"-14
17	.7"-14	.0" 0	.0" 0	.0" 0	.0" 0
18	.7"-14	.0" 0	.0" 0	.0" 0	-.7"-14
19	.7"-14	.0" 0	.0" 0	.0" 0	.0" 0
20	.0" 0	.0" 0	.0" 0	.0" 0	-.7"-14

SUBSECTION: BESS K01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS K01(X, K0, K1); "VALUE" X; "REAL" X, K0, K1;
"CODE" 35173;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;
K0: <VARIABLE>;
    EXIT: K0 HAS THE VALUE OF THE MODIFIED BESSEL FUNCTION
          OF THE THIRD KIND OF ORDER 0 WITH ARGUMENT X;
K1: <VARIABLE>;
    EXIT: K1 HAS THE VALUE OF THE MODIFIED BESSEL FUNCTION
          OF THE THIRD KIND OF ORDER ONE.
```

PROCEDURES USED:

NONEXP BESS K01 = CP35178

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME: DEPENDS ON THE VALUE OF X;
THE GLOBAL VALUES IN MILLISECONDS ARE:

```
0   < X <= 1.5   :       2.2 MS,
1.5 < X <= 5.0   :       5.5 MS,
5.0 < X          :       2.3 MS, ON THE CYBER 73/28.
```

METHOD AND PERFORMANCE:

FOR THE COMPUTATION OF K0 AND K1 THREE DIFFERENT METHODS
ARE USED DEPENDING ON THE VALUE OF X:
FOR 0 < X <= 1.5 K0 AND K1 ARE EVALUATED BY MEANS OF TAYLOR SERIES
EXPANSIONS (SEE [1], P. 375, FORMULA 9.6.13);
FOR X > 1.5 K0 AND K1 ARE COMPUTED BY MEANS OF A CALL
OF THE CODE PROCEDURE NONEXP BESS K01 (SEE DESCRIPTION AHEAD)
AND MULTIPLICATION BY EXP(- X).

EXAMPLE OF USE: THE PROGRAM

```
"BEGIN" "REAL" X, K0, K1;
    "PROCEDURE" BESS K01(X, K0, K1); "CODE" 35173;
    "FOR" X:= .5, 1.5, 2.5 "DO"
    "BEGIN" BESS K01(X, K0, K1);
        OUTPUT(61,"("/,4BD.D,2(B-.14D"-ZD)")",X,K0,K1)
    "END"
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
0.5  .92441907122766"  0  .16564411200033"  1
1.5  .21380556264754"  0  .27738780045683"  0
2.5  .62347553200366" -1  .73890816347746" -1
```

SUBSECTION: BESS K.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS K(X, N, K); "VALUE" X, N;
"INTEGER" N; "REAL" X; "ARRAY" K;
"CODE" 35174;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY K; N >= 0;
K:  <ARRAY IDENTIFIER>;
    "ARRAY" K[0 : N];
    EXIT: K[I] POSSESSES THE VALUE OF THE MODIFIED BESSEL FUNCTION
          OF THE THIRD KIND OF ORDER I (0 <= I <= N).
```

PROCEDURES USED: K01 = CP 35173.

REQUIRED CENTRAL MEMORY:

NO AUXILIARY ARRAYS ARE USED.

RUNNING TIME :

DEPENDS ON THE VALUE OF X (SEE TABLE BELONGING TO BESS K01) AND N.

METHOD AND PERFORMANCE:

K[0], ..., K[N] ARE COMPUTED ACCORDING TO THE RECURRENCE RELATION

K[I + 1] = K[I - 1] + (2 * I / X) * K[I], I = 2, ..., N,

(SEE [1], P. 376, FORMULA 9.6.26).

EXAMPLE OF USE: THE PROGRAM

```
"BEGIN" "ARRAY"  K[0 : 2]; "REAL" X;
    "PROCEDURE" BESS K(X, N, K); "CODE" 35174;
    "FOR" X:= .5, 1.0, 1.5, 2.0 "DO"
    "BEGIN" BESS K(X, 2, K);
       OUTPUT(61,"("/D.D,3(BB.12D"-D)")",X,K)
    "END"
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
0.5  .924419071228"0  .165644112000"1  .755018355124"1
1.0  .421024438241"0  .601907230197"0  .162483889864"1
1.5  .213805562648"0  .277387800457"0  .583655963257"0
2.0  .113893872750"0  .139865881817"0  .253759754566"0
```

SUBSECTION: NONEXP BESS I0.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" NONEXP BESS I0(X); "VALUE" X; "REAL" X;
"CODE" 35175;

NONEXP BESS I0 DELIVERS THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF ORDER 0 WITH ARGUMENT X MULTIPLIED BY EXP(-ABS(X)).

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

BESS I0 = CP35170.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR X = 0 NONEXP BESS I0 IS ASSIGNED ITS VALUE IMMEDIATELY; FOR 0 < ABS(X) <= 15.0 18 MULTIPLICATIONS, ONE DIVISION AND ONE EVALUATION OF THE EXPONENTIAL FUNCTION ARE REQUIRED; FOR ABS(X) > 15.0 10 MULTIPLICATIONS, 3 DIVISIONS AND ONE EVALUATION OF THE SQUARE ROOT ARE REQUIRED.

METHOD AND PERFORMANCE:

SEE [6].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" NONEXP BESS I0(X); "CODE" 35175;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D"-ZD")",
    X, NONEXP BESS I0(X))
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
  1      .46575960759364"  0
```

SUBSECTION: NONEXP BESS I1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"REAL" "PROCEDURE" NONEXP BESS I1(X); "VALUE" X; "REAL" X;
"CODE" 35176;
```

NONEXP BESS I1 DELIVERS THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF ORDER 1 WITH ARGUMENT X MULTIPLIED BY EXP(-ABS(X)).

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTION.

PROCEDURES USED:

BESS I1 = CP35171.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

FOR X = 0 NONEXP BESS I1 IS ASSIGNED ITS VALUE IMMEDIATELY;
FOR 0 < ABS(X) <= 15.0 18 MULTIPLICATIONS, ONE DIVISION AND ONE EVALUATION OF THE EXPONENTIAL FUNCTION ARE REQUIRED;
FOR X > 15.0 11 MULTIPLICATIONS, 3 DIVISIONS AND ONE EVALUATION OF THE SQUARE ROOT ARE REQUIRED.

METHOD AND PERFORMANCE:

SEE [6].

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X;
    "REAL" "PROCEDURE" NONEXP BESS I1(X); "CODE" 35176;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D"-ZD")",
    X, NONEXP BESS I1(X))
"END"
```

DELIVERS THE FOLLOWING RESULTS:

```
  1      .20791041534972"  0
```

SUBSECTION: NONEXP BESS I.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
```
"PROCEDURE" NONEXP BESS I(X, N, I); "VALUE" X, N;
"INTEGER" N; "REAL" X; "ARRAY" I;
"CODE" 35177;
```

THE MEANING OF THE FORMAL PARAMETERS IS:
```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY I; N >= 0;
I:  <ARRAY IDENTIFIER>;
    "ARRAY" I[0:N];
    EXIT: I[L] POSSESSES THE VALUE OF THE MODIFIED
          BESSEL FUNCTION OF THE FIRST KIND OF ORDER L (L=0,...,N)
          MULTIPLIED BY EXP (- ABS(X)).
```

PROCEDURES USED: START = CP 35185;

REQUIRED CENTRAL MEMORY:

NO AUXILIARY ARRAYS ARE USED.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO THE MAXIMUM OF 1.359*X + 72 AND N+18.

METHOD AND PERFORMANCE: SEE [5].

EXAMPLE OF USE: THE PROGRAM

```
"BEGIN" "REAL" X; "ARRAY" I[0:2];
  "PROCEDURE" NONEXP BESS I(X, N, I); "CODE" 35179;
"FOR" X:= .5, 1.0, 1.5, 2.0, 2.5 "DO"
  "BEGIN" NONEXP BESS I(X, 2, I);
    OUTPUT(61, "("/,4BZ.D,3(B-.12D"-D)")", X,
    I[0], I[1], I[2])
  "END"
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
 .5  .152410938577" 1  .273100970821" 1  .124481482186" 2
1.0  .114446307981" 1  .163615348626" 1  .441677005233" 1
1.5  .958210053295" 0  .124316587355" 1  .261576455136" 1
2.0  .841568215071" 0  .103347684707" 0  .187504506214" 1
2.5  .759548690328" 0  .900174423908" 0  .147968822945" 1
```

SUBSECTION: NONEXP BESS K01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NONEXP BESS K01(X, K0, K1);
"VALUE" X; "REAL" X, K0, K1;
"CODE" 35178;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;

K0: <VARIABLE>;
EXIT: K0 HAS THE VALUE OF THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND OF ORDER 0 WITH ARGUMENT X MULTIPLIED BY EXP(X);

K1: <VARIABLE>;
EXIT: K1 HAS THE VALUE OF THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND OF ORDER 1 MULTIPLIED BY EXP(X).

PROCEDURES USED:

BESS K01 = CP35173.

REQUIRED CENTRAL MEMORY:

NO ARRAYS ARE USED.

RUNNING TIME:

DEPENDS ON THE VALUE OF X; BECAUSE OF THE STRONG INTERDEPENDENCE OF THE BESS K01 (= CP35173) AND NONEXP BESS K01 THE READER IS REFERRED TO THE TABLE OF RUNNING TIMES BELONGING TO BESS K01.

METHOD AND PERFORMANCE:

FOR THE COMPUTATION OF K0 AND K1 THREE
DIFFERENT METHODS ARE USED DEPENDING ON THE VALUE OF X:
FOR 0 < X <= 1.5 K0 AND K1 ARE COMPUTED BY MEANS OF
MULTIPLICATION OF THE MODIFIED BESSEL FUNCTIONS OF ORDER
ZERO AND ONE (SEE DESCRIPTION OF K0) BY EXP(X);
FOR 1.5 < X <= 5 K0 AND K1 ARE COMPUTED BY
THE EVALUATION OF THEIR INTEGRAL REPRESENTATIONS (SEE [1],
P. 376, FORMULA 9.6.23) BY MEANS OF THE TRAPEZOIDAL RULE (SEE [2]);
FOR X > 5 K0 AND K1 ARE COMPUTED BY MEANS OF
A FINITE CHEBYSHEV SERIES EXPANSION (SEE [3], P. 339 AND [4]).

EXAMPLE OF USE: THE PROGRAM

```
"BEGIN" "REAL" X, K1;
  "PROCEDURE" NONEXP BESS K01(X, K0, K1); "CODE" 35178;
  "FOR" X:= .5, 1.0, 1.5, 2.0, 2.5 "DO"
  "BEGIN" NON EXP BESS K01(X, K0, K1);
    OUTPUT(61,"("/,4BZ.D,2(5B-.14D"-ZD)")",
    X, K0, K1)
  "END"
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
 .5     .15241093857739"  1     .27310097082118"  1
1.0     .11444630798069"  1     .16361534862633"  1
1.5     .95821005329496"  0     .12431658735525"  1
2.0     .84156821507078"  0     .10334768470687"  1
2.5     .75954869032810"  0     .90017442390788"  0
```

SUBSECTION: NONEXP BESS K.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NONEXP BESS K(X, N,K); "VALUE" X, N;
"INTEGER" N; "REAL" X; "ARRAY" K;
"CODE" 35179;
```

```
THE MEANING OF THE FORMAL PARAMETERS IS:
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY K; N >= 0;
K:  <ARRAY IDENTIFIER>;
    "ARRAY" K[0:N];
    EXIT: K[I] POSSESSES THE VALUE OF THE MODIFIED BESSEL
          FUNCTION OF THE THIRD KIND OF ORDER I (I = 0, ..., N)
          MULTIPLIED BY EXP(X).
```

PROCEDURES USED:

NONEXP BESS K01 = CP 35178.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE USED.

METHOD AND PERFORMANCE:

K[0] AND K[1] ARE COMPUTED BY USING NONEXP BESS K01 (CP 35178), WHILE K[2], ..., K[N] ARE COMPUTED ACCORDING TO THE RECURRENCE RELATION
K[I+1]=K[I]+(2*I/X)*K[I], I>=2
(SEE [1], P. 376, FORMULA 9.6.26).

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X; "ARRAY" K[0:1];
  "PROCEDURE" NONEXP BESS K(X, N, K); "CODE" 35179;
  "FOR" X:= .5, 1.0, 1.5, 2.0 "DO"
  "BEGIN" NONEXP BESS K(X, 2, K);
    OUTPUT(61, "("/,Z.D,3(5B.14D"D)")",X,K)
  "END"
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
 .5   .15241093857739"1  .27310097082118"1  .12448148218621"2
1.0   .11444630798069"1  .16361534862633"1  .44167700523334"1
1.5   .95821005329496"0  .12431658735525"1  .26157645513649"1
2.0   .84156821507078"0  .10334768470687"U  .18750450621395"1
```

SOURCE TEXT(S):

```
"CODE" 35170;
    "REAL" "PROCEDURE" BESS I0(X); "VALUE" X; "REAL" X;
    "IF" X= 0 "THEN" BESS I0:=1
    "ELSE" "IF" ABS(X) < = 15.0 "THEN"
    "BEGIN" "REAL" Z, DENOMINATOR, NUMERATOR;
      Z:= X*X; NUMERATOR:=
      (Z*(Z*(Z*(Z*(Z*(Z*(Z*
      (Z*(Z*(Z*(Z*(Z*(Z*(Z*
       .21058  07228  90567  "-22
      +.38071  52423  45326  "-19)
      +.47944  02575  48300  "-16)
      +.43512  59712  62668  "-13)
      +.30093  11271  12960  "-10)
      +.16022  46793  95361  "-07)
      +.65485  83700  96785  "-05)
      +.20259  10841  43397  "-02)
      +.46307  62847  21000  "+00)
      +.75433  73289  48189  "+02)
      +.83079  25418  09429  "+04)
      +.57166  11305  63785  "+06)
      +.21641  55723  61227  "+08)
      +.35664  44822  44025  "+09)
      +.14404  82982  27235  "+10);
      DENOMINATOR:= (Z*(Z*
      (Z-.30764  69126  82801  "04)
        +.34762  63324  05882  "07)
        -.14404  82982  27235  "10);
      BESS I0:= -NUMERATOR/DENOMINATOR;
    "END" "ELSE"
    "BEGIN" "REAL" "PROCEDURE" NONEXP BESS I0(X); "CODE" 35175;
      BESS I0:= EXP(ABS(X)) * NONEXP BESS I0(X)
    "END"
          "EOP"
```

```
"CODE" 35171;
    "REAL" "PROCEDURE" BESS I1(X); "VALUE" X; "REAL" X;
    "IF" X=0 "THEN" BESS I1:=0 "ELSE"
    "IF" ABS(X) <= 15.0 "THEN"
    "BEGIN" "REAL" Z, DENOMINATOR, NUMERATOR;
      Z:= X*X;
      DENOMINATOR:=
      Z*(Z-.22258  36740  00860  "4)
        +.13629  35930  52499  "7;
      NUMERATOR:=
      (Z*(Z*(Z*(Z*(Z*(Z*(Z*
      (Z*(Z*(Z*(Z*(Z*(Z*(Z*
       .20717  57672  32792  "-26
      +.25709  19055  84414  "-23)
      +.30627  92836  56135  "-20)
      +.26137  27721  58124  "-17)
      +.17846  93614  10091  "-14)
      +.96362  88915  18450  "-12)
      +.41006  89068  47159  "-09)
      +.13545  52288  41096  "-06)
      +.33947  28903  08516  "-04)
      +.62472  61951  27003  "-02)
      +.80614  48788  21295  "-00)
      +.68210  05679  80207  "+02)
      +.34106  97522  84422  "+04)
      +.84070  57728  77836  "+05)
      +.68146  79652  62502  "+06);
      BESS I1:= X*(NUMERATOR/DENOMINATOR)
    "END" "ELSE"
    "BEGIN" "REAL" "PROCEDURE" NONEXP BESS I1(X); "CODE" 35176;
    BESS I1:= EXP(ABS(X))*NONEXP BESS I1(X)
    "END";
          "EOP"

"CODE" 35172;
    "PROCEDURE" BESS I(X, N, I);
    "VALUE" X, N; "INTEGER" N; "REAL" X; "ARRAY" I;
    "IF" X = 0 "THEN"
    "BEGIN" I[0]:= 1;
        "FOR" N:= N "STEP" - 1 "UNTIL" 1 "DO" I[N]:= 0;
    "END" "ELSE"
    "BEGIN" "REAL" EXPX;
        "PROCEDURE" NONEXP BESS I(X, N, NONEXPI); "CODE" 35177;
        EXPX:= EXP(ABS(X)); NONEXP BESS I(X, N, I);
        "FOR" N:= N "STEP" - 1 "UNTIL" 0 "DO"
        I[N]:= I[N] * EXPX
    "END" BESS I;
        "EOP"
```

```
"CODE" 35173;
    "PROCEDURE" BESS KO1(X, KO, K1); "VALUE" X; "REAL" X, KO, K1;
    "IF" X <= 1.5 "THEN"
    "BEGIN" "INTEGER" K; "REAL" C, D, R, S, SUMO, SUM1, T,
        TERM, TO, T1;
        SUMO:= D:= LN(2/X) -.57721566490153286O6;
        SUM1:=    C:= -1 -2    * D; R:= TERM:= 1; T:= X * X/4;
        "FOR" K:= 1,K+1 "WHILE" ABS(TO/SUMO) + ABS(T1/SUM1) >
        "-15 "DO"
        "BEGIN"  TERM:= T * TERM * R * R; D:= D + R;
            C:= C - R; R:= 1/(K+1); C:= C - R;
            TO:= TERM * D; T1:= TERM * C * R;
            SUMO:= SUMO + TO; SUM1:= SUM1 + T1
        "END";
        KO:= SUMO; K1:= (1 + T * SUM1) / X
    "END" "ELSE"
    "BEGIN" "REAL" EXPX;
        "PROCEDURE" NONEXP BESS KO1(X, KO, K1); "CODE" 35178;
        EXPX:= EXP(- X);
        NONEXP BESS KO1(X, KO, K1); K1:= EXPX * K1; KO:= KO * EXPX
    "END" BESS KO1;
         "EOP"

"CODE" 35174;
    "PROCEDURE" BESS K(X, N, K); "VALUE" X, N;
    "REAL" X; "INTEGER" N; "ARRAY" K;
    "BEGIN"  "INTEGER" I; "REAL" KO, K1, K2;
        "PROCEDURE" BESS KO1(X, KO, K1); "CODE" 35173;
        BESS KO1(X, KO, K1); K[O]:= KO; "IF" N > O "THEN" K[1]:= K1;
        X:= 2 / X;
        "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" K[I]:= K2:= KO + X * (I-1)* K1;
            KO:= K1; K1:= K2
        "END"
    "END" BESS K;
         "EOP"

"CODE" 35175;
    "REAL" "PROCEDURE" NONEXP BESS IO(X);
    "VALUE" X; "REAL" X;
    "IF" X= O "THEN"
    NONEXP BESS IO:=1 "ELSE"
    "IF" ABS(X) <= 15.0 "THEN"
    "BEGIN" "REAL" "PROCEDURE" BESS IO(X); "CODE" 35170;
      NONEXP BESS IO:= EXP(-ABS(X))*BESS IO(X)
    "END" "ELSE"
    "BEGIN" "REAL" SQRTX, AR, BR, BR1, BR2, Z, Z2, NUMERATOR,
      DENOMINATOR;
      X:=ABS(X); SQRTX:= SQRT(X);
                                                              "COMMENT"
```

```
      BR1:= BR2:= 0; Z:= 30/X-1; Z2:= Z+Z;
      "FOR" AR:= .24392  60769  778,
      -.11559  19781  04435  "3,
      +.78403  42490  05088  "4,
      -.14346  46313  13583  "6  "DO"
      "BEGIN" BR:= Z2*BR1-BR2+AR; BR2:= BR1; BR1:= BR "END";
      NUMERATOR:= Z*BR1-BR2+.34651  98333  57379  "6;
      BR1:= BR2:= 0;
      "FOR" AR:= 1, -.32519  73333  69824  "3,
      +.20312  84361  00794  "5,
      -.36184  77792  19653  "6  "DO"
      "BEGIN" BR:= Z2*BR1 - BR2 + AR;
          BR2:= BR1; BR1:= BR
      "END";
      DENOMINATOR:= Z*BR1 - BR2 +.86566  52748  32055  "6;
      NONEXP BESS I0:= (NUMERATOR/DENOMINATOR)/SQRTX;
    "END";
          "EOP"

"CODE" 35176;
    "REAL" "PROCEDURE" NONEXP BESS I1(X); "VALUE" X; "REAL" X;
    "IF" X=0 "THEN" NONEXP BESS I1:= 0
    "ELSE" "IF" ABS(X)> 15.0 "THEN"
    "BEGIN" "INTEGER" SIGNX ;
      "REAL" AR, BR, BR1, BR2, Z, Z2,
      SQRTX, DENOMINATOR, NUMERATOR;
      SIGNX:= SIGN(X); X:= ABS(X); SQRTX:= SQRT(X);
      Z:= 30/X - 1; Z2 := Z + Z;
          BR1:= BR2:= 0;
      "FOR" AR:=
      +.14940  52814  740  "+1,
      -.36202  64202  42263  "+3,
      +.22054  97222  60336  "+5,
      -.40892  80849  44275  "+6 "DO"
      "BEGIN" BR:= Z2 * BR1 - BR2 + AR;
          BR2:= BR1; BR1:= BR
      "END";
      NUMERATOR:= Z * BR1 -BR2 +.10277  66923  71524  "7;
      BR1:= BR2:= 0; "FOR" AR:= 1,
      -.63100  32005  51590  "3,
      +.49681  19495  33398  "5,
      -.10042  54281  33695  "7  "DO"
      "BEGIN" BR:= Z2 * BR1 - BR2 + AR; BR2:= BR1; BR1:=BR "END";
      DENOMINATOR:= Z * BR1 - BR2 +.26028  87678  9105  "7;
      NONEXP BESS I1:= ((NUMERATOR/DENOMINATOR)/SQRTX) * SIGN X
    "END" "ELSE"
    "BEGIN" "REAL" "PROCEDURE"  BESS I1(X); "CODE" 35171;
      NONEXP BESS I1:= EXP(-ABS(X))*BESS I1(X)
    "END";
          "EOP"
```

```
"CODE" 35177;
    "PROCEDURE" NONEXP BESS I(X, N, I); "VALUE" X, N;
    "INTEGER" N; "REAL" X; "ARRAY" I;
    "IF" X = 0 "THEN"
    "BEGIN" I[0]:= 1; "FOR" N:= N "STEP" - 1 "UNTIL" 1 "DO"
        I[N]:= 0
    "END" "ELSE"
    "BEGIN" "INTEGER" K; "REAL" X2, R, S; "BOOLEAN" NEGATIVE;
        "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
        NEGATIVE:= X < 0; X:= ABS(X);
        R:= S:= 0; X2:= 2/X; K:= START(X,N,1);
        "FOR" K:= K "STEP" - 1 "UNTIL" 1 "DO"
        "BEGIN" R:= 1 / (R + X2 * K); S:= R * (2 + S);
            "IF" K <= N "THEN" I[K]:= R
        "END";
        I[0]:= R:= 1 / (1 + S);
        "IF" NEGATIVE "THEN"
        "BEGIN" "FOR" K:= 1 "STEP" 1 "UNTIL" N "DO"
            I[K]:= R:= - R * I[K]
        "END" "ELSE"
        "FOR" K:=1 "STEP" 1 "UNTIL" N "DO" I[K]:= R:= R * I[K];
    "END" NONEXP BESS I;
         "EOP"

"CODE" 35178;
    "PROCEDURE" NONEXP BESS K01(X, K0, K1);"VALUE" X;"REAL" X, K0, K1;
    "IF" X <= 1.5 "THEN"
    "BEGIN" "REAL" EXPX;
        "PROCEDURE" BESS K01(X, K0, K1); "CODE" 35173;
        EXPX:= EXP(X); BESS K01(X, K0, K1); K0:= K0 * EXPX;
        K1:= EXPX * K1
    "END" "ELSE" "IF" X <= 5 "THEN"
    "BEGIN" "INTEGER" R; "REAL" T2, FAC, S1, S2, TERM1, TERM2,
        SQRTEXPR, EXPH2, X2;
        S1:= .5; S2:=0; R:= 0; X2:= X + X;
        EXPH2:= 1 / SQRT(5 * X);
        "FOR" FAC:= .90483741803596,
        .67032004603564,       .40656965974060,       .20189651799466,
        .82084998623899"-1,    .27323722447293"-1,    .74465830709243"-2,
        .16615572731739"-2,    .30353913807887"-3,    .45399929762485"-4,
        .55595132416500"-5,    .55739036926944"-6,    .45753387694459"-7,
        .30748798795865O"-8,   .16918979226151"-9,    .76218651945127"-11,
        .28111852987891"-12,   .84890440338729"-14,   .20987910487393"-15,
        .42483542552916"-17 "DO"
        "BEGIN" R:= R + 1; T2:= R * R / 10;
            SQRTEXPR:= SQRT(T2 / X2 + 1);
            TERM1:= FAC / SQRTEXPR; TERM2:= FAC * SQRTEXPR * T2;
            S1:= S1 + TERM1; S2:= S2 + TERM2
        "END";
                                                        "COMMENT"
```

```
;
            K0:= EXPH2 * S1; K1:= EXPH2 * S2 * 2
        "END" "ELSE"
        "BEGIN" "INTEGER" R;
            "REAL" BR, BR1, BR2, CR, CR1, CR2, DR, ERMIN1, ERPLUS1, ER,
            F0, F1, EXPX, Y, Y2;
            Y:= 10 / X - 1; Y2:= Y + Y; R:= 30;
            BR1:= BR2:= CR1:= CR2:= ERPLUS1:= ER:= 0;
            "FOR" DR:= .27545" - 15, -.172697" - 14, .1136042 " - 13,
            -.7883236        " -13, .58081063        " -12,
            -.457993622      " -11, .3904375576      " -10,
            -.36454717921    " - 9, .379299645568    " - 8,
            -.450473376411   " - 7, .53257510850049  " - 6,
            -.11106685196665" - 4, .26953261276272 " - 3,
            -.11310504646928" - 1 "DO"
            "BEGIN" R:= R - 2; BR:= Y2 * BR1 - BR2 + DR;
                CR:= CR1 * Y2 - CR2 + ER;
                ERMIN1:= R * DR + ERPLUS1; ERPLUS1:= ER; ER:= ERMIN1;
                BR2:= BR1; BR1:= BR; CR2:= CR1; CR1:= CR
            "END";
            F0:= Y * BR1 - BR2 + .9884081742308258;
            F1:= Y * CR1 - CR2 + ER / 2;
            EXPX:= SQRT(1.5707963267949 / X); K0:= F0:= F0 * EXPX;
            K1:= (1 + .5 / X) * F0 + (10 / X / X) * EXPX * F1
          "END"   K0;
             "EOP"

"CODE" 35179;
        "PROCEDURE" NONEXP BESS K(X, N, K); "VALUE" X, N;
        "REAL" X; "INTEGER" N; "ARRAY" K;
        "BEGIN"  "INTEGER" I; "REAL" K0, K1, K2;
            "PROCEDURE" NONEXP BESS K01(X, K0, K1); "CODE" 35178;
            NONEXP BESS K01(X, K0, K1);
            K[0]:= K0; "IF" N> 0 "THEN" K[1]:= K1; X:= 2 / X;
            "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" K[I]:= K2:= K0 + X * (I-1)* K1;
                K0:= K1; K1:= K2
            "END"
        "END" NONEXP BESS K;
             "EOP"
```

AUTHORS: M.BAKKER AND N.M.TEMME.

CONTRIBUTOR: R.MONTIJN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 781101.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE PROCEDURES:

BESS JAPLUSN:

THIS PROCEDURE CALCULATES THE BESSEL FUNCTIONS OF THE FIRST KIND OF ORDER A+K (0<=K<=N, 0<=A<1) AND ASSIGNS THEM TO AN ARRAY. THE ARGUMENT MUST BE NON-NEGATIVE.

BESS YA01:

THIS PROCEDURE CALCULATES THE BESSEL FUNCTIONS OF THE SECOND KIND (ALSO CALLED NEUMANN'S FUNCTIONS) OF ORDER A AND A+1 AND ARGUMENT X>0.

BESS YAPLUSN:

THIS PROCEDURE GENERATES AN ARRAY OF BESSEL FUNCTIONS OF THE SECOND KIND OF ORDER A+N, N=0, 1, 2, ..., NMAX, AND ARGUMENT X>0.
THE BESSEL FUNCTIONS OF THE SECOND KIND CORRESPOND TO THE FUNCTION DEFINED IN FORMULA 9.1.2 OF REFERENCE [1].

BESS PQA01:

THIS PROCEDURE IS AN AUXILIARY PROCEDURE FOR THE COMPUTATION OF THE BESSEL FUNCTIONS FOR LARGE VALUES OF THEIR ARGUMENT.

BESS ZEROS:

THIS PROCEDURE CALCULATES THE FIRST N ZEROS OF A BESSEL FUNCTION OF THE FIRST OR THE SECOND KIND OR ITS DERRIVATIVE.

START:

THIS IS AN AUXILIARY PROCEDURE WHICH COMPUTES A STARTING VALUE OF AN ALGORITHM USED IN SEVERAL BESSEL FUNCTION PROCEDURES.

KEYWORDS:

BESSEL FUNCTION, BESSEL FUNCTION OF THE SECOND KIND, NEUMANN'S FUNCTION, ZEROS OF BESSEL FUNCTIONS.

REFERENCES:

[1]. ABRAMOWITZ, M., AND STEGUN, I. (EDS),
HANDBOOK OF MATHEMATICAL FUNCTIONS WITH FORMULAS, GRAPHS AND MATHEMATICAL TABLES.
APPL. MATH. SER. 55, U.S. GOVT. PRINTING OFFICE, WASHINGTON, D.C. , 1974.

[2]. GAUTSCHI, W., COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS.
SIAM REVIEW, VOLUME 9(1967), NUMBER 1, P.24 FF.

[3]. TEMME, N.M. ON THE NUMERICAL EVALUATION OF THE ORDINARY BESSEL FUNCTION OF THE SECOND KIND.
J. COMP. PHYS., 21, P. 343 FF, 1976.

[4]. WATSON, G.N.
A TREATISE ON THE THEORY OF BESSEL FUNCTIONS.
CAMBRIDGE UNIV. PRESS, LONDON AND NEW YORK, 1945.

[5]. TEMME, N.M., SPECIALE FUNCTIES, IN:
COLLOQUIUM NUMERIEKE PROGRAMMATUUR,
J.C.P. BUS (RED.), MC SYLLABUS 29.1B,
MATHEMATICAL CENTRE, AMSTERDAM, 1976.

[6]. TEMME, N.M., AN ALGOLRITHM WITH ALGOL 60 IMPLEMENTATION FOR THE CALCULATION OF THE ZEROS OF A BESSEL FUNCTION,
REPORT TW 179 MATHEMATICAL CENTRE, AMSTERDAM, 1978.

SUBSECTION: BESS JAPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS JAPLUSN(A, X, N, JA);
"VALUE" A, X, N;
"INTEGER" N; "REAL" A, X; "ARRAY" JA;
"CODE" 35180;
```

```
THE MEANING OF THE FORMAL PARAMETERS IS:
A:  < ARITHMETIC EXPRESSION > ;
    THE NONINTEGER PART OF THE ORDER; 0 <= A < 1;
X:  < ARITHMETIC EXPRESSION >;
    THE ARGUMENT VALUE; X > = 0;
N:  < ARITHMETIC EXPRESSION >;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY JA;
JA:  < ARRAY IDENTIFIER >;
    "ARRAY" JA[0:N];
    EXIT:  JA[K] IS ASSIGNED THE VALUE OF THE BESSEL
           FUNCTION OF THE FIRST KIND J[K+A](X),
           0 < = K < = N.
```

PROCEDURES USED:

```
BESS J         =  CP 35162,
SPHER BESS J   =  CP 35150,
GAMMA          =  CP 35061,
START          =  CP 35185.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

IN ALL THE CASES THE BESSEL FUNCTIONS ARE COMPUTED ACCORDING TO THE MILLER METHOD DISCRIBED IN [2, P.46-52]. THE STARTING VALUE IS COMPUTED BY THE PROCEDURE START.

RUNNING TIME: ROUGHLY PROPORTIONAL TO THE MAXIMUM OF X AND N.

EXAMPLE OF USE:

```
"BEGIN" "INTEGER" N; "REAL" A, X; "ARRAY" JA[0:2];
    "PROCEDURE" BESS JAPLUSN(A, X, N, JA); "CODE" 35180;
    X:= 2; A:= .78; N:= 2;
    BESS JAPLUSN(A, X, N, JA);
    OUTPUT(61, "("/, "("X=")"D, 3B"("A=")".DD, 3B"("N=")"D,
    /, 3(3B-.14D"-ZD)")", X, A, N, JA[0], JA[1], JA[2])
"END"
```

RESULTS:

```
X=2  A= .78  N=2
.57306126928364"0  .41529475124424" 0  .16616338793111" 0
```

SUBSECTION: BESS YA01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS YA01(A, X, YA, YA1);
"VALUE" A, X; "REAL" A, X, YA, YA1;
"CODE" 35181;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER;
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;
YA:     <VARIABLE>;
        EXIT: THE NEUMANN FUNCTION OF ORDER A
              AND ARGUMENT X;
YA1:    <VARIABLE>;
        EXIT: THE NEUMANN FUNCTION OF ORDER A+1.
```

PROCEDURES USED:

```
RECIP GAMMA = CP 35060;
BESS PQA01  = CP 35183;
SINH        = CP 35111.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

FOR 0<X<3 THE BESSEL FUNCTIONS ARE COMPUTED BY USING TAYLOR SERIES. THE METHOD IS DESCRIBED IN REFERENCE [3].
FOR X>=3 THE PROCEDURE CALLS FOR THE PROCEDURE BESS PQA01 (SEE SUBSECTION BESS PQA01).
THE RELATIVE ACCURACY IS ABOUT "-13, EXCEPT FOR LARGE VALUES OF X; IN THAT CASE THE ACCURACY MAINLY DEPENDS ON THE ACCURACY OF THE FUNCTIONS SIN(X) AND COS(X).

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "REAL" P, Q;
    "PROCEDURE" BESS YA01(A, X, YA, YA1); "CODE" 35181;
    BESS YA01(0, 1, P, Q);
    OUTPUT(61, "("2(N)")", P, Q)
"END"
```

YIELDS THE FOLLOWING RESULTS

+8.8256964215677"-002 -7.8121282130028"-001.

SUBSECTION: BESS YAPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS YAPLUSN(A, X, NMAX, YAN); "VALUE" A, X, NMAX;
"REAL" A, X; "INTEGER" NMAX; "ARRAY" YAN;
"CODE" 35182;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER;
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X>0;
NMAX:   <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE INDICES OF THE ARRAY YAN;
YAN:    <ARRAY IDENTIFIER>;
        "ARRAY" YAN[0:NMAX]; NMAX>=0;
        EXIT: THE VALUES OF THE BESSEL FUNCTIONS OF
              THE SECOND KIND OF ORDER A+K, FOR THE ARGUMENT X
              ARE ASSIGNED TO YAN[K],0<=K<=NMAX.
```

PROCEDURES USED: BESS YA01 = CP 35181.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

THE RECURRENCE RELATION

YAN[N+1]= -YAN[N-1] + 2*(N+A)*YAN[N]/X

IS USED. THE INITIAL VALUES ARE OBTAINED FROM THE PROCEDURE BESS YA01. THE RECURRENCE RELATION IS NUMERICALLY STABLE IN THE FORWARD DIRECTION (IF A >= 0).

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "ARRAY" YAN[0:2];
    "PROCEDURE" BESS YAPLUSN(A, X, NMAX, YAN); "CODE" 35182;
    BESS YAPLUSN(0, 1, 2, YAN);
    OUTPUT(61, "("3(N)")", YAN[0], YAN[1], YAN[2])
"END"
```

YIELDS THE FOLLOWING RESULTS

+8.8256964215677"-002 -7.8121282130028"-001 -1.6506826068163"+000.

SUBSECTION: BESS PQA01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS PQA01(A, X, PA, QA, PA1, QA1); "VALUE" X, A;
"REAL" X, A, PA, QA, PA1, QA1;
"CODE" 35183;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARITHMETIC EXPRESSION>;
THE ORDER;

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;

PA: <VARIABLE>;
EXIT: THIS FUNCTION CORRESPONDS TO THE FUNCTION P(X, A) DEFINED ON P. 205 OF REFERENCE [4]. SEE ALSO REFERENCE [1], FORMULA 9.2.6;

QA: <VARIABLE>;
EXIT: THIS FUNCTION CORRESPONDS TO THE FUNCTION Q(X, A) DEFINED ON P.205 OF REFERENCE [4]. SEE ALSO REFERENCE [1], FORMULA 9.2.6;

```
    PA1:    <VARIABLE>;
            EXIT: THE FUNCTION P(X, A+1);
    QA1:    <VARIABLE>;
            EXIT: THE FUNCTION Q(X, A+1).
```

PROCEDURES USED:

```
    BESS JAPLUSN      =     CP35180,
    BESS YA01         =     CP35181.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

```
    X < 3 :
    PA, QA, PA1, QA1 ARE COMPUTED FROM THE RELATIONS

        PA  = B * (YA0 * S + JA0 * C),
        QA  = B * (YA0 * C - JA0 * S),

        PA1 = B * (JA1 * S - YA1 * C),
        QA1 = B * (JA1 * C + YA1 * S),

    WHERE

          B = SQRT(HALFPI * X),
          C = COS(X - HALFPI * (A + .5)),
          S = SIN(X - HALFPI * (A + .5)),
     HALFPI = 1.57079 63267 9489,
        YA0 = Y[A](X),
        YA1 = Y[A + 1](X),
        JA0 = J[A](X),
        JA1 = J[A + 1](X);

    X >= 3:
    THE METHOD IS DESCRIBED IN REFERENCE [3]. IT DEPENDS ON USING
    A MILLER ALGORITHM FOR CONFLUENT HYPERGEOMETRIC FUNCTIONS.
    THE ACCURACY IS ABOUT "-13 AND IS BETTER FOR LARGE X.
    THE FUNCTIONS PA AND QA CAN ALSO BE USED FOR THE COMPUTATION
    OF THE BESSEL FUNCTION J OF THE FIRST KIND.
    SEE REFERENCE[1], FORMULA 9.2.5.
```

EXAMPLE OF USE:

FROM SOME PROPERTIES OF THE BESSEL FUNCTIONS IT CAN BE PROVED THAT PA*PA1+QA*QA1=1, WHATEVER X AND A. IN THE FOLLOWING PROGRAM WE VERIFY THIS RELATION.

```
"BEGIN" "REAL" A, X, P, Q, R, S;
    "PROCEDURE" BESS PQA01(A, X, PA, QA, PA1, QA1); "CODE" 35183;
    "FOR" X:= 1, 3, 5, 10, 15, 20, 50 "DO"
    "BEGIN" BESS PQA01(0, X, P, Q, R, S);
        OUTPUT(61, "("BB, D.2D"+3D")", ABS(P*R+Q*S-1))
    "END"
"END"
```

THIS PROGRAM GIVES THE FOLLOWING RESULTS:

1.42"-014 7.11"-015 7.11"-015 7.11"-015 1.42"-014 0.00"+000 2.13"-014.

SUBSECTION: BESS ZEROS.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS ZEROS(A,N,Z,D);
"VALUE" A,N,D; "REAL" A;
"INTEGER" N,D; "ARRAY" Z;
"CODE" 35184;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER OF THE BESSEL FUNCTION, A>=0.
N:      <ARITHMETIC EXPRESSION>;
        THE NUMBER OF ZEROS TO BE EVALUATED, N>=1.
Z:      <ARRAY IDENTIFIER>;
        "ARRAY" Z[1:N];
        EXIT:  Z[J] IS THE J-TH ZERO OF THE
               SELECTED BESSEL FUNCTON;
D:      <ARITHMETIC EXPRESSION>;
        THE CHOICE OF D DETERMINES THE TYPE OF THE
        BESSEL FUNCTION OF WHICH THE ZEROS ARE COMPUTED:
        IF D=1 THEN JA       ,
        IF D=2 THEN YA       ,
        IF D=3 THEN JA-PRIME,
        IF D=4 THEN YA-PRIME.
```

PROCEDURES USED: BESS PQA01 = CP 35183.

REQUIRED CENTRAL MEMMORY: NO AUXILIARY ARRAYS ARE USED.

RUNNING TIME: DEPENDS ON THE VALUES OF A AND N AND ON THE MUMBER OF ITERATIONS IN THE ALGORITHM. FROM TESTS IT FOLLOWS THAT FOR EACH ZERO AT MOST 3 EVALUATIONS OF THE PROCEDURE BESS PQA01 ARE NEEDED.

METHOD AND PERFORMANCE:

A FIRST APPROXIMATION OF THE ZEROS OF THE SELECTED BESSEL FUNCTION IS CALCULATED BY MEANS OF THE ASYMPTOTIC EXPANTIONS (SEE THE FORMULAS 9.5.12, 9.5.13 (FOR A < 3) AND 9.5.22, 9.5.24(FOR A >= 3) OF REF [1]). THIS VALUE IS CORRECTED BY THE USE OF A FOURTH ORDER NEWTON-RAPHSON METHOD AS DISCRIBED ON P. 179 OF REF [6]. MORE DETAILS CAN BE FOUND IN REF [7].
A RELATIVE PRECISION OF 13 DIGITS IS PERSUED.
THE COMPUTATION OF A ZERO IS TERMINATED IF THIS ACCURRACY IS ACHIEVED OR IF MORE THAN 5 ITERATIONS ARE NEEDED.
THE PROCEDURE DOES NOT CHECK ON THE RANGE OF THE PARAMETERS A,N AND D.

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" A; "INTEGER" N,D; "ARRAY" Z[1:2];
   "PROCEDURE"BESS ZEROS(A,N,Z,D); "CODE" 35184;
   A:=3.14; N:= 2; D:= 2;
   BESS ZEROS(A,N,Z,D);
   OUTPUT(61,"M"N,/,N")",Z[1],Z[2])
"END"
```

PRINTS THE FIRST TWO ZEROS OF THE BESSEL FUNCTION Y OF THE ORDER 3.14; THE RESULT IS:

```
+4.68478470787 99"+000
+8.27658983383 92"+000
```

SUBSECTION: START.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"INTEGER" "PROCEDURE" START(X,N,T);
"VALUE" X,N,T; "REAL" X;
"INTEGER" N,T;
"CODE" 35185;
```

START:= A STARTING VALUE FOR THE MILLER ALGORITHM
FOR COMPUTING AN ARRAY OF BESSEL FUNCTIONS;

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTIONS, X > 0;

N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF BESSEL FUNCTIONS TO BE COMPUTED, N >= 0;

T: <ARITHMETIC EXPRESSION>;
THE TYPE OF BESSEL FUNCTION IN QUESTION,
T = 0 CORRESPONDS TO ORDINARY BESSEL FUNCTIONS,
T = 1 CORRESPONDS TO MODIFIED BESSEL FUNCTIONS.

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

THE PROCEDURE IS CALLED IN THE FOLLOWING PROCEDURES:

BESS J	CODE 35162
NON EXP BESS I	CODE 35177
BESS JAPLUSN	CODE 35180
BESS KAPLUSN	CODE 35192
NON EXP BESS IAPLUSN	CODE 35193
SPHER BESS J	CODE 35150
NON EXP SPHER BESS I	CODE 35154.

IN THESE PROCEDURES AN ARRAY OF BESSEL FUNCTIONS IS GENERATED BY USING MILLER 'S ALGORITHM (SEE REF[5]). FOR STARTING THIS ALGORITHM ONE NEEDS AN INTEGER NU WHICH CAN BE COMPUTED BY USING GAUTSCHI 'S ESTIMATES OF THE ERROR (SEE REF[5,FORMULA (5.11)]). WE COMPUTE THIS STARTING VALUE NU BY USING ASYMPTOTIC APPROXIMA- TIONS OF THE BESSEL FUNCTIONS, AS GIVEN IN REF[1, FORMULA 9.3.7, 9.3.8, 9.7.7, AND 9.7.8]. GAUTSCHI USED DIFFERENT FORMULAS, BUT THOSE USED HERE GIVE FOR LARGE X AND N MORE REALISTIC ESTIMATES. THE PERSUED ACCURACY IN THE ABOVE MENTIONED PROCEDURES IS ABOUT "-14 . FOR OBTAINING AN ACCURACY OF "-D THE NUMBERS 36 AND 18 APPEARING IN THE FOURTH AND SIXTH LINE OF THE SOURCE TEXT OF START SHOULD BE REPLACED BY (D+1)* LN(10) AND .5*(D+1)* LN(10), RESPECTIVELY. FOR MODIFIED BESSEL FUNCTIONS THE ACCURRACY IS IN A RELATIVE SENSE; FOR ORDINARY BESSEL FUNCTIONS THE ACCURRACY IS ABSOLUTE IF THE ORDER OF THE BESSEL FUNCTION IS SMALLER THAN X, OTHERWISE IT IS RELATIVE.

RUNNING TIME: NEGLECTABLE IF COMPARED WITH THE TIME NEEDED FOR THE BESSEL FUNCTION PROCEDURES.

EXAMPLE OF USE: SEE THE ABOVE MENTIONED PROCEDURES.

SOURCE TEXT(S):

```
"CODE" 35180;
  "PROCEDURE" BESS JAPLUSN(A, X, N, JA); "VALUE" A, X, N;
  "INTEGER" N; "REAL" X, A; "ARRAY" JA;
  "IF" X = 0 "THEN"
  "BEGIN" JA[0]:= "IF" A = 0 "THEN" 1 "ELSE" 0;
    "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" JA[N]:= 0
  "END" "ELSE"
  "IF" A = 0 "THEN"
  "BEGIN" "PROCEDURE" BESS J(X, N, J); "CODE" 35162;
    BESS J(X, N, JA)
  "END" "ELSE"
  "IF" A = .5 "THEN"
  "BEGIN" "REAL" S;
    "PROCEDURE" SPHER BESS J(X, N, J); "CODE" 35150;
    S:= SQRT(X) * .797 884 560 802 865; "COMMENT" S = SQRT(2X / PI);
    SPHER BESS J(X, N, JA);
    "FOR" N:= N "STEP" - 1 "UNTIL" 0 "DO" JA[N]:= JA[N] * S
  "END" "ELSE"
                                                          "COMMENT"
```

```
;

  "BEGIN" "REAL" A2, X2, R, S, L, LABDA; "INTEGER" K, M, NU;
    "REAL" "PROCEDURE" GAMMA(Y); "CODE" 35061;
    "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
    L:= 1; NU:= START(X,N,0);
    "FOR" M:= 1 "STEP" 1 "UNTIL" NU "DO"
    L:= L * (M+A) / (M+1); R:= S:= 0; X2:= 2 / X; K:= -1; A2:= A + A;
    "FOR" M:= NU+NU "STEP" - 1 "UNTIL" 1 "DO"
    "BEGIN" P:= 1 / (X2 * (A + M)    - R);
      "IF" K = 1 "THEN" LABDA:= 0 "ELSE"
      "BEGIN" L:= L * (M + 2) / (M + A2); LABDA:= L * (M + A) "END";
      S:= R * (LABDA + S); K:= -K;
      "IF" M<= N "THEN" JA[M]:= R
    "END";
    JA[0]:= R:= 1 / GAMMA(1 + A) / (1 + S) / X2 ** A;
    "FOR" M:= 1 "STEP" 1 "UNTIL" N "DO" JA[M]:= R:= R * JA[M];
  "END" BESS JAPLUSN;
       "EOP"

"CODE" 35181;
  "PROCEDURE" BESS YA01(A,X,YA,YA1);"VALUE" A,X; "REAL" A,X,YA,YA1;
  "IF" A = 0 "THEN"
  "BEGIN" "PROCEDURE" BESS Y01(X,Y0,Y1); "CODE" 35163;
    BESS Y01(X,YA,YA1)
  "END" "ELSE"
  "BEGIN" "REAL" B,C,D,E,F,G,H,P,PI,Q,R,S;"INTEGER" N,NA;
    "BOOLEAN" REC,REV;
    PI:=4*ARCTAN(1);NA:=ENTIER(A+.5);REC:=A>=.5;
    REV:=A<-.5;"IF" REV "OR" REC "THEN" A:=A-NA;
    "IF" A=-.5 "THEN"
    "BEGIN" P:=SQRT(2/PI/X);F:=P*SIN(X);G:=-P*COS(X) "END" "ELSE"
    "IF" X<3 "THEN"
    "BEGIN" "REAL" "PROCEDURE" RECIP GAMMA(X,ODD,EVEN); "CODE" 35060;
      "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
      B:=X/2;D:=-LN(B);E:=A*D;
      C:="IF" ABS(A)<"-8  "THEN" 1/PI "ELSE" A/SIN(A*PI);
      S:="IF" ABS(E)<"-8  "THEN" 1 "ELSE" SINH(E)/E;
      E:=EXP(E);G:=RECIP GAMMA(A,P,Q)*E;E:=(E+1/E)/2;
      F:=2*C*(P*E+Q*S*D);E:=A*A;
      P:=G*C;Q:=1/G/PI;C:=A*PI/2;
      R:="IF" ABS(C)<"-8  "THEN" 1 "ELSE" SIN(C)/C;R:=PI*C*R*R;
      C:=1;D:=-B*B;YA:=F+R*Q;YA1:=P;
      "FOR" N:=1,N+1 "WHILE"
      ABS(G/(1+ABS(YA)))+ABS(H/(1+ABS(YA1)))>"-15 "DO"
      "BEGIN" F:=(F*N+P+Q)/(N*N-E);C:=C*D/N;
        P:=P/(N-A);Q:=Q/(N+A);
        G:=C*(F+R*Q);H:=C*P-N*G;
        YA:=YA+G;YA1:=YA1+H;
      "END";
      F:=-YA;G:=-YA1/B
    "END" "ELSE"
                                                              "COMMENT"
```

```
    "BEGIN" "PROCEDURE" BESS PQA01(A,X,PA,QA,PA1,QA1); "CODE" 35183;
      B:=X-PI*(A+.5)/2;C:=COS(B);S:=SIN(B);
      D:=SQRT(2/X/PI);
      BESS PQA01(A,X,P,Q,B,H);
      F:=D*(P*S+Q*C);G:=D*(H*S-B*C)
    "END";
    "IF" REV "THEN"
    "BEGIN" X:=2/X;NA:=-NA-1;
      "FOR" N:=0 "STEP" 1 "UNTIL" NA "DO"
      "BEGIN" H:=X*(A-N)*F-G;G:=F;F:=H "END"
    "END" "ELSE" "IF" REC "THEN"
    "BEGIN" X:=2/X;
      "FOR" N:=1 "STEP" 1 "UNTIL" NA "DO"
      "BEGIN" H:=X*(A+N)*G-F;F:=G;G:=H "END"
    "END";
    YA:=F;YA1:=G
  "END" BESS YA01;
        "EOP"
"CODE" 35182;
  "PROCEDURE" BESS YAPLUSN(A, X, NMAX, YAN); "VALUE" A, X, NMAX;
  "REAL" A, X; "INTEGER" NMAX; "ARRAY" YAN;
  "BEGIN" "INTEGER" N; "REAL" Y1;
    "PROCEDURE" BESS YA01(A, X, YA, YA1); "CODE" 35181;
    BESS YA01(A, X, YAN[0], Y1); A:= A-1; X:= 2/X;
    "IF" NMAX > 0 "THEN" YAN[1]:= Y1;
    "FOR" N:= 2 "STEP" 1 "UNTIL" NMAX "DO"
    YAN[N]:= -YAN[N-2] + (A+N)*X*YAN[N-1]
  "END" BESS YAPLUSN;
        "EOP"


"CODE" 35183;
  "PROCEDURE" BESS PQA01(A,X,PA,QA,PA1,QA1);"VALUE" A,X;
  "REAL" A,X,PA,PA1,QA,QA1;
  "IF" A = 0 "THEN"
  "BEGIN" "PROCEDURE" BESS PQ0(X,P0,Q0); "CODE" 35165;
    "PROCEDURE" BESS PQ1(X,P1,Q1); "CODE" 35166;
    BESS PQ0(X,PA,QA); BESS PQ1(X,PA1,QA1)
  "END" "ELSE"
  "BEGIN" "INTEGER" N,NA;  "REAL" B, PI, P0, Q0 ; "BOOLEAN" REC, REV;
    PI:= 4 * ARCTAN(1);
    REV:=A<-.5;"IF" REV "THEN" A:=-A-1;
    REC:=A>=.5;"IF" REC "THEN"
    "BEGIN" NA:=ENTIER(A+.5);A:=A-NA "END";
    "IF" A=-.5 "THEN"
    "BEGIN" PA:=PA1:=1;QA:=QA1:=0 "END" "ELSE"
    "IF" X >= 3 "THEN"
                                                                   "COMMENT"
```

;

```
"BEGIN" "REAL" C,D,E,F,G,H,P,Q,R,S;
  C:=.25 - A*A; B:= X + X; F:= R:= 1; G:= -X; S:= 0;
  E:=(X*COS(A*PI)/PI*"15)**2;
  "FOR" N:=2,N+1 "WHILE" (P*P + Q*Q)*N*N<E "DO"
  "BEGIN" D:=(N-1+C/N);
    P:= (2 * N * F + B * G - D * R) / (N + 1);
    Q:= (2 * N * G - B * F - D * S) / (N + 1);
    R:= F; F:= P; S:= G; G:= Q
  "END";
  E:= F * F + G * G;
  P:= (R * F + S * G) / E;
  Q:= (S * F - R * G) / E;
  F:= P; G:= Q;
  "FOR" N:=N-1 "WHILE" N>0 "DO"
  "BEGIN" R:=(N+1)*(2-P)-2;S:=B+(N+1)*Q;D:=(N-1+C/N)/
    (R*R+S*S);P:=D*R;Q:=D*S;E:=F;
    F:=P*(E+1)-G*Q;G:=Q*(E+1)+P*G
  "END";
  F:=1+F; D:=F*F + G*G;
  PA:=F/D;QA:=-G/D;D:=A+.5-P;Q:=Q+X;
  PA1:=(PA*Q-QA*D)/X; QA1:=(QA*Q+PA*D)/X
"END" "ELSE"
"BEGIN" "REAL" C, S, CHI, YA, YA1; "ARRAY" JA[0:1];
  "PROCEDURE" BESS JAPLUSN(A, X, N, JA); "CODE" 35180;
  "PROCEDURE" BESS YA01(A, X, YA, YA1); "CODE" 35181;
  B:= SQRT(PI * X / 2);
  CHI:= X - PI * (A / 2 + .25); C:= COS(CHI); S:= SIN(CHI);
  BESS YA01(A, X, YA, YA1); BESS JAPLUSN(A, X, 1, JA);
  PA:= B * (YA * S + C * JA[0]); QA:= B * (C * YA - S * JA[0]);
  PA1:= B * (S * JA[1] - C * YA1);
  QA1:= B * (JA[1] * C + YA1 * S)
"END";
"IF" REC "THEN"
"BEGIN"  X:=2/X;B:=(A+1)*X;
  "FOR" N:=1 "STEP" 1 "UNTIL" NA "DO"
  "BEGIN" P0:=PA-QA1*B; Q0:=QA+PA1*B;
    PA:=PA1;PA1:=P0; QA:=QA1; QA1:=Q0; B:=B+X
  "END"
"END";
"IF" REV "THEN"
"BEGIN" P0:=PA1;PA1:=PA;PA:=P0;Q0:=QA1;QA1:=QA;QA:=Q0 "END"
"END" BESS PQA01;
    "EOP"
```

```
CODE" 35184;
PROCEDURE" BESS ZEROS(A,N,Z,D); "VALUE" A,N,D; "REAL" A;"ARRAY" Z;
                                    "INTEGER" N,D;
COMMENT" COMPUTES Z[1],...Z[N],THE FIRST N ZEROS OF A BESSEL FUNCTION.
   THE CHOICE OF D DETERMINES THE TYPE OF THE BESSEL FUNCTION :
   IF D=1 THEN JA       ELSE
   IF D=2 THEN YA       ELSE
   IF D=3 THEN JA-PRIME ELSE
   IF D=4 THEN YA-PRIME.
   A IS THE ORDER OF THE BESSEL FUNCTION, IT MUST BE NON-NEGATIVE.;
BEGIN" "REAL" AA,A2,B,BB,C,CHI,CO,MU,MU2,MU3,MU4,P,PI,PA,PA1,P0,P1,PP1,
          Q,QA,QA1,Q1,QQ1,R0,SI,T,TT,U,V,W,X,XX,X4,Y; "INTEGER" J,S;

   "REAL" "PROCEDURE" FI(Y); "VALUE" Y; "REAL" Y;
   "COMMENT" COMPUTES FI FROM THE EQUATION
         TAN(FI)-FI=Y , WHERE Y>=0.
   THE RELATIVE ACCURACY IS AT LEAST 5 DIGITS;
   "IF" Y=0  "THEN" FI:=0            "ELSE"
   "IF" Y>"5 "THEN" FI:=1.570796   "ELSE"
   "BEGIN" "REAL" R,P,PP;
       "IF" Y<1 "THEN"
       "BEGIN" P:=(3*Y)**(1/3); PP:=P*P;
           P:=P*(1+PP*(-210+PP*(27-2*PP))/1575)
       "END" "ELSE"
       "BEGIN" P:=1/(Y+1.570796); PP:=P*P;
           P:= 1.570796-P*(1+PP*(2310+PP*(3003+PP*(4818+PP*
                 (8591+PP*16328))))/3465)
       "END";
       PP:=(Y+P)*(Y+P); R:=(P-ARCTAN(P+Y))/PP;
       FI:=P-(1+PP)*R*(1+R/(P+Y))
   "END" FI;

   "PROCEDURE" BESS PQA01(A,X,PA,QA,PA1,QA1); "CODE" 35183;

   "REAL" "PROCEDURE" R;
   "BEGIN" BESS PQA01(A,X,PA,QA,PA1,QA1);
       CHI:=X-PI*(A/2+0.25);
       SI :=SIN(CHI); CO:=COS(CHI);
       R:= "IF" D=1 "THEN" (PA*CO-QA*SI)/(PA1*SI+QA1*CO) "ELSE"
           "IF" D=2 "THEN" (PA*SI+QA*CO)/(QA1*SI-PA1*CO) "ELSE"
           "IF" D=3 "THEN" A/X-(PA1*SI+QA1*CO)/(PA*CO-QA*SI) "ELSE"
                           A/X-(QA1*SI-PA1*CO)/(PA*SI+QA*CO)
   "END" R;
                                                            "COMMENT"
```

```
;
    PI:=4*ARCTAN(1); AA:=A*A; MU:=4*AA; MU2:=MU*MU;
    MU3:=MU*MU2; MU4:=MU2*MU2;
    "IF" D<3 "THEN"
    "BEGIN" P:=7*MU-31; P0:=MU-1;
        P1:=4*(253*MU2-3722*MU+17869)/15/P*P0;
        Q1:=8*( 83*MU2- 982*MU+ 3779)/ 5/P
    "END" "ELSE"
    "BEGIN" P:=7*MU2+82*MU-9; P0:=MU+3;
        P1:=(4048*MU4+131264*MU3-221984*MU2-417600*MU+1012176)/60/P;
        Q1:=1.6*(83*MU3+2075*MU2-3039*MU+3537)/P
    "END";
    T:="IF" D=1"OR"D=4 "THEN" 0.25 "ELSE" 0.75; TT:=4*T;
    "IF" D<3 "THEN"
    "BEGIN" PP1:= 5/48; QQ1:= -5/36  "END" "ELSE"
    "BEGIN" PP1:=-7/48; QQ1:= 35/288 "END";
    Y:= 3*PI/8; BB:= "IF" A>=3 "THEN" A **(-2/3) "ELSE" 0.0 ;
    "FOR" S:=1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" "IF" A=0"AND"S=1"AND"D=3 "THEN"
        "BEGIN" X:=0; J:=0 "END" "ELSE"
        "BEGIN" "IF" S >= 3*A -8  "THEN"
            "BEGIN" B:=(S+A/2-T)*PI; C:=1/B/B/64;
                X:=B-1/B/8*(P0-P1*C)/(1-Q1*C)
            "END" "ELSE"
            "BEGIN" "IF" S=1 "THEN"
                "BEGIN" X:= "IF" D=1 "THEN" -2.33811 "ELSE"
                           "IF" D=2 "THEN" -1.17371 "ELSE"
                           "IF" D=3 "THEN" -1.01879 "ELSE" -2.29444
                "END" "ELSE"
                "BEGIN" X:= Y*(4*S-TT); V:= 1/X/X;
                    X:= -X**(2/3)*(1+V*(PP1+QQ1*V))
                "END";
                U:=X*BB; V:=FI(2/3*(-U)**1.5);
                W:=1/COS(V); XX:=1-W*W; C:=SQRT(U/XX);
                X:=W*(A+C/A/U*
                ("IF" D<3 "THEN" -5/48/U-C*(-5/24/XX+1/8)
                         "ELSE"  7/48/U+C*(-7/24/XX+3/8)))
            "END";  J:=0;
            L1: XX:=X*X; X4:=XX*XX; A2:=AA-XX; R0:=R; J:=J+1;
                "IF" D<3 "THEN"
                "BEGIN" U:=R0; P:=(1-4*A2)/6/X/(2*A+1);
                    Q:=(2*(XX-MU)-1-6*A)/3/X/(2*A+1)
                "END" "ELSE"
                "BEGIN" U:=-XX*R0/A2; V:=2*X*A2/(AA+XX)/3;
                    W:=A2*A2*A2;
                    Q:=V*(1+( MU2+32*MU*XX+48*X4)/32/W);
                    P:=V*(1+(-MU2+40*MU*XX+48*X4)/64/W)
                "END";
            W:=U*(1+P*R0)/(1+Q*R0); X:=X+W;
            "IF" ABS(W/X)>"-13 "AND"J<5 "THEN" "GOTO" L1
        "END"; Z[S]:=X
    "END"
"END" BESS ZEROS;
     "EOP"
```

```
CODE" 35185;
INTEGER" "PROCEDURE" START(X,N,T); "VALUE" X,N,T; "REAL" X;
                                   "INTEGER" N,T;
BEGIN" "REAL"P,Q,R,Y;   "INTEGER" S;
   S:= 2*T-1; P:= 36/X-T; R:= N/X; "IF" R>1 "OR" T=1 "THEN"
   "BEGIN" Q:= SQRT(R*R+S); R:= R*LN(Q+R)-Q "END" "ELSE" R:= 0;
   Q:= 18/X+R; R:= "IF" P>Q "THEN" P "ELSE" Q;
   P:= SQRT(2*(T+R)); P:= X*((1+R)+P)/(1+P); Y:= 0;
   "FOR" Q:= Y, Y "WHILE" P>Q "OR" P<Q-1 "DO"
   "BEGIN" Y:=P; P:= P/X; Q:= SQRT(P*P+S); P:= X*(R+Q)/LN(P+Q) "END";
   START:= "IF" T=1 "THEN" ENTIER(P+1) "ELSE" -ENTIER(-P/2)*2
END" START;
     "EOP"
```

AUTHORS: M.BAKKER AND N.M.TEMME.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 781101.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE PROCEDURES:

BESS IAPLUSN:

THIS PROCEDURE GENERATES AN ARRAY OF MODIFIED BESSELFUNCTIONS OF THE FIRST KIND OF ORDER A+N, N=0, ..., NMAX, 0 < = A < 1 AND ARGUMENT X > = 0.

NONEXP BESS IAPLUSN:

THIS PROCEDURE GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS OF THE FIRST KIND OF ORDER A + N, N = 0, ..., NMAX, 0<=A <1 AND ARGUMENT X > = 0 MULTIPLIED BY THE FACTOR EXP(-X). THUS, APART FROM THE EXPONENTIAL FACTOR THE ARRAY ENTRIES ARE THE SAME AS THOSE COMPUTED BY BESS IAPLUSN.

BESS KA01:

THIS PROCEDURE CALCULATES THE MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDER A AND A+1, AND ARGUMENT X, X>0;

BESS KAPLUSN:

THIS PROCEDURE GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDER A+N, N=0, 1, ..., NMAX, AND ARGUMENT X>0.
THE MODIFIED BESSEL FUNCTIONS CORRESPOND TO THE FUNCTION DEFINED IN FORMULA 9.6.2 OF REFERENCE[1];

NONEXP BESS KA01:

THIS PROCEDURE CALCULATES THE MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDER A AND A + 1, AND ARGUMENT X, X > 0, MULTIPLIED BY THE FACTOR EXP(X). THUS, APART FROM THE EXPONENTIAL FACTOR, THE FUNCTIONS ARE THE SAME AS THOSE COMPUTED BY BESS KA01;

NONEXP BESS KAPLUSN:

THIS PROCEDURE GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND OF ORDER A + N, N = 0, 1,..., NMAX, AND ARGUMENT X>0 MULTIPLIED BY THE FACTOR EXP(X). THUS, APART FROM THE EXPONENTIAL FACTOR, THE FUNCTIONS ARE THE SAME AS THOSE COMPUTED BY THE PROCEDURE BESS KAPLUSN.

KEYWORDS:

BESSEL FUNCTION,
MODIFIED BESSEL FUNCTION,
MODIFIED BESSEL FUNCTION OF THE THIRD KIND.

REFERENCES:

[1]. ABRAMOWITZ, M., AND STEGUN, I. (EDS.), HANDBOOK OF MATHEMATICAL FUNCTIONS WITH FORMULAS, GRAPHS AND MATHEMATICAL TABLES. APPL. MATH. SER. 55, U.S. GOVT. PRINTING OFFICE, WASHINGTON, D.C. (1964).

[2]. GAUTSCHI, W., COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS. SIAM REVIEW, VOLUME 9, (1967), NUMBER 1, P.24.

[3]. TEMME, N.M., ON THE NUMERICAL EVALUATION OF THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND. J. COMP. PHYSICS, VOL. 19, (1975), NUMBER 3, P. 324.

SUBSECTION: BESS IAPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS IAPLUSN(A, X, N, IA);
"VALUE" X, N, A; "REAL" X, A;
"INTEGER" N; "ARRAY" IA;
"CODE" 35190;
```

```
THE MEANING OF THE FORMAL PARAMETERS IS:
A:  < ARITHMETIC EXPRESSION >;
    THE NONINTEGER PART OF THE ORDER OF THE
    BESSEL FUNCTIONS; 0 < = A < 1;
X:  < ARITHMETIC EXPRESSION >;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > = 0;
N:  < ARITHMETIC EXPRESSION >;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY IA; N>= 0;
IA: < ARRAY IDENTIFIER >;
    "ARRAY" IA[0:N]; N > = 0;
    EXIT: THE VALUES OF THE MODIFIED BESSEL FUNCTIONS
          OF THE FIRST KIND , OF ORDER A+K AND ARGUMENT X,
          I[A+K](X) ARE ASSIGNED TO THE ARRAY IA.
```

PROCEDURES USED:

```
NONEXP BESS IAPLUSN     =  CP 35193,
BESS I                  =  CP 35172,
NONEXP SPHER BESS I     =  CP 35154.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE: SEE SUBSECTION NONEXP BESS IAPLUSN.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO THE MAXIMUM OF X AND N.

EXAMPLE OF USE:

```
"BEGIN" "REAL" X, A; "ARRAY" IA[0:2]  ;
    "PROCEDURE" BESS IAPLUSN(A, X, N, IA); "CODE" 35190;
    A:= .25; X:= 2;    BESS IAPLUSN(A, X, 2, IA);
    OUTPUT(61,"("2(4BD.DD),/,3(4B-.14D"-ZD)")",
    A, X, IA[0], IA[1], IA[2])
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
0.25    2.00
.22033544516736" 1   .13401967589829" 1   .52810850294501" 0
```

SUBSECTION: NONEXP BESS IAPLUSN.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:

    "PROCEDURE" NONEXP BESS IAPLUSN(A, X, N, IA);
    "VALUE" A, X, N;
    "REAL" A, X; "INTEGER" N; "ARRAY" IA;
    "CODE" 35193;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    A:  < ARITHMETIC EXPRESSION >;
        THE NONINTEGER PART OF THE ORDER A+N, 0 < = A < 1;
    X:  < ARITHMETIC EXPRESSION >;
        THE ARGUMENT OF THE BESSEL FUNCTIONS; X >= 0;
    N:  < ARITHMETIC EXPRESSION >;
        THE UPPER BOUND OF THE INDICES OF THE ARRAY IA; N>= 0;
    IA:  < ARRAY IDENTIFIER >;
        "ARRAY" IA[0:N]; N > = 0;
        EXIT:  IA[K] HAS THE VALUE OF
               THE MODIFIED BESSEL FUNCTION OF THE FIRST KIND OF
               ORDER A + K AND ARGUMENT X MULTIPLIED BY
               EXP (-X), 0 < = K < = N.
```

PROCEDURES USED:

```
    NONEXP BESS I          =  CP 35177
    NONEXP SPHER BESS I    =  CP 35154
    GAMMA                  =  CP 35061
    START                  =  CP 35185
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

IN ALL THE CASES THE BESSEL FUNCTIONS ARE COMPUTED ACCORDING TO THE MILLER METHOD DESCRIBED IN [2, P.46-52]. THE STARTING VALUE IS COMPUTED BY THE PROCEDURE START (SECTION 6.10.1).

RUNNING TIME: ROUGHLY PROPORTIONAL TO THE MAXIMUM OF X AND N.

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X, A; "ARRAY" IA[0:2];
"PROCEDURE" NON EXPBESS IAPLUSN(A, X, N, IA); "CODE" 35193;
    A:= .25; X:= 2; NON EXPBESS IAPLUSN(A, X, 2, IA);
    OUTPUT(61,"("2(4BD.DD),/,3(4B-.14D"-ZD)")",
    A, X, IA[0], IA[1], IA[2])
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
0.25    2.00
.29819159878790"  0   .18137590796974"  0   .71471713825726"  -1
```

SUBSECTION: BESS KA01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS KA01(A, X, KA, KA1); "VALUE" A, X;
"REAL" A, X, KA, KA1;
"CODE" 35191;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARITHMETIC EXPRESSION>;
THE ORDER;

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;

KA: <VARIABLE>;
EXIT: THE VALUE OF THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND OF ORDER A AND ARGUMENT X;

KA1: <VARIABLE>;
EXIT: THE VALUE OF THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND OF ORDER A+1 AND ARGUMENT X.

PROCEDURES USED:

```
RECIP GAMMA         = CP 35060;
NONEXP BESS KA01    = CP 35194;
SINH                = CP 35111.
```

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRYS ARE DECLARED.

METHOD AND PERFORMANCE:

FOR 0<X<1 THE BESSEL FUNCTIONS ARE COMPUTED BY USING TAYLOR SERIES. THE METHOD IS DESCRIBED IN REFERENCE [3]. FOR X>=1 THE PROCEDURE CALLS FOR THE PROCEDURE NONEXP BESS KA (SEE SUBSECTION NONEXP BESS KA). THE RELATIVE ACCURACY IS ABOUT "-13, EXCEPT FOR LARGE VALUES OF X; IN THAT CASE THE ACCURACY ALSO DEPENDS ON THE RELATIVE ACCURACY OF THE EXPONENTIAL FUNCTION. IF ONE IS INTERESTED IN THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND TIMES THE FACTOR EXP(X), THE PROCEDURE NONEXP BESS KA SHOULD BE USED.

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "REAL" P, Q;
    "PROCEDURE" BESS KA01(A, X, KA, KA1); "CODE" 35191;
    BESS KA01(0, 1, P, Q);
    OUTPUT(61, "("2(N)")", P, Q)
"END"
```

YIELDS THE FOLLOWING RESULTS

```
+4.2102443824071"-001   +6.0190723019724"-001.
```

SUBSECTION: BESS KAPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" BESS KAPLUSN(A, X, NMAX, KAN); "VALUE" A, X, NMAX;
"INTEGER" NMAX; "REAL" A, X; "ARRAY" KAN;
"CODE" 35192;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER. IT IS ADVISED TO TAKE A >= 0;
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;
NMAX:   <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE INDICES OF THE ARRAY KAN;
KAN:    <ARRAY IDENTIFIER>;
        "ARRAY" KAN[0:NMAX]; NMAX>=0;
        EXIT: THE VALUE OF THE MODIFIED BESSEL FUNCTION
              OF THE THIRD KIND OF ORDER N+A IS ASSIGNED TO KAN[N],
              0 <= N <= NMAX.
```

PROCEDURES USED: BESS KA01 = CP 35191.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

THE RECURRENCE RELATION KAN[N+1]=KAN[N-1]+2*(N+A)*KAN[N]/X IS USED. THE STARTING VALUES ARE OBTAINED FROM THE PROCEDURE BESS KA01. IF A>=0, RECURSION IS NUMERICALLY STABLE IN THE FORWARD DIRECTION. IF ONE IS INTERESTED IN THE MODIFIED BESSEL FUNCTIONS OF THE THIRD KIND TIMES THE FACTOR EXP(X), THE PROCEDURE NONEXP BESS KAPLUSN SHOULD BE USED.

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "ARRAY" KAN[0:2];
"PROCEDURE" BESS KAPLUSN(A, X, NMAX, KAN); "CODE" 35193;
    BESS KAPLUSN(0, 1, 2, KAN);
    OUTPUT(61, "("3(N)")", KAN[0], KAN[1], KAN[2])
"END"
```

YIELDS THE FOLLOWING RESULTS

+4.2102443824071"-001 +6.0190723019724"-001 +1.6248388986352"+000.

SUBSECTION: NONEXP BESS KA01.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NONEXP BESS KA01(A, X, KA, KA1);
"VALUE" A, X; "REAL" A, X, KA, KA1;
"CODE" 35194;
```

```
THE MEANING OF THE  FORMAL PARAMETERS IS:
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER;
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;
KA:     <VARIABLE>;
        EXIT: KA HAS THE VALUE OF THE MODIFIED BESSEL
              FUNCTION OF THE THIRD KIND OF ORDER A
              AND ARGUMENT X TIMES THE FACTOR EXP(X);
KA1:    <VARIABLE>;
        EXIT: THE VALUE OF THE MODIFIED BESSEL FUNCTION OF THE
              THIRD KIND OF ORDER A+1 AND ARGUMENT X TIMES THE
              FACTOR EXP(X).
```

PROCEDURES USED: BESS KA01 = CP 35191.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

FOR 0<X<1 THE PROCEDURE NONEXP BESS KA CALLS FOR THE PROCEDURE BESS KA01. FOR X>=1 THE BESSEL FUNCTIONS ARE COMPUTED WITH A MILLER ALGORITHM FOR CONFLUENT HYPERGEOMETRIC FUNCTIONS. THE METHOD IS DESCRIBED IN REFERENCE [3]. FOR ALL VALUES OF X CONSIDERED (X>0) THE FUNCTIONS DELIVERED ARE EQUAL TO THE VALUES COMPUTED BY THE PROCEDURE BESS KA01, APART FROM AN EXPONENTIAL FACTOR. THE RELATION BETWEEN THE TWO PROCEDURES WILL BE DESCRIBED BY THE PROGRAM:

```
"BEGIN" "REAL" A, X, KA, NEKA, KA1, NEKA1;
    "PROCEDURE" BESS KA01(A, X, KA, KA1); "CODE" 35191;
    "PROCEDURE" NONEXP BESS KA(A, X, KA, KA1); "CODE" 35194;
    A:= .3; X:= 3.14;
    BESS KA01(A, X, KA, KA1);
    NONEXP KA 01(A, X, NEKA, NEKA1)
"END"
```

THEN WE HAVE

KA = EXP(-X)*NEKA, KA1 = EXP(-X)*NEKA1. THE RELATIVE ACCURACY IS ABOUT "-13.

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "REAL" P, Q;
    "PROCEDURE" NONEXP BESS KA(A, X, KA, KA1); "CODE" 35194;
    NONEXP BESS KA(0, 2, P, Q);
    OUTPUT(61, "("2(N)")", P, Q)
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
8.4156821507078"-001 +1.0334768470687"+000.
```

SUBSECTION: NONEXP BESS KAPLUSN.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NONEXP BESS KAPLUSN(A, X, NMAX, KAN);
"VALUE" A, X, NMAX; "REAL" A, X; "INTEGER" NMAX; "ARRAY" KAN;
"CODE" 35195;
```

NONEXP BESS KAPLUSN GENERATES AN ARRAY OF MODIFIED BESSEL FUNCTIONS THE THIRD KIND OF ARGUMENT X AND ORDERS A+N, N=0, 1,..., NMAX TIMES THE FACTOR EXP(X).

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER. IT IS ADVISED TO TAKE A >= 0;
X:      <ARITHMETIC EXPRESSION>;
        THE ARGUMENT. THIS ARGUMENT SHOULD SATISFY X>0;
NMAX:   <ARITHMETIC EXPRESSION>;
        THIS PARAMETER SHOULD SATISFY NMAX>=0; NMAX INDICATES THE
        MAXIMUM NUMBER OF FUNCTION VALUES TO BE GENERATED;
KAN:    <ARRAY IDENTIFIER>;
        "ARRAY" KAN[0:NMAX]; NMAX>=0;
        EXIT: KAN[N] IS THE MODIFIED BESSEL FUNCTION OF THE THIRD
              KIND OF ORDER N+A AND OF ARGUMENT X (N=0(1)NMAX)
              TIMES THE FACTOR EXP(X).
```

PROCEDURES USED: NONEXP BESS KA = CP 35194.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

THE RECURRENCE RELATION KAN[N+1]=KAN[N-1]+2*(N+A)*KAN[N]/X IS USED. THE STARTING VALUES ARE OBTAINED FROM THE PROCEDURE NONEXP BESS KA. IF A>=0, RECURSION IS NUMERICALLY STABLE IN THE FORWARD DIRECTION. FOR ALL VALUES OF X AND NMAX CONSIDERED (X>0) THE FUNCTIONS DELIVERED ARE EQUAL TO THE VALUES COMPUTED BY THE PROCEDURE BESS KAPLUSN,APART FROM AN EXPONENTIAL FACTOR. THE RELATION BETWEEN THE TWO PROCEDURES WILL BE DESCRIBED BY THE PROGRAM:

```
"BEGIN" "REAL" X, A; "ARRAY" KA, NEKA[0:10];
    "PROCEDURE" BESS KAPLUSN(A, X, NMAX, KA); "CODE" 35193;
    "PROCEDURE" NONEXP BESS KAPLUSN(A, X, NMAX, KAN); "CODE" 35195;
    X:= 2.78; A:= .96;
    BESS KAPLUSN(A, X, 10, KA);
    NONEXP BESS KAPLUSN(A, X, 10, NEKA)
"END"
```

THEN WE HAVE KA[N] = EXP(-X)*NEKA[N], N=0, 1, ..., 10.

EXAMPLE OF USE:

THE PROGRAM:

```
"BEGIN" "ARRAY" KAN[0:2];
"PROCEDURE" NONEXP BESS KAPLUSN(A, X, NMAX, KAN); "CODE" 35195;
     NONEXP BESS KAPLUSN(0, 5, 2, KAN);
     OUTPUT(61, "("3(N)")", KAN[0], KAN[1], KAN[2])
"END"
```

YIELDS THE FOLLOWING RESULTS:

+5.4780756431352"-001 +6.0027385878831"-001 +7.8791710782884"-001.

SOURCE TEXT(S):

```
"CODE" 35190;
  "COMMENT" COMPUTATION OF I[A](X),    , I[N+A](X);
  "PROCEDURE" BESS IAPLUSN(A, X, N, IA); "VALUE" A, X, N;
  "INTEGER" N; "REAL" X, A; "ARRAY" IA;
  "IF" X= 0 "THEN"
  "BEGIN" IA[0]:= "IF" A= 0 "THEN" 1 "ELSE" 0;
    "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" IA[N]:= 0
  "END" "ELSE" "IF" A= 0 "THEN"
  "BEGIN" "PROCEDURE" BESS I(X, N, I); "CODE" 35172;
    BESS I(X, N, IA);
  "END" "ELSE" "IF" A= .5 "THEN"
  "BEGIN" "REAL" C;
    "PROCEDURE" NONEXP SPHER BESSI(X, N, I); "CODE" 35154;
    C:= .797 884 560 802 865 * SQRT(ABS(X)) * EXP (ABS (X));
    NONEXP SPHER BESSI(X, N, IA);
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" IA[N]:= C*IA[N]
  "END" "ELSE"
  "BEGIN" "REAL" EXPX;
    "PROCEDURE" NONEXP BESS IAPLUSN(A, X, N, IA); "CODE" 35193;
    EXPX:= EXP(ABS(X));
    NONEXP BESS IAPLUSN(A, X, N, IA);
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" IA[N]:= EXPX * IA[N]
  "END" BESS IAPLUSN;
        "EOP"

"CODE" 35191;
  "PROCEDURE"  BESS KA01(A, X, KA, KA1); "VALUE" A, X;
  "REAL" A, X, KA, KA1;
  "IF" A = 0 "THEN"
  "BEGIN" "PROCEDURE" BESS K01(X,K0,K1); "CODE" 35173;
    BESS K01(X,KA,KA1)
  "END" "ELSE"
  "BEGIN" "REAL" F, G, H, PI; "INTEGER" N, NA; "BOOLEAN" REC, REV;
    PI:= 4 * ARCTAN(1);
    REV:= A < -.5; "IF" REV "THEN" A:= -A-1;
    REC:= A >=  .5; "IF" REC "THEN"
    "BEGIN" NA:=ENTIER(A+.5); A:= A - NA "END";
    "IF" A = .5 "THEN" F:= G:= SQRT(PI / X / 2) * EXP (-X) "ELSE"
    "IF" X < 1 "THEN"
    "BEGIN"                                                "COMMENT"
```

```
    "REAL" A1, B, C, D, E, P, Q, S;
      "REAL" "PROCEDURE" RECIP GAMMA(X, ODD, EVEN); "CODE" 35060;
      "REAL" "PROCEDURE" SINH(X);            "CODE" 35111;
      B:=X/2;D:=-LN(B);E:=A*D;C:=A*PI;
      C:="IF" ABS(C)<"-15 "THEN" 1 "ELSE" C/SIN(C);
      S:="IF" ABS(E)<"-15 "THEN" 1 "ELSE" SINH(E)/E;
      E:=EXP(E);A1:=(E+1/E)/2;G:=RECIP GAMMA(A,P,Q)*E;
      KA:=F:=C*(P*A1+Q*S*D);E:=A*A;
      P:=.5*G*C;Q:=.5/G;C:=1;D:=B*B;KA1:=P;
      "FOR" N:=1,N+1 "WHILE" H/KA+ABS(G)/KA1>"-15 "DO"
      "BEGIN" F:=(F*N+P+Q)/(N*N-E);C:=C*D/N;
        P:=P/(N-A);Q:=Q/(N+A);G:=C*(P-N*F);
        H:=C*F;KA:=KA+H;KA1:=KA1+G
      "END";
      F:=KA;G:=KA1/B
    "END" "ELSE"
    "BEGIN" "REAL" EXPON;
      "PROCEDURE" NONEXP BESS KA01(A, X, KA, KA1); "CODE" 35194;
      EXPON:= EXP(-X); NONEXP BESS KA01(A, X, KA, KA1);
      F:= EXPON * KA; G:= EXPON * KA1
    "END";
    "IF" REC "THEN"
    "BEGIN" X:= 2 / X;
      "FOR" N:= 1 "STEP" 1 "UNTIL" NA "DO"
      "BEGIN" H:= F + (A + N) * X * G; F:= G; G:= H "END"
    "END";
    "IF" REV "THEN" "BEGIN" KA1:= F; KA:= G "END" "ELSE"
    "BEGIN" KA:= F; KA1:= G "END"
  "END" BESS KA01;
        "EOP"

"CODE" 35192;
  "PROCEDURE" BESS KAPLUSN(A, X, NMAX, KAN); "VALUE" A, X, NMAX;
  "REAL" A, X; "INTEGER" NMAX; "ARRAY" KAN;
  "BEGIN" "INTEGER" N; "REAL" K1;
    "PROCEDURE" BESS KA01(A, X, KA, KA1);"CODE" 35191;
    BESS KA01(A, X, KAN[0], K1); A:= A-1; X:= 2/X;
    "IF" NMAX > 0 "THEN" KAN[1]:= K1;
    "FOR" N:= 2 "STEP" 1 "UNTIL" NMAX "DO"
    KAN[N]:= KAN[N-2] + (A+N) * X * KAN[N-1]
  "END" BESS KAPLUSN;
        "EOP"
```

```
"CODE" 35193;
  "COMMENT" COMPUTATION OF NONEXPONENTIAL MODIFIED BESSEL
  FUNCTIONS OF FRACTIONAL ORDER;
  "PROCEDURE" NONEXP BESS IAPLUSN(A, X, N, IA); "VALUE" A, X, N;
  "REAL" X, A; "INTEGER" N; "ARRAY" IA;
  "IF" X= 0 "THEN"
  "BEGIN" IA[0]:= "IF" A= 0 "THEN" 1 "ELSE" 0;
    "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" IA[N]:= 0 "END"
  "ELSE" "IF" A= 0 "THEN"
  "BEGIN" "PROCEDURE" NONEXP BESSI(X, N, I); "CODE" 35177;
    NONEXP BESSI(X, N, IA)
  "END" "ELSE" "IF" A= .5 "THEN"
  "BEGIN" "REAL" C;
    "PROCEDURE" NONEXP SPHER BESSI(X, N, I); "CODE" 35154;
    C:= .797 884 560 802 865 * SQRT(X);
    NONEXP SPHER BESSI(X, N, IA);
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" IA[N]:= C * IA[N]
  "END" "ELSE"
  "BEGIN" "INTEGER" M, NU; "REAL" R, S, LABDA, L, A2, X2;
    "REAL" "PROCEDURE" GAMMA(X); "CODE" 35061;
    "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
    A2:= A+A; X2:= 2/X; L:=1;
    NU:= START(X,N,1) ; R:= S:= 0;
    "FOR" M:= 1 "STEP" 1 "UNTIL" NU "DO" L:= L * (M+A2)/(M+1);
    "FOR" M:= NU "STEP" -1 "UNTIL" 1 "DO"
    "BEGIN" R:= 1/(X2 *(A+M) +R); L:= L*(M+1)/(M+A2);
      LABDA:= L*(M+A) * 2; S:= R * (LABDA + S);
      "IF" M  <=  N "THEN" IA[M]:= R
    "END";
    IA[0]:= R:= 1/(1+S)/GAMMA(1+A)/X2 **A;
    "FOR" M:= 1 "STEP" 1 "UNTIL" N "DO" IA[M]:= R:= IA[M] * R;
  "END";
        "EOP"

"CODE" 35194;
  "PROCEDURE" NONEXP BESS KA01(A, X, KA, KA1); "VALUE" A, X;
  "REAL" A, X, KA, KA1;
  "IF" A = 0 "THEN"
  "BEGIN" "PROCEDURE" NONEXP BESS K01(X,K0,K1); "CODE" 35178;
    NONEXP BESS K01(X,KA,KA1)
  "END" "ELSE"
  "BEGIN" "REAL" F, G, H, PI; "INTEGER" N, NA; "BOOLEAN" REC, REV;
    PI:= 4 * ARCTAN(1);
    REV:= A < -.5; "IF" REV "THEN" A:= -A-1;
    REC:= A >=  .5; "IF" REC "THEN"
    "BEGIN" NA:=ENTIER(A+.5); A:= A - NA "END";
    "IF" A = -.5 "THEN" F:= G:= SQRT(PI / X / 2) "ELSE"
    "IF" X < 1 "THEN"
    "BEGIN"                                                 "COMMENT"
```

```
        "REAL" EXPON;
        "PROCEDURE" BESS KA01(A, X, KA, KA1);"CODE" 35191;
        EXPON:= EXP(X); BESS KA01(A, X, KA, KA1);
        F:= EXPON * KA; G:= EXPON * KA1
      "END" "ELSE"
      "BEGIN" "REAL" B, C, E, P, Q;
        C:=.25-A*A;B:=X+X;G:=1;F:=0;E:=COS(A*PI)/PI*X*"15;
        "FOR" N:=1,N+1 "WHILE" H*N<E "DO"
        "BEGIN" H:=(2*(N+X)*G-(N-1+C/N)*F)/(N+1);F:=G;
            G:=H
        "END";
        P:= Q:= F / G; E:= B - 2;
        "FOR" N:=N,N-1 "WHILE" N>0 "DO"
        "BEGIN" P:=(N-1+C/N)/(E+(N+1)*(2-P));Q:=P*(1+Q) "END";
        F:=SQRT(PI/B)/(1+Q);G:=F*(A+X+.5-P)/X
      "END";
      "IF" REC "THEN"
      "BEGIN" X:= 2 / X;
        "FOR" N:= 1 "STEP" 1 "UNTIL" NA "DO"
        "BEGIN" H:= F + (A + N) * X * G; F:= G; G:= H "END"
      "END";
      "IF" REV "THEN" "BEGIN" KA1:= F; KA:= G "END" "ELSE"
      "BEGIN" KA:= F; KA1:= G "END"
    "END" NONEXP BESS KA01;
          "EOP"

"CODE" 35195;
  "PROCEDURE" NONEXP BESS KAPLUSN(A, X, NMAX, KAN);
  "VALUE" A, X, NMAX;
  "REAL" A, X; "INTEGER" NMAX; "ARRAY" KAN;
  "BEGIN" "INTEGER" N; "REAL" K1;
    "PROCEDURE" NONEXP BESS KA(A, X, KA, KA1); "VALUE" A, X;
    "REAL" A, X, KA, KA1; "CODE" 35194;
    NONEXP BESS KA(A, X, KAN[0], K1); A:= A-1; X:= 2/X;
    "IF" NMAX > 0 "THEN" KAN[1]:= K1;
    "FOR" N:= 2 "STEP" 1 "UNTIL" NMAX "DO"
    KAN[N]:= KAN[N-2] + (A+N)*X*KAN[N-1];
  "END" NONEXP BESS KAPLUSN;
        "EOP"
```

AUTHOR: M. BAKKER.

INSTITUTE: MATHEMATICAL CENTRE.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE PROCEDURES

SPHER BESS J:
THIS PROCEDURE CALCULATES THE SPHERICAL BESSEL FUNCTIONS
J[K+.5](X)*SQRT(PI/(2*X)),K=0, ..., N, WHERE J[K+.5](X)
DENOTES THE BESSEL FUNCTION OF THE FIRST KIND OF ORDER K+.5;
X>= 0;

SPHER BESS Y:
THIS PROCEDURE CALCULATES THE SPHERICAL BESSEL FUNCTIONS
Y[K+.5](X)*SQRT(PI/(2*X)), K=0, ..., N, WHERE Y[K+.5](X)
DENOTES THE BESSEL FUNCTION OF THE THIRD KIND OF ORDER K+.5;
X SHOULD BE POSITIVE;

SPHER BESS I:
THIS PROCEDURE CALCULATES THE MODIFIED SPHERICAL BESSEL
FUNCTIONS I[K+.5](X)*SQRT(PI/(2*X))), K=0, ..., N,
WHERE I[K+.5](X) DENOTES THE MODIFIED BESSEL FUNCTION OF THE
FIRST KIND OF ORDER K+.5; X>=0;

NONEXP SPHER BESS I:
THIS PROCEDURE CALCULATES THE MODIFIED SPHERICAL BESSEL
FUNCTIONS MULTIPIED BY EXP(-X)
EXP(-X)*I[K+.5](X)*SQRT(PI/(2*X)), K=0, ...,N,
WHERE I[K+.5](X) DENOTES THE MODIFIED BESSEL FUNCTION
OF THE FIRST KIND OF ORDER K+.5; X>= 0;

SPHER BESS K:
THIS PROCEDURE CALCULATES THE MODIFIED SPHERICAL BESSEL
FUNCTIONS
K[I+.5](X)*SQRT(PI/(2*X)), I=0, ...,N,
WHERE K[I+.5](X) DENOTES THE MODIFIED BESSEL FUNCTION
OF THE THIRD KIND OF ORDER I+.5; X>0;

NONEXP SPHER BESS K:
THIS PROCEDURE CALCULATES THE MODIFIED SPHERICAL BESSEL
FUNCTIONS MULTIPLIED BY EXP(+X)
EXP(+X)*K[I+.5](X)*SQRT(PI/(2*X)), I=0, ..., N,
WHERE K[I+.5](X) DENOTES THE MODIFIED BESSEL
FUNCTION OF THE THIRD KIND OF ORDER I+.5; X>0;

KEYWORDS:

BESSEL FUNCTIONS,
SPHERICAL BESSEL FUNCTIONS,
MODIFIED SPHERICAL BESSEL FUNCTIONS.

REFERENCES:

[1]. ABRAMOWITZ, M., AND STEGUN, I. (EDS),
HANDBOOK OF MATHEMATICAL FUNCTIONS WITH FORMULAS, GRAPHS AND MATHEMATICAL TABLES.
APPL. MATH. SER. 55, U.S. GOVT. PRINTING OFFICE, WASHINGTON, D.C. , 1974.

[2]. GAUTSCHI, W., COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS.
SIAM REVIEW, VOLUME 9(1967), NUMBER 1, P.24 FF.

SUBSECTION: SPHER BESS J.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" SPHER BESS J (X, N, J); "VALUE" X, N;
"REAL" X; "INTEGER" N; "ARRAY" J;
"CODE" 35150;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: < ARITHMETIC EXPRESSION >;
THE VALUE OF THE ARGUMENT; X > = 0;

N: < ARITHMETIC EXPRESSION >;
THE UPPER BOUND OF THE INDICES OF THE ARRAY J; N > = 0;

J: < ARRAY IDENTIFIER >;
"ARRAY" J[O:N];
EXIT: J[K] HAS THE VALUE OF THE SPHERICAL BESSEL FUNCTION
J[K+.5](X) * SQRT(PI/(2*X)), 0< = K < = N;

PROCEDURES USED: START = CP 35185.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

AT FIRST THE RATIO OF TWO CONSEQUENT ARRAY ELEMENTS IS COMPUTED BY MEANS OF A BACKWARD RECURRENCE FORMULA USING MILLER 'S METHOD (SEE[2, P.46-52]) AND HENCE ALL THE ARRAY ELEMENTS ARE COMPUTED SINCE THE ZEROTH ELEMENT IS KNOWN TO BE SIN(X)/X. THE STARTING VALUE IS COMPUTED BY START.

RUNNING TIME:

ROUGHLY PROPERTIONAL TO THE MAXIMUM OF X AND N.

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X  ; "ARRAY" J[0:2]; "INTEGER" N;
    "PROCEDURE" SPHER BESS J (X, N, J);  "CODE" 35150;
    X:= 1.5; N:= 2; SPHER BESS J(X, N, J);
    OUTPUT(61, "("/, "("X=")" D.D, B"("N=")"D,
    3(3B-.14D"-ZD)")", X, N, J[0], J[1], J[2])
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
X=1.5  N=2
.66499665773603"0  .39617297071 22"0  .12734928368841"0
```

SUBSECTION: SPHER BESS Y.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" SPHER BESS Y(X, N, Y); "VALUE" X, N;
"REAL" X; "INTEGER" N; "ARRAY" Y;
"CODE" 35151;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
X:  < ARITHMETIC EXPRESSION >;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;
N:  < ARITHMETIC EXPRESSION >;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY Y; N > = 0;
Y:  < ARRAY IDENTIFIER >;
    "ARRAY" Y[0:N];
    EXIT:  Y[K] HAS THE VALUE OF THE K-TH SPHERICAL
           BESSEL FUNCTION OF THE SECOND KIND;
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

Y[0] AND Y[1] ARE GIVEN IN [1, P.438, FORMULA 10.1.12]
AND Y[2], ..., Y[N] ARE COMPUTED BY USING THE
RECURRENCE FORMULA

```
Y[K]:= ((2*K-1)/X) * Y[K-1] - Y[K-2], K > = 2.
```

EXAMPLE OF USE:

THE PROGRAM

```
"BEGIN" "REAL" X; "INTEGER" N; "ARRAY" Y[0:2];
    "PROCEDURE" SPHER BESS Y(X, N, Y); "CODE" 35151;
    X:= 1.5707 96326 79489; "COMMENT" X= PI/2; N:= 2;
    SPHER BESS Y(X, N, Y);
    OUTPUT(61, "("5(4B-.10D"-ZD)")", X, N, Y)
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
.15707963271"1  .2000000000"1
-.6223649549"-13  -.6366197724"0  -.1215854200"0
```

SUBSECTION: SPHER BESS I.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" SPHER BESS I(X, N, I); "VALUE" X, N;
"REAL" X; "INTEGER" N; "ARRAY" I;
"CODE" 35152;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: < ARITHMETIC EXPRESSION >;
THE ARGUMENT OF THE BESSEL FUNCTIONS; X > = 0;

N: < ARITHMETIC EXPRESSION >;
THE UPPER BOUND OF THE INDICES OF THE ARRAY I; N > = 0;

I: < ARRAY IDENTIFIER >;
"ARRAY" I[0:N];
EXIT: I[K] HAS THE VALUE OF THE MODIFIED SPHERICAL BESSEL FUNCTION AS DESCRIBED IN [1, CH.10.2].

METHOD AND PERFORMANCE:
AT FIRST THE NONEXPONENTIAL MODIFIED SPHERICAL BESSEL FUNCTIONS ARE COMPUTED BY USING THE PROCEDURE NONEXP SPHER BESS I; AFTERWARDS THEY ARE MULTIPLIED BY EXP(X).

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

PROCEDURES USED: NONEXP SPHER BESS I = CP 35154.

EXAMPLE OF USE:

THE PROGRAM SHOWS THAT THE RESULTS OF SPHER BESS I AND NONEXP SPHER BESS I DIFFER ONLY BY A FACTOR EXP(X):

```
"BEGIN" "REAL" X, EXPX; "INTEGER" N; "ARRAY" I1, I2[0:3];
    "PROCEDURE" SPHER BESS I(X, N, I); "CODE" 35152;
    "PROCEDURE" NE SPHER BESS I(X, N, NEI); "CODE" 35154;
    X:=1; EXPX:= EXP(X); N:= 3; SPHER BESS I(X, N,I1);
    NESPHER BESS I(X, N, I2);"FOR" N:=0, 1, 2, 3 "DO"
    OUTPUT(61, "("/ZD, 2(5B-.14D"-ZD)")", N, I1[N], I2[N]*EXPX)
"END"
```

RESULTS:

```
0  .11752011936438" 1    .11752011936438" 1
1  .36787944117144" 0    .36787944117144" 0
2  .71562870129474"-1    .71562870129474"-1
3  .10065090524070"-1    .10065090524070"-1
```

SUBSECTION: NONEXP SPHER BESS I.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NONEXP SPHER BESS I(X, N, I);
"VALUE" X, N; "REAL" X; "INTEGER" N; "ARRAY" I;
"CODE" 35154;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
X:  <ARITHMETIC EXPRESSION>;
    THE ARGUMENT OF THE BESSEL FUNCTIONS; X >= 0;
N:  <ARITHMETIC EXPRESSION>;
    THE UPPER BOUND OF THE INDICES OF THE ARRAY I; N >= 0;
I:  <ARRAY IDENTIFIER>;
    "ARRAY" I[0:N];
    EXIT: I[K] HAS THE VALUE OF THE FUNCTION
          I[K+.5](X)*EXP(-X)*SQRT(PI/(2*X)), K=0, ..., N, N >=0.
```

PROCEDURES USED: SINH = CP 35111,
START = CP 35185.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:
THE RATIO OF TWO SUBSEQUENT ELEMENTS IS COMPUTED USING A BACKWARD RECURRENCE FORMULA ACCORDING MILLER'S METHOD (SEE[2]); SINCE THE ZEROETH ELEMENT IS KNOWN TO BE (1-EXP(-2*X))/(2*X), THE OTHER ELEMENTS FOLLOW IMMEDIATELY.THE STARTING VALUE IS COMPUTED BY START.

EXAMPLE OF USE: SEE SPHER BESS I.

SUBSECTION: SPHER BESS K.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" SPHER BESS K(X, N, K); "VALUE" X, N;
"REAL" X; "INTEGER" N; "ARRAY" K;
"CODE" 35153;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: < ARITHMETIC EXPRESSION >;
THE ARGUMENT VALUE; X > 0;
N: < ARITHMETIC EXPRESSION >;
THE UPPER BOUND OF THE INDICES OF THE ARRAY K; N > = 0;
K: < ARRAY IDENTIFIER >;
"ARRAY" K[0:N];
EXIT: K[J] HAS THE VALUE OF THE J-TH MODIFIED
SPHERICAL BESSEL FUNCTION OF THE THIRD KIND,
0 < = J < = N.

PROCEDURES USED: NON EXP SPHER BESS K = CP 35155.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

AT FIRST THE NONEXPONENTIAL MODIFIED SPHERICAL BESSEL FUNCTIONS OF THE THIRD KIND ARE COMPUTED BY THE PROCEDURE NONEXP SPHER BESS K; AFTERWARDS THEY ARE MULTIPLIED BY EXP(-X).

EXAMPLE OF USE:

THE FOLLOWING PROGRAM SHOWS THAT THE RESULTS OF THE PROCEDURES SPHER BESS K EN NONEXP SPHER BESS K DIFFER ONLY BY A FACTOR EXP(X);

```
"BEGIN" "REAL" X, EXPX; "INTEGER" N; "ARRAY" K1, K2[0:3];
    "PROCEDURE" SPHER BESS K (X, N, K); "CODE" 35153;
    "PROCEDURE" NESPHER BESS K (X, N, K); "CODE" 35155;
    X:= 2; EXPX:= EXP(-X); N:= 3; SPHER BESS K (X, N, K1);
    NESPHER BESS K (X, N, K2); "FOR" K:= 0, 1, 2, 3 "DO"
    OUTPUT(61, "("/D, 2(5B-.14D"-ZD)")", N, K1[N], K2[N]*EXPX)
"END"
```

RESULTS:

```
0    .10629208289691"0    .10629208289691"0
1    .15943812434536"0    .15943812434536"0
2    .34544926941495"0    .34544926941494"0
3    .10230612978828"1    .10230612978828"1
```

SUBSECTION: NONEXP SPHER BESS K.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NON EXP SPHER BESS K(X, N, K);
"VALUE" X, N; "REAL" X; "INTEGER" N; "ARRAY" K;
"CODE" 35155;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTIONS; X > 0;

N: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE INDICES OF THE ARRAY K; N >= 0;

K: <ARRAY IDENTIFIER>;
"ARRAY" K[0:N];
EXIT: K[J] HAS THE VALUE OF THE FUNCTION
K[J+.5](X)*EXP(X)*SQRT(PI/(2*X)), J=0,...,N.

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY : NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

THE FUNCTIONS ARE COMPUTED BY USING THE (NUMERICALLY STABLE) RECURRENCE FORMULA : K[J]=((2*J-1)/X)*K[J-1]+K[J-2], J >=2,
K[0]=PI/(2*X), K[1]=K[0]*(1+1/X) .

EXAMPLE OF USE: SEE SPHER BESS K.

SOURCE TEXT(S):

```
"CODE" 35150;
  "COMMENT" SPHERICAL BESSEL FUNCTIONS J[.5](X),  , J[N+.5](X);
  "PROCEDURE" SPHER BESS J(X, N, J); "VALUE" X, N;
  "REAL" X; "INTEGER" N; "ARRAY" J;
  "IF" X = 0 "THEN"
  "BEGIN" J[0]:= 1;
    "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" J[N]:=0
  "END" "ELSE" "IF" N = 0 "THEN"
  "BEGIN" "REAL" X2;
    "IF" ABS(X) < .015 "THEN"
    "BEGIN" X2:= X * X / 6; J[0]:= 1 + X2 * (X2 * .3 - 1) "END" "ELSE"
    J[0]:= SIN(X)/X
  "END" "ELSE"
  "BEGIN" "INTEGER" M; "REAL" R, S;
    "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
    R:= 0; M:= START(X,N,0);
    "FOR" M:= M "STEP" - 1 "UNTIL" 1 "DO"
    "BEGIN" R:= 1 / ((M + M + 1) / X - R); "IF" M <= N "THEN" J[M]:= R
    "END"; "IF" X < .015  "THEN"
    "BEGIN" S:= X * X / 6;
      J[0]:= R:= S * (S * .3 - 1) + 1 "END" "ELSE"
    J[0]:= R:= SIN(X) / X;
    "FOR" M:= 1 "STEP" 1 "UNTIL" N "DO" J[M]:= R:= J[M] * R;
  "END" SPHER BESS J;
        "EOP"

"CODE" 35151;
  "COMMENT" SPHERICAL BESSEL FUNCTIONS Y[.5](X),   , Y[N+.5](X);
  "PROCEDURE" SPHER BESS Y(X, N, Y); "VALUE" X, N;
  "INTEGER" N; "REAL" X; "ARRAY" Y;
  "IF" N=0 "THEN"  Y[0]:= - COS(X)/X "ELSE"
  "BEGIN" "REAL" YI, YI1, YI2; "INTEGER" I;
    YI2:= Y[0]:= -COS(X)/X; YI1:= Y[1]:= (YI2 - SIN(X))/X;
    "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" Y[I]:= YI:= -YI2 + (I+I-1) * YI1/X;
      YI2:= YI1; YI1:= YI
    "END"
  "END";
        "EOP"
```

```
"CODE" 35152;
  "COMMENT" SPHERICAL BESSEL FUNCTIONS I[.5](X),    , I[N+.5](X);
  "PROCEDURE" SPHER BESS I(X, N, I); "VALUE" X, N;
  "REAL" X; "INTEGER" N; "ARRAY" I;
  "IF" X= 0 "THEN"
  "BEGIN" I[0]:=1;
    "FOR" N:= N "STEP" -1 "UNTIL"  1 "DO" I[N]:= 0
  "END" "ELSE"
  "BEGIN" "REAL" EXPX;
    "PROCEDURE" NONEXP SPHER BESS I(X, N, I); "CODE" 35154;
    EXPX:= EXP(X);
    NONEXP SPHER BESS I(X, N, I);
    "FOR" N:= N "STEP" - 1 "UNTIL" 0 "DO" I [N]:= I [N] * EXPX
  "END" SPHER BESS I;
        "EOP"

"CODE" 35153;
  "COMMENT" MODIFIED SPHERICAL BESSEL FUNCTIONS
      K[.5](X),    , K[N+.5](X);
  "PROCEDURE" SPHER BESS K(X, N, K); "VALUE" X, N;
  "INTEGER" N; "REAL" X; "ARRAY" K;
  "BEGIN" "REAL" EXPX;
    "PROCEDURE" NONEXP SPHER BESS K(X, N, K); "CODE" 35155;
    EXPX:= EXP(-X);
    NONEXP SPHER BESS K(X, N, K);
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" K[N]:= K[N] * EXPX
  "END";
        "EOP"
```

```
"CODE" 35154;
  "PROCEDURE" NONEXP SPHER BESS I(X, N, I); "VALUE" X, N;
  "REAL" X; "INTEGER" N; "ARRAY" I;
  "IF" X= 0 "THEN"
  "BEGIN" I[0]:=1;
    "FOR" N:= N "STEP" -1 "UNTIL" 1 "DO" I[N]:= 0
  "END" "ELSE"
  "BEGIN" "REAL" X2, R, S; "INTEGER" M;
    "REAL" "PROCEDURE" SINH(X); "CODE" 35111;
    "INTEGER" "PROCEDURE" START(X,N,T); "CODE" 35185;
    X2:= X+X;
    I[0]:= X2:= "IF" X = 0 "THEN" 1 "ELSE" "IF" X2 < 0.7 "THEN"
         SINH(X) / (X * EXP(X)) "ELSE" (1-EXP(-X2))/X2;
    "IF" N= 0 "THEN" "GO TO" EXIT;
    R:= 0; M:= START(X,N,1);
    "FOR" M:= M "STEP" -1 "UNTIL" 1 "DO"
    "BEGIN" R:= 1/((M+M+1)/X+R);
      "IF" M  <=  N "THEN" I[M]:= R
    "END";
    "FOR" M:= 1 "STEP" 1 "UNTIL" N "DO"
    I[M]:= X2:= X2 * I[M];       EXIT:
  "END";
        "EOP"

"CODE" 35155;
  "PROCEDURE" NONEXP SPHER BESS K(X, N, K); "VALUE" X, N;
  "REAL" X; "INTEGER" N; "ARRAY" K;
  "BEGIN" "INTEGER" I; "REAL" KI, KI1, KI2;
    X:= 1/X; K[0]:= KI2:= X*1.57079632679 49;
    "IF" N=0 "THEN" "GO TO" EXIT;
    K[1]:= KI1:= KI2 * (1+X);
    "FOR" I:= 2 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" K[I]:= KI:= KI2 + (I+I-1) * X * KI1;
      KI2:= KI1; KI1:= KI "END";
    EXIT:
  "END";
        "EOP"
```

AUTHOR : P.W.HEMKER.

CONTRIBUTOR : F.GROEN.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED : 740620.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS TWO PROCEDURES FOR THE EVALUATION OF AIRY FUNCTIONS AND COMPUTING THEIR ZEROS. FOR THE DEFINITION OF THESE FUNCTIONS SEE REF[1].

AIRY EVALUATES THE AIRY FUNCTIONS AI(Z) AND BI(Z) AND THEIR DERIVATIVES.

AIRYZEROS COMPUTES THE ZEROS AND ASSOCIATED VALUES OF THE AIRY FUNCTIONS AI(Z) AND BI(Z) AND THEIR DERIVATIVES.

KEYWORDS :

AIRY FUNCTION,
DERIVATIVE AIRY FUNCTION,
ZERO OF AIRY FUNCTION.

SUBSECTION : AIRY.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" AIRY(X,AI,AID,BI,BID,EXPON,FIRST);
"VALUE" X,FIRST; "BOOLEAN" FIRST;
"REAL" X,AI,AID,BI,BID,EXPON;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

X: <ARITHMETIC EXPRESSION>;
ENTRY : THE REAL ARGUMENT OF THE AIRY FUNCTIONS.

AI: <VARIABLE>;
EXIT : THE VALUE OF THE AIRY FUNCTION AI IS GIVEN BY : EXP(-EXPON) * AI.
NOTE : IF X < 9 THEN EXPON = 0.

AID: <VARIABLE>;
EXIT : THE VALUE OF THE DERIVATIVE OF THE AIRY FUNCTION AI IS GIVEN BY : EXP(-EXPON) * AID.
NOTE : IF X < 9 THEN EXPON = 0.

BI: <VARIABLE>;
EXIT : THE VALUE OF THE AIRY FUNCTION BI IS GIVEN BY : EXP(EXPON) * BI.
NOTE : IF X < 9 THEN EXPON = 0.

BID: <VARIABLE>;
EXIT : THE VALUE OF THE DERIVATIVE OF THE AIRY FUNCTION BI IS GIVEN BY : EXP(EXPON) * BID.
NOTE : IF X < 9 THEN EXPON = 0.

EXPON: <VARIABLE>;
EXIT : IF X < 9 THEN 0 ELSE 2/3 * X ** (3/2).

FIRST: <BOOLEAN EXPRESSION>;
FIRST SHOULD BE "FALSE" UNLESS THE PROCEDURE IS CALLED FOR THE FIRST TIME. IF FIRST IS "TRUE" THEN TWO OWN ARRAYS OF COEFFICIENTS ARE BUILT UP.

PROCEDURES USED : NONE.

REQUIRED CENTRAL MEMORY : TWO OWN ARRAYS OF ORDER 10 ARE DECLARED.

RUNNING TIME : IF 2.5 <= X <= 8 THEN ABOUT 8"-3 SEC., ELSE BETWEEN 3"-3 AND 4"-3 SEC. ON THE CYBER 73/28.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

SEE REF[2] OF THE SUBSECTION AIRYZEROS (THIS SECTION).

REFERENCES :

SEE REFERENCES OF THE SUBSECTION AIRYZEROS (THIS SECTION).

EXAMPLE OF USE :

```
"BEGIN" "REAL" A,B,C,D,E;
    "PROCEDURE" AIRY(X,AI,AID,BI,BID,EXPON,FIRST);"CODE" 35140;
    AIRY(9.654894,A,B,C,D,E,"TRUE");
    OUTPUT(61,"("/,"("AI (9.654894) = ")",N")",A*EXP(-E));
    OUTPUT(61,"("/,"("AID(9.654894) = ")",N")",B*EXP(-E));
    OUTPUT(61,"("/,"("BI (9.654894) = ")",N")",C*EXP( E));
    OUTPUT(61,"("/,"("BID(9.654894) = ")",N")",D*EXP( E));
"END"
```

RESULTS :

```
AI (9.654894) = +3.2873525549165"-010
AID(9.654894) = -1.0297999323482"-009
BI (9.654894) = +1.5583887049670"+008
BID(9.654894) = +4.8010374682654"+008
```

SUBSECTION : AIRYZEROS.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"REAL" "PROCEDURE" AIRYZEROS(N,D,ZAI,VAI);
"VALUE" N,D; "INTEGER" N,D; "ARRAY" ZAI,VAI;
```

AIRYZEROS := THE N-TH ZERO OF THE SELECTED AIRY-FUNCTION.

```
THE MEANING OF THE FORMAL PARAMETERS IS :
N :         <ARITHMETIC EXPRESSION>;
            ENTRY : THE NUMBER OF ZEROS TO BE CALCULATED;
D :         <ARITHMETIC EXPRESSION>;
            ENTRY : AN  INTEGER  WHICH  SELECTS THE  REQUIRED AIRY
                    FUNCTION. D = 0, 1, 2 OR 3.
ZAI :       <ARRAY IDENTIFIER>;
            "ARRAY" ZAI[1 : N];
            EXIT : ZAI[J] CONTAINS THE J-TH ZERO OF THE SELECTED
                   AIRY-FUNCTION :
                        IF D = 0 THEN AI(Z),
                        IF D = 1 THEN (D/DX) AI(X),
                        IF D = 2 THEN BI(X),
                        IF D = 3 THEN (D/DX) BI(X);
VAI :       <ARRAY IDENTIFIER>;
            "ARRAY" VAI[1 : N];
            EXIT: VAI[J] CONTAINS THE VALUE AT X = ZAI[J] OF THE
                  FOLLOWING FUNCTION :
                       IF D = 0 THEN (D/DX) AI(X),
                       IF D = 1 THEN AI(X),
                       IF D = 2 THEN (D/DX) BI(X),
                       IF D = 3 THEN BI(X);
```

PROCEDURES USED :

AIRY = CP35140;

REQUIRED CENTRAL MEMORY : NO AUXILIARY ARRAYS ARE DECLARED.

RUNNING TIME : DEPENDENT ON THE VALUES OF N AND D. IN MOST CASES THE RUNNING TIME IS LESS THAN N * 0.01 SEC. ON THE CYBER 73/28.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

A FIRST APPROXIMATION OF THE ZEROS OF THE SELECTED AIRY-FUNCTION IS CALCULATED BY MEANS OF THE ASYMPTOTIC EXPANSION (SEE THE FORMULAS 10.4.94 - 10.4.105 OF REF[1]); THIS VALUE IS CORRECTED BY THE (REPEATED) USE OF A QUADRATIC INTERPOLATION RULE.
THE COMPUTED ZEROS WILL SATISFY AT LEAST ONE OF THE FOLLOWING CONDITIONS :
1: THE ABSOLUTE VALUE OF THE SELECTED AIRY-FUNCTION AT A COMPUTED ZERO IS LESS THAN "-12. NOTE: THE VALUES OF THE AIRY-FUNCTIONS ARE CALCULATED BY MEANS OF THE PROCEDURE AIRY (THIS SECTION).
2: THE RELATIVE PRECISION OF THE COMPUTED ZERO IS "-14.
THE ASSOCIATED VALUES (DELIVERED IN THE ARRAY VAI) ARE ALSO CALCULATED BY MEANS OF THE PROCEDURE AIRY (THIS SECTION).

REFERENCES :

[1] : M.ABRAMOWITZ AND I.A.STEGUN,
HANDBOOK OF MATHMATICAL FUNCTIONS,
DOVER PUBLICATIONS, INC. NEW YORK, 1965.

[2] : R.G.GORDON,
EVALUATION OF AIRY FUNCTIONS,
THE JOURNAL OF CHEMICAL PHYSICS, VOLUME 51, 1969, PP. 23-24.

EXAMPLE OF USE :

```
"BEGIN" "ARRAY" ZBI,VBID[1 : 3];
"REAL" "PROCEDURE" AIRYZEROS(N,D,ZAI,VAI); "CODE"35145;
OUTPUT(61,"("/"("THE THIRD ZERO OF BI(X) IS")"/,N,
             /"("THE VALUE OF (D/DX)BI(X) IN THIS POINT IS")"/,N")"
             ,AIRYZEROS(3,2,ZBI,VBID),VBID[3])
"END"
```

RESULTS :

```
THE THIRD ZERO OF BI(X) IS
-4.8307378416626"+000
THE VALUE OF (D/DX)BI(X) IN THIS POINT IS
+8.3699101261986"-001
```

SOURCE TEXT(S):

```
"CODE" 35140;
 "PROCEDURE" AIRY(Z,AI,AID,BI,BID,EXPON,FIRST);
 "VALUE" Z,FIRST; "BOOLEAN" FIRST;
 "REAL" Z,AI,AID,BI,BID,EXPON;
 "BEGIN" "REAL" S,T,U,V,SC,TC,UC,VC,X,K1,K2,K3,K4,
      C,ZT,SI,CO,EXPZT,SQRTZ,WWL,PL,PL1,PL2,PL3;
      "OWN" "REAL" C1,C2,SQRT3,SQRT1OPI,PIO4;
      "OWN" "REAL" "ARRAY" XX,WW[1:10];
      "INTEGER" N,L;

      "IF" FIRST "THEN"
      "BEGIN" SQRT3:=     1.73205080756887729;
              SQRT1OPI:= 0.56418958354775629;
              PIO4:=      0.78539816339744831;
              C1:=        0.35502 80538 87817;
              C2:=        0.25881 94037 92807;
              XX[ 1]:=    1.40830 81072 180964 "+1;
              XX[ 2]:=    1.02148 85479 197331 "+1;
              XX[ 3]:=    7.44160 18450 450930     ;
              XX[ 4]:=    5.30709 43061 781927     ;
              XX[ 5]:=    3.63401 35029 132462     ;
              XX[ 6]:=    2.33106 52303 052450     ;
              XX[ 7]:=    1.34479 70824 609268     ;
              XX[ 8]:=    6.41888 58369 567296 "-1;
              XX[ 9]:=    2.01003 45998 121046 "-1;
              XX[10]:=    8.05943 59172 052833 "-3;
              WW[ 1]:=    3.15425 15762 964787"-14;
              WW[ 2]:=    6.63942 10819 584921"-11;
              WW[ 3]:=    1.75838 89061 345669"- 8;
              WW[ 4]:=    1.37123 92370 435815"- 6;
              WW[ 5]:=    4.43509 66639 284350"- 5;
              WW[ 6]:=    7.15550 10917 718255"- 4;
              WW[ 7]:=    6.48895 66103 335381"- 3;
              WW[ 8]:=    3.64404 15875 773282"- 2;
              WW[ 9]:=    1.43997 92418 590999"- 1;
              WW[10]:=    8.12311 41336 261486"- 1;
      "END";

      EXPON:= 0;
      "IF" Z >= -5.0 "AND" Z <= 8 "THEN"
      "BEGIN" U:= V:= T:= UC:= VC:= TC:= 1;
          S:= SC:= 0.5; N:= 0; X:= Z*Z*Z;
          "FOR" N:= N+3 "WHILE" ABS(U)+ABS(V)+ABS(S)+ABS(T)
                                  > "-18 "DO"
          "BEGIN" U:=U*X/(N*(N-1)); V:= V*X/(N*(N+1));
                  S:=S*X/(N*(N+2)); T:= T*X/(N*(N-2));
                  UC:= UC+U; VC:= VC+V; SC:= SC+S; TC:= TC+T
          "END";
                                                                  "COMMENT"
```

```
            BI:= SQRT3 * (C1*UC + C2*Z*VC);
            BID:=SQRT3 * (C1*Z*Z*SC +C2*TC);
            "IF" Z<2.5 "THEN"
            "BEGIN" AI:= C1*UC - C2*Z*VC;
                    AID:= C1*SC*Z*Z - C2*TC;
                    "GOTO" END
            "END"
        "END";

        K1:= K2:= K3:= K4:= 0;
        SQRTZ:= SQRT(ABS(Z));
        ZT:= 0.66666 66666 66667 * ABS(Z)*SQRTZ;
        C:= SQRT1OPI/SQRT(SQRTZ);
        "IF" Z<0 "THEN"
        "BEGIN" Z:= -Z; CO:= COS(ZT-PI04); SI:= SIN(ZT-PI04);
            "FOR" L:= 1 "STEP" 1 "UNTIL" 10 "DO"
            "BEGIN" WWL:= WW[L]; PL:= XX[L]/ZT;
                PL2:=PL*PL; PL1:= 1+PL2; PL3:= PL1*PL1;
                K1:= K1 + WWL/PL1;
                K2:= K2 + WWL*PL/PL1;
                K3:= K3 + WWL*PL*(1+PL*(2/ZT+PL))/PL3;
                K4:= K4 + WWL*(-1-PL*(1+PL*(ZT-PL))/ZT)/PL3;
            "END";
            AI:= C*(CO*K1+SI*K2);
            AID:= 0.25*AI/Z - C*SQRTZ*(CO*K3+SI*K4);
            BI:= C*(CO*K2-SI*K1);
            BID:= 0.25*BI/Z - C*SQRTZ*(CO*K4-SI*K3);
        "END" "ELSE"
        "BEGIN" "IF" Z < 9 "THEN" EXPZT:= EXP(ZT) "ELSE"
            "BEGIN" EXPZT:= 1; EXPON:= ZT "END";
            "FOR" L:= 1 "STEP" 1 "UNTIL" 10 "DO"
            "BEGIN" WWL:= WW[L]; PL:= XX[L]/ZT;
                PL1:= 1+PL; PL2:= 1-PL;
                K1:= K1 + WWL/PL1;
                K2:= K2 + WWL*PL/(ZT*PL1*PL1);
                K3:= K3 + WWL/PL2;
                K4:= K4 + WWL*PL/(ZT*PL2*PL2);
            "END";
            AI:= 0.5*C*K1/EXPZT;
            AID:= AI*(-.25/Z-SQRTZ) + 0.5*C*SQRTZ*K2/EXPZT;
            "IF" Z >= 8 "THEN"
            "BEGIN" BI:= C*K3*EXPZT;
                    BID:= BI*(SQRTZ-0.25/Z) - C*K4*SQRTZ*EXPZT;
            "END";
        "END";
   END:
 "END" AIRY;
        "EOP"
```

```
"CODE" 35145;
"REAL" "PROCEDURE" AIRYZEROS(N,D,ZAI,VAI);
"VALUE" N,D; "INTEGER" N,D; "ARRAY" ZAI,VAI;
"BEGIN" "BOOLEAN" A, FOUND; "INTEGER" I;
    "REAL" C,E,R,ZAJ,ZAK,VAJ,DAJ,KAJ,ZZ;
    "PROCEDURE" AIRY(A,B,C,D,E,F,G); "CODE" 35140;

    A := D = 0 "OR" D = 2;
    R := "IF" D = 0 "OR" D = 3 "THEN" -1.1780 97245 09617
                               "ELSE" -3.5342 91735 28852;
    "COMMENT" R := "IF" D = 0 "OR" D = 3 "THEN" -3 * PI / 8
                                         "ELSE" -9 * PI / 8;
    AIRY(0,ZAJ,VAJ,DAJ,KAJ,ZZ,"TRUE");
    "FOR" I := 1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN" R := R + 4.7123 88980 38469; "COMMENT" R := R + 3 * PI / 2;
        ZZ := R * R;
        ZAJ := "IF" I = 1 "AND" D = 1 "THEN" -1.01879 297 "ELSE"
               "IF" I = 1 "AND" D = 2 "THEN" -1.17371 322 "ELSE"
               R ** 0.66666 66666 66667 * ( "IF" A "THEN"
               - ( 1 + ( 5/48 - ( 5/36 - ( 77125/82944 - (
               1080 56875 / 69 67296 - (16 23755 96875 / 3344 30208)
               /ZZ)/ZZ)/ZZ)/ZZ)/ZZ)
                                                  "ELSE"
               - ( 1 - ( 7/48 - ( 35/288 - ( 1 81223 / 2 07360 - (
               186 83371 / 12 44160 - ( 9 11458 84361 / 1911 02976 )
               /ZZ)/ZZ)/ZZ)/ZZ)/ZZ));
        "IF" D <= 1 "THEN" AIRY(ZAJ,VAJ,DAJ,C,E,ZZ,"FALSE")
                  "ELSE" AIRY(ZAJ,C,E,VAJ,DAJ,ZZ,"FALSE");
        FOUND := ABS( "IF" A "THEN" VAJ "ELSE" DAJ ) < "-12;
        "FOR" C := C "WHILE" "NOT" FOUND "DO"
        "BEGIN" "IF" A "THEN"
            "BEGIN" KAJ := VAJ / DAJ;
                ZAK := ZAJ - KAJ * (1 + ZAJ * KAJ * KAJ)
            "END" "ELSE"
            "BEGIN" KAJ := DAJ / (ZAJ * VAJ);
                ZAK := ZAJ - KAJ * (1 + KAJ * (KAJ * ZAJ + 1 / ZAJ))
            "END";
            "IF" D <= 1 "THEN" AIRY(ZAK,VAJ,DAJ,C,E,ZZ,"FALSE")
                      "ELSE" AIRY(ZAK,C,E,VAJ,DAJ,ZZ,"FALSE");
            FOUND := ABS(ZAK - ZAJ) < "-14 * ABS(ZAK) "OR"
                     ABS("IF" A "THEN" VAJ "ELSE" DAJ) < "-12;
            ZAJ := ZAK
        "END";
        VAI[I] := "IF" A "THEN" DAJ "ELSE" VAJ;
        ZAI[I] := ZAJ;
    "END";
    AIRYZEROS := ZAI[N];
"END" AIRYZEROS;
        "EOP"
```

*********** M1067DA //// END OF LIST ////
*********** M1067DA //// END OF LIST ////

AUTHOR: C.G. VAN DER LAAN

CONTRIBUTORS: C.G. VAN DER LAAN, M. VOORINTHOLT

INSTITUTE: REKENCENTRUM RIJKSUNIVERSITEIT GRONINGEN

RECEIVED: 780601

BRIEF DESCRIPTION:

NEWTON CALCULATES THE COEFFICIENTS OF THE NEWTON POLYNOMIAL THROUGH GIVEN INTERPOLATION POINTS AND CORRESPONDING FUNCTION VALUES.

KEYWORDS:

NEWTON INTERPOLATION,
POLYNOMIAL COEFFICIENTS,
DIVIDED DIFFERENCES.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" NEWTON(N,X,F);
"VALUE"N;"INTEGER"N;"ARRAY"X,F;
"CODE" 36010;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:  <ARITHMETIC EXPRESSION>;
    THE DEGREE OF THE POLYNOMIAL;
X:  <ARRAY IDENTIFIER>;
    "ARRAY"X[0:N];
    ENTRY:  THE INTERPOLATION POINTS;
F:  <ARRAY IDENTIFIER>;
    "ARRAY"F[0:N];
    ENTRY:  THE FUNCTION VALUES AT THE INTERPOLATION POINTS;
    EXIT:   THE COEFFICIENTS OF THE NEWTON POLYNOMIAL.
```

PROCEDURES USED: NONE.

RUNNING TIME: THE NUMBER OF DIVISIONS IS N(N+1)/2.

METHOD AND PERFORMANCE:

THE POLYNOMIAL OF DEGREE N IN X IS REPRESENTED AS

```
         N              K-1
        SUM (A[K] * PROD (X-X[L])).
        K=0             L=0
```

THE COEFFICIENTS OF THE (NEWTON) POLYNOMIAL, A[0:N], ARE CALCULATED BY INTERPOLATION AT THE GIVEN ARGUMENTS, X[0:N], AND FUNCTION VALUES, F[0:N]; THE RESULTING SET OF EQUATIONS IS SOLVED BY TRANSFORMING THE CORRESPONDING LOWER TRIANGULAR MATRIX TO DIAGONAL FORM.

EXAMPLE OF USE:

```
"BEGIN" "ARRAY" X,F[0:2];
    "PROCEDURE"NEWTON(N,X,F);
    "VALUE" N; "INTEGER" N; "ARRAY" X,F;
    "CODE"36010;
    X[0]:=0;X[1]:=.5;X[2]:=1;
    F[0]:=1;F[1]:=F[2]:=0;
    NEWTON(2,X,F);
    OUTPUT(61,"("/,"("THE NEWTON COEFF. ARE")",
        /,3(N)")",F[0],F[1],F[2]);
"END"TSTNEWTON;
```

```
THE NEWTON COEFF. ARE
+1.0000000000000"+000  -2.0000000000000"+000  +2.0000000000000"+000
```

SOURCE TEXT(S):

```
"CODE"36010;
"PROCEDURE" NEWTON(N,X,F);
"VALUE" N; "INTEGER" N; "ARRAY" X,F;
"COMMENT" NEWTON DETERMINES THE COEFFICIENTS C[J],J=0,...N,
OF THE INTERPOLATION POLYNOMIAL C[0] + C[1] *(X-X[0])+...+
C[N] * (X-X[0])*...*(X-X[N-1]) OUT OF N+1 LIN. EQUAT.
THE ARGUMENTS AND FUNCTION VALUES MUST BE GIVEN IN
ARRAY X, F[0:N]. THE ARRAY F IS OVERWRITTEN BY
THE COEFFICIENTS C[J], J=0,...N;
"BEGIN" "INTEGER" K,I,IM1;
"REAL" XIM1,FIM1;
IM1:=0;
"FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
"BEGIN" FIM1:=F[IM1];XIM1:=X[IM1];
    "FOR" K:= I "STEP" 1 "UNTIL" N "DO" F[K]:= (F[K]-FIM1)/(X[K]-XIM1);
    IM1:= I
"END"
"END" NEWTON;
        "EOP"
```

AUTHOR: C.G. VAN DER LAAN

CONTRIBUTORS: C.G. VAN DER LAAN, M. VOORINTHOLT

INSTITUTE: REKENCENTRUM RIJKSUNIVERSITEIT GRONINGEN

RECEIVED: 780601

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THREE PROCEDURES:

MINMAXPOL: CALCULATES THE COEFFICIENTS OF THE POLYNOMIAL (AS A SUM OF POWERS) WHICH APPROXIMATES A FUNCTION, GIVEN FOR DISCRETE ARGUMENTS, IN SUCH A WAY THAT THE INFINITY NORM OF THE ERROR VECTOR IS MINIMIZED.
INI: SELECTS A (SUB)SET OF INTEGERS OUT OF A GIVEN SET OF INTEGERS;
SNDREMEZ: EXCHANGES AT MOST N+1 NUMBERS WITH NUMBERS OUT OF A REFERENCE SET;
(INI AND SNDREMEZ ARE AUXILIARY PROCEDURES USED IN MINMAXPOL.)

KEYWORDS:

(SECOND) REMEZ ALGORITHM,
MINIMAX POLYNOMIAL APPROXIMATION.

REFERENCES:

MEINARDUS, G. (1964):
APPROXIMATION OF FUNCTION AND THEIR NUMERICAL TREATMENT (GERMAN).
SPRINGER TRACTS IN NATURAL PHILOSOPHY, VOL. 4.

DEKKER, T.J. (1967):
CURSUS WETENSCHAPPELIJK REKENEN A.
MATHEMATISCH CENTRUM.

SUBSECTION : MINMAXPOL.

CALLING SEQUENCE:

```
    THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
    "PROCEDURE"MINMAXPOL(N,M,Y,FY,CO,EM);
    "VALUE"N,M;"INTEGER"N,M;"ARRAY"Y,FY,CO,EM;
    "CODE" 36022;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:      <ARITHMETIC EXPRESSION>;
            THE DEGREE OF THE APPROXIMATING POLYNOMIAL (N>=0);
    M:      <ARITHMETIC EXPRESSION>;
            THE NUMBER OF REFERENCE FUNCTION VALUES VIZ. ARGUMENTS
            IS M+1;
    Y,FY:   <ARRAY IDENTIFIERS>;
            "ARRAY"Y,FY[0:M];
            ENTRY: FY[I] IS THE FUNCTION VALUE AT Y[I], FOR I=0,...M;
    CO:     <ARRAY IDENTIFIER>;
            "ARRAY"CO[0:N];
            EXIT: THE COEFFICIENTS OF THE APPROXIMATING POLYNOMIAL
                  (CO[N] IS COEFFICIENT OF Y**N);
    EM:     <ARRAY IDENTIFIER>;
            "ARRAY"EM[0:3];
            ENTRY:  EM[2]:THE MAXIMUM ALLOWED NUMBER OF
                          ITERATIONS (SAY 10*N+5);
            EXIT:    EM[0]:THE DIFFERENCE OF THE GIVEN FUNCTION AND
                          THE POLYNOMIAL IN THE FIRST APPROXIMATION
                          POINT;
                     EM[1]:THE INFINITY NORM OF THE ERROR OF
                          APPROXIMATION OVER THE DISCRETE INTERVAL;
                     EM[3]:THE NUMBER OF ITERATIONS PERFORMED.
```

```
PROCEDURES USED:  ELMVEC    = CP34020,
                  DUPVEC    = CP31030,
                  NEWTON    = CP36010,
                  POL       = CP31040,
                  NEWGRN    = CP31050,
                  INI       = CP36020,
                  SNDREMEZ  = CP36021.
```

REQUIRED CENTRAL MEMORY:

AN INTEGER ARRAY AND THREE (REAL) ARRAYS OF N+2 ELEMENTS AS WELL AS A (REAL) ARRAY OF M+1 ELEMENTS ARE INTERNALLY DECLARED.

RUNNING TIME:

THE SECOND REMEZ ALGORITHM (ON A DISCRETE SET) IS QUADRATIC CONVERGENT;IN EACH ITERATION THE NUMBER OF OPERATIONS (MULTIPLICATIONS AND ADDITIONS) IS PROPORTIONAL TO M*N.

```
S:      <ARRAY IDENTIFIER>;
        "INTEGER" "ARRAY" S[O:N];
        ENTRY: IN S ONE MUST GIVE N+1 (STRICTLY)
               MONOTONE INCREASING NUMBERS OUT OF O,...,M;
        EXIT : N+1 (STRICTLY) MONOTONE INCREASING NUMBERS OUT OF
               THE NUMBERS O,1,...,M;
G:      <ARRAY IDENTIFIER>;
        "ARRAY" G[O:M];
        ENTRY: IN ARRAY G[O:M] ONE MUST GIVE FUNCTION VALUES;
EM:     <ARRAY IDENTIFIER>;
        "ARRAY" EM[O:1];
        ENTRY: O<EM[O]<=G[I],I=O,...,M;
        EXIT : EM[1]:=INFINITY NORM OF ARRAY G[O:M].
```

PROCEDURES USED: INFNRMVEC = CP31061.

METHOD AND PERFORMANCE:

THE SECOND REMEZ ALGORITHM IS USED (MEINARDUS,G.(1964)).

EXAMPLE OF USE:

```
"BEGIN""ARRAY"EM[O:1],G[O:7];"INTEGER""ARRAY"S[O:2];
"PROCEDURE"SNDREMEZ(N,M,S,G,EM);"CODE"36021;
G[O]:=10;G[1]:=12;G[2]:=-15;G[3]:=-10;
G[4]:=-14;G[5]:=15;G[6]:=10;G[7]:=11;
EM[O]:=10;S[O]:=0;S[1]:=3;S[2]:=6;
OUTPUT(61,"(""("THE NUMBERS:")",/,"("S[J]:")",3(B-D),/,
   "("G[S[J]]:")",3(B-DD)")",
   S[O],S[1],S[2],G[S[O]],G[S[1]],G[S[2]]);
SNDREMEZ(2,7,S,G,EM);
OUTPUT(61,"("//,"("ARE EXCHANGED WITH:")",/,"("S[J]:")",3(B-D),/,
   "("G[S[J]]:")",3(B-DD),//,
   "("THE REFERENCE SET OF FUNCTIONVALUES IS:")",/,8(B-DD)")",
   S[O],S[1],S[2],G[S[O]],G[S[1]],G[S[2]] ,
   G[O],G[1],G[2],G[3],G[4],G[5],G[6],G[7])
"END"
```

```
THE NUMBERS:
S[J]:  0  3  6
G[S[J]]:  10 -10  10

ARE EXCHANGED WITH:
S[J]:  0  2  5
G[S[J]]:  10 -15  15

THE REFERENCE SET OF FUNCTIONVALUES IS:
  10  12 -15 -10 -14  15  10  11
```

SOURCE TEXT(S) :

```
"CODE"36022;
"PROCEDURE" MINMAXPOL(N,M,Y,FY,CO,EM);
"VALUE" N,M;"INTEGER" N,M;
"ARRAY" Y,FY,CO,EM;
"COMMENT" MINMAXPOL CALCULATES THE COEFFICIENTS,
CO[I],I=,.....N OF THE POLYNOMIAL
P(Y)=CO[0]+CO[1]*Y+...+CO[N]*Y**N,
THAT APPROXIMATES THE DISCRETE FUNCTION FY[I],I=0,...M,
GIVEN FOR THE ARGUMENTS Y[I],I=0,...M,
IN THE MINIMAX NORM.
THE ARGUMENTS MUST BE GIVEN IN MONOTONE INCREASING ORDER.
IN ARRAY EM[0:3], ONE MUST GIVE THE MAXIMUM ALLOWED NUMBER OF
ITERATIONS,EM[2].
MOREOVER,
EM[0]:=THE DIFFERENCE OF THE GIVEN FUNCTION AND THE POLYNOMIAL
IN THE FIRST APPROXIMATION POINT,
EM[1]:=THE MAXIMUM OF ! P(Y[I])-FY[I]! FOR I=0,...M,
EM[3]:=THE NUMBER OF ITERATIONS PERFORMED.
THE PROCEDURES ELMVEC,DUPVEC,POL,NEWTON,NEWGRN,
INI,SNDREMEZ
ARE USED.
REFERENCE:MEINARDUS,G.(1964,CH.7),
APPROXIMATION VON FUNKTIONEN UND IHRE NUMERISCHE BEHANDLUNG;
"BEGIN" "INTEGER" NP1,K,POMK,COUNT,CNT,J,MI;
"REAL" E,ABSE,ABSEH;
NP1:=N+1;
"BEGIN"
"INTEGER" "ARRAY" S[0:NP1];
"ARRAY" X,B,COEF[0:NP1]
,G[0:M];

"PROCEDURE" ELMVEC(L,U,SHIFT,A,B,X);
"VALUE" L,U,SHIFT,X; "INTEGER" L,U,SHIFT; "REAL" X; "ARRAY" A,B;
"CODE" 34020;
"PROCEDURE" DUPVEC(L,U,SHIFT,A,B);
"VALUE" L,U,SHIFT; "INTEGER" L,U,SHIFT; "ARRAY" A,B;
"CODE" 31030;
"REAL" "PROCEDURE" POL(N,X,A);
"VALUE"N,X; "INTEGER" N; "REAL" X; "ARRAY" A;
"CODE" 31040;
"PROCEDURE" NEWTON(N,X,F);
"VALUE" N; "INTEGER" N; "ARRAY" X,F;
"CODE" 36010;
"PROCEDURE" NEWGRN(N,X,C);
"VALUE" N; "INTEGER" N; "ARRAY" X,C;
"CODE" 31050;
"PROCEDURE" INI(N,M,S);
"VALUE" N,M; "INTEGER" N,M; "ARRAY" S;
"CODE" 36020;                                        "COMMENT"
```

```
 "PROCEDURE" SNDREMEZ(N,M,S,G,EM);
 "VALUE" N,M; "INTEGER" N,M; "ARRAY" S,G,EM;
 "CODE" 36021;
 "PROCEDURE" ERRPOL(N,M,E,CO,S,Y,FY,G);
 "VALUE" N,M,E;"INTEGER" N,M;
 "REAL" E;
 "INTEGER" "ARRAY" S;"ARRAY" CO,Y,FY,G;
"COMMENT"ERRPOL DELIVERS THE VALUE OF
CO[0]+CO[1]*Y[I]+...+CO[N]*Y[I]**N - FY[I]
IN G[I] FOR I=0,1,...M AND I NOT EQUAL S[J],J=0,1,...N+1.
FOR J=0,1,...N+1 THEN G[S[J]]:=(-1)**J*E.
THE INTEGERS S[J],FOR J=0,1,...N+1 ARE A SUBSET OF 0,1,...M;
 "BEGIN" "INTEGER" J,K,NP1,SJM1,SJ,SO,UP;
 NP1:=N+1;SO:=SJM1:=S[0];
 G[SO]:=E;
 "FOR" J:=1 "STEP" 1 "UNTIL" NP1 "DO"
 "BEGIN" SJ:=S[J];UP:=SJ-1;
 "FOR" K:= SJM1+1 "STEP" 1 "UNTIL" UP "DO"
    G[K]:=FY[K]-POL(N,Y[K],CO);
 G[SJ]:=E:=-E;
 SJM1:=SJ;
 "END" J;
 "FOR" K:= SO-1 "STEP"-1 "UNTIL" 0 "DO"
    G[K]:=FY[K]-POL(N,Y[K],CO);
 "FOR" K:= SJ+1 "STEP" 1 "UNTIL" M "DO"
    G[K]:=FY[K]-POL(N,Y[K],CO);
 "END" ERRPOL;

 INI(NP1,M,S);
 MI:=EM[2];
 ABSE:= 0;
 "FOR" COUNT:= 1, COUNT + 1 "WHILE" COUNT <= MI & ABSE > ABSEH "DO"
 "BEGIN"
 POMK:=1;
 "FOR" K:= 0 "STEP" 1 "UNTIL" NP1 "DO"
 "BEGIN" X[K]:= Y[S[K]]; COEF[K]:= FY[S[K]]; B[K]:= POMK;
     POMK:=-POMK "END";
 NEWTON(NP1,X,COEF); NEWTON(NP1,X,B);
 EM[0]:=
 E:= COEF[NP1]/B[NP1];
 ELMVEC(0,N,0,COEF,B,-E);
 NEWGRN(N,X,COEF);
 ERRPOL(N,M,E,COEF,S,Y,FY,G);
 SNDREMEZ(NP1,M,S,G,EM);
    ABSEH:=ABSE;      ABSE:=ABS(E);
 CNT:=COUNT;
 "END" WHILE COUNT;
 EM[2]:=MI;
 EM[3]:=CNT;
 DUPVEC(0,N,0,CO,COEF);
 "END";
 "END" MINMAXPOL;
        "EOP"
```

```
"CODE"36020;
"PROCEDURE" INI(N,M,S);
"VALUE" N,M;"INTEGER" N,M;
"INTEGER" "ARRAY" S;
 "COMMENT" INI DELIVERS (MONOTONE) THE ROUNDED VALUES
 OF THE ARGUMENTS,WHERE THE CHEBYSHEV POLYNOMIAL
   OF DEGREE N(TRANSFORMED TO THE INTERVAL [O,M],M>=N)
 ATTAINS ITS MAXIMUM VALUES,
 IN INTEGER ARRAY S[O:N];
"BEGIN""INTEGER"I,J,K,L;"REAL"PIN2;
    PIN2:=ARCTAN(1)*2/N;
    K:=0;L:=N-1;J:=S[0]:=0;S[N]:=M;
 "FOR" K:=K+1 "WHILE" K <  L "DO"
    "BEGIN"I:=SIN(K*PIN2)**2*M;
         J:=S[K]:="IF"I<=J"THEN"J+1"ELSE"I;
         S[L]:=M-J;L:=L-1
"END"K;
"IF"L*2=N"THEN"S[L]:=M/2;
 "END" INI;
        "EOP"


 "CODE"36021;
 "PROCEDURE" SNDREMEZ(N,M,S,G,EM);
 "VALUE" N,M;"INTEGER" N,M;
 "INTEGER" "ARRAY" S; "ARRAY" G,EM;
 "COMMENT" SNDREMEZ EXCHANGES ATMOST N+1 NUMBERS ,GIVEN IN
           INTEGER ARRAY S[O:N], WITH NUMBERS OUT OF THE
           REFERENCE SET 0,...M, UNDER THE CONDITIONS:
      I.   THE ALTERNANCE PROPERTY OF THE FUNCTIONVALUES G[S[J]],
           J=0,...N IS PRESERVED.
      II.  !G[S[J]]!>=!EM[0]!,J=0,...N.
      III. THE FIRST INDEX K , WITH G[K]=INFINITY NORM OF G,
           IS ONE OF THE RESULTING NUMBERS S[0],...S[N].
           IN ARRAY G[O:M] ONE MUST GIVE ERROR FUNCTION VALUES.
           MOREOVER,
           EM[1]:=INFINITY NORM OF G,
      THE PROCEDURE INFNRMVEC IS USED;
 "BEGIN" "INTEGER" SO,SN,SJP1,I,J,K,UP,INDEXMAX,LOW,NM1;
    "REAL" MAX,MSJP1,HI,HJ,HE,ABSE,H;
 "REAL" "PROCEDURE" INFNRMVEC(L,U,K,A);
 "VALUE" L,U; "INTEGER" L,U,K; "ARRAY" A;
 "CODE" 31061;
     INDEX MAX:=SO:=SJP1:=S[0];
     HE:=EM[0];LOW:=SO+1;
     MAX:=MSJP1:=ABSE:=ABS(HE);
     NM1:=N-1;
                                                          "COMMENT"
```

```
;
     "FOR" J:= 0 "STEP" 1 "UNTIL" NM1 "DO"
     "BEGIN"
        UP:= S[J+1]-1;
        H:= INFNRMVEC(LOW,UP,I,G);
        "IF" H > MAX  "THEN" "BEGIN" MAX:= H; INDEX MAX:= I "END";
        "IF" H > ABSE "THEN"
        "BEGIN" "IF" HE * G[I] > 0 "THEN"
           "BEGIN" S[J]:= "IF" MSJP1 < H "THEN" I "ELSE" SJP1;
              SJP1:= S[J+1]; MSJP1:= ABSE
           "END" "ELSE"
           "BEGIN" S[J]:= SJP1; SJP1:= I; MSJP1:= H "END"
        "END" "ELSE"
        "BEGIN" S[J]:=SJP1; SJP1:=S[J+1]; MSJP1:= ABSE "END";
        HE:=-HE;LOW:=UP+2;
    "END" FOR J; SN:= S[N]; S[N]:= SJP1;

    HI:=INFNRMVEC(0,S0-1,I,G);
    HJ:=INFNRMVEC(SN+1,M,J,G);
    "IF" J  > M  "THEN" J:=M;
    "IF" HI > HJ "THEN"
    "BEGIN" "IF" HI > MAX "THEN" "BEGIN" MAX:= HI; INDEXMAX:= I "END";
       "IF" SIGN(G[I]) =  SIGN(G[S[0]]) "THEN"
       "BEGIN" "IF" HI > ABS(G[S[0]])   "THEN"
          "BEGIN" S[0]:= I;
             "IF" G[J]/G[S[N]] > 1 "THEN" S[N]:=J
          "END"
       "END" "ELSE"
       "IF" HI > ABS(G[S[N]]) "THEN"
       "BEGIN" S[N]:= "IF" G[J]/G[S[NM1]] > 1  "THEN" J "ELSE" S[NM1];
          "FOR" K:= NM1 "STEP" -1 "UNTIL" 1 "DO" S[K]:= S[K-1];
          S[0]:= I
       "END"
    "END" "ELSE"
    "BEGIN" "IF" HJ > MAX "THEN" "BEGIN" MAX:= HJ; INDEXMAX:= J "END";
       "IF" SIGN(G[J]) = SIGN(G[S[N]]) "THEN"
       "BEGIN" "IF" HJ > ABS(G[S[N]])  "THEN"
          "BEGIN" S[N]:= J; "IF" G[I]/G[S[0]] > 1 "THEN"S[0]:=I "END"
       "END" "ELSE"
       "IF" HJ > ABS(G[S[0]]) "THEN"
       "BEGIN" S[0]:= "IF" G[I]/G[S[1]] > 1 "THEN" I "ELSE" S[1];
          "FOR" K:= 1 "STEP" 1 "UNTIL" NM 1 "DO" S[K]:= S[K+1];
          S[N]:= J
       "END"
    "END" RANDGEBIEDEN;
    EM[1]:=MAX;
 "END" SNDREMEZ;
        "EOP"
```

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP34231	ABSMAXM	AMX=
		DIM=2
PAR=DP	ALGEPP	MESSAGE
ABSMAXM	AMX=	*NOREF*
INFNRMC		
INFNRMR		
INFNRMV		
MATNRM=	ASM=	*NOREF*
ONENRMC		
ONENRMR		
ONENRMV		
COLCST	CML=	*NOREF*
LNGRESV		
RESVEC		
ROWCST		
VECCST		
CP33191	COLCST	CML=
		DIM=2
CP31370	CONST=	*NOREF*
CP31371		
CP34150		
CP34432		
CP34436		
CP34453		
CP34501		
CP34502		
CP34703		
CP35111		
CP35116		
CP35120		
CP34214	CP31013	*NOREF*
CP34215		
CP35080	CP31040	*NOREF*
CP36022		
CP31254	CP31044	*NOREF*
CP31253	CP31045	*NOREF*
CP35023	CP31046	*NOREF*
CP35080		
CP35084		
CP35085		
CP31252	CP31049	*NOREF*
CP36022	CP31050	ELMVEC

REFD BY	PROGRAM	REFS TO
CP31053	CP31051	*NOREF*
CP31054	CP31052	*NOREF*
NOREF	CP31053	CP31051
		CP31250
NOREF	CP31054	CP31052
		CP31250
CP34600	CP31070	ELMCOLV
CP34602		TAMVEC
CP34605		
CP34606		
NOREF	CP31071	ELMCOL
		TAMMAT
NOREF	CP31072	ELMCOLR
		MATMAT
CP34600	CP31073	ELMROWV
CP34607		MATVEC
CP34608		
CP34609		
CP34610		
NOREF	CP31074	ELMROWC
		MATMAT
NOREF	CP31075	ELMROW
		MATTAM
CP31094	CP31090	*NOREF*
CP31094	CP31091	*NOREF*
NOREF	CP31094	CP31090
		CP31091
CP31097	CP31095	*NOREF*
NOREF	CP31097	CP31095
NOREF	CP31100	CP31109
		LNGMUL
		LNGSUB
CP31100	CP31109	CP31110
CP31109	CP31110	LNGDIV
		LNGMUL
CP31204	CP31202	*NOREF*
NOREF	CP31204	CP31202

REFD BY	PROGRAM	REFS TO
CP31243	CP31242	*NOREF*
CP31250		
NOREF	CP31243	CP31242
CP31053	CP31250	CP31242
CP31054		
NOREF	CP31252	CP31049
NOREF	CP31253	CP31045
NOREF	CP31254	CP31044
CP31370	CP31362	CP34160
CP31371		DUPVEC
NOREF	CP31363	DUPVEC
		INFNRMV
NOREF	CP31364	CP34151
CP31425	CP31370	CONST=
		CP31362
CP31427	CP31371	CONST=
		CP31362
NOREF	CP31425	CP31370
		CP35061
NOREF	CP31427	CP31371
		CP35061
NOREF	CP31502	CP34018
NOREF	CP31504	CP34018
NOREF	CP31507	CP34418
NOREF	CP31509	CP34418
		DPMUL
CP32020	CP32010	*NOREF*
NOREF	CP32020	CP32010
NOREF	CP33016	CP34150
NOREF	CP33017	CP34150
NOREF	CP33018	CP34150
NOREF	CP33040	VECVEC
NOREF	CP33050	CP34150

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
	CP33050	DUPVEC
		ELMVEC
		INIVEC
		VECVEC
NOREF	CP33061	CP34301
		DUPVEC
		ELMVEC
		INIVEC
		MULVEC
		VECVEC
NOREF	CP33066	DUPMAT
		ELMCOL
NOREF	CP33070	CP34051
		CP34300
		ELMVEC
NOREF	CP33080	CP34051
		CP34300
		MATVEC
NOREF	CP33120	CP34051
		CP34300
		MATVEC
		VECVEC
NOREF	CP33130	CP34051
		CP34300
		VECVEC
NOREF	CP33131	CP34051
		CP34300
		MATMAT
		MATVEC
		VECVEC
NOREF	CP33132	CP34051
		CP34300
		DUPVEC
		ELMVEC
		INIVEC
		MATVEC
		MULROW
		MULVEC
		VECVEC
NOREF	CP33135	CP34051
		CP34300
		DUPMAT
		DUPROWV
		DUPVEC
		ELMROW
		ELMVEC
		INIMAT

REFD BY	PROGRAM	REFS TO
	CP33135	INIVEC
		MATMAT
		MATVEC
		MULROW
		MULVEC
		VECVEC
NOREF	CP33160	CP34061
		CP34231
		MATMAT
		MATVEC
		VECVEC
CP33171	CP33170	*NOREF*
NOREF	CP33171	CP33170
		CP34150
		CP35113
		CP35120
		CP35122
NOREF	CP33191	COLCST
		CP34061
		CP34231
		DUPVEC
		ELMROW
		ELMVEC
		MATMAT
		MATVEC
		MULVEC
		VECVEC
NOREF	CP33303	CP34333
NOREF	CP33314	DUPVEC
CP31502	CP34018	SEQVEC
CP31504		VECVEC
CP34214		
CP34215		
CP34401		
CP34709		
NOREF	CP34050	ICHROW
		MATMAT
		MATTAM
CP33070	CP34051	MATVEC
CP33080		
CP33120		
CP33130		
CP33131		
CP33132		
CP33135		
CP34061		
CP34301		

REFD BY	PROGRAM	REFS TO
CP34444	CP34051	
CP34235	CP34053	DUPCOLV
CP34302		ICHCOL
		MATMAT
CP33160	CP34061	CP34051
CP33191		
CP34232		
CP34243		
CP34250		
NOREF	CP34071	ELMVEC
		VECVEC
CP34200	CP34130	ELMCOL
		ICHCOL
		TAMMAT
CP34135	CP34131	ELMVECC
CP34200		MATVEC
CP34441		TAMVEC
CP34135	CP34132	TAMVEC
CP34200		VECVEC
CP34135	CP34134	ELMCOL
CP34441		ICHCOL
		TAMMAT
NOREF	CP34135	CP34131
		CP34132
		CP34134
CP34441	CP34136	CP34400
		ICHCOL
		ICHROW
		ICHROWC
NOREF	CP34137	ELMCOL
		ICHCOL
		MATMAT
		TAMMAT
NOREF	CP34138	ELMVECC
		ICHCOL
		LNGMATV
		LNGSUB
		LNGTAMV
		MATVEC
		TAMVEC
		VECVEC
CP34153	CP34140	ELMCOL
CP34154		ELMCOLV
CP34162		ELMVECC

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP34163	CP34140	MATMAT
		TAMMAT
		TAMVEC
CP34154	CP34141	ELMCOL
		TAMMAT
CP34163	CP34142	ELMCOL
		TAMMAT
CP34155	CP34143	ELMVEC
CP34156		SEQVEC
CP34164		VECVEC
CP34156	CP34144	ELMVEC
		VECVEC
CP33016	CP34150	CONST=
CP33017		
CP33018		
CP33050		
CP33171		
CP34151		
CP31364	CP34151	CP34150
CP34153		
CP34154		
CP34155		
CP34156		
CP34368		
CP34369		
CP34154	CP34152	ELMVECC
CP34156		TAMVEC
CP34369		VECVEC
NOREF	CP34153	CP34140
		CP34151
NOREF	CP34154	CP34140
		CP34141
		CP34151
		CP34152
NOREF	CP34155	CP34143
		CP34151
CP34214	CP34156	CP34143
		CP34144
		CP34151
		CP34152
CP31362	CP34160	*NOREF*
CP34162		
CP34164		
CP34370		

REFD BY	PROGRAM	REFS TO
CP34163	CP34161	ROTCOL
CP34371		
NOREF	CP34162	CP34140
		CP34160
CP36401	CP34163	CP34140
		CP34142
		CP34161
NOREF	CP34164	CP34143
		CP34160
CP34182	CP34170	ICHCOL
CP34184		ICHROW
CP34187		MATMAT
CP34192		MATVEC
CP34194		
NOREF	CP34171	MATVEC
CP34184	CP34172	ICHROW
CP34187		TAMVEC
CP34194		
CP34182	CP34173	ICHCOL
CP34184		ICHROW
CP34187		MATTAM
CP34192		TAMMAT
CP34194		
CP34184	CP34174	ICHROW
CP34187		
CP34194		
CP34362		
CP34182	CP34180	ROTCOL
CP34184		ROTROW
CP34184	CP34181	MATVEC
CP34194		VECVEC
NOREF	CP34182	CP34170
		CP34173
		CP34180
CP34184	CP34183	*NOREF*
CP34187		
NOREF	CP34184	CP34170
		CP34172
		CP34173
		CP34174
		CP34180
		CP34181
		CP34183

REFD BY	PROGRAM	REFS TO
CP34187	CP34186	MATVEC
		ROTCOL
		ROTROW
NOREF	CP34187	CP34170
		CP34172
		CP34173
		CP34174
		CP34183
		CP34186
CP34192	CP34190	*NOREF*
CP34194		
CP34500		
CP34194	CP34191	MATVEC
		TAMVEC
		VECVEC
NOREF	CP34192	CP34170
		CP34173
		CP34190
CP34194	CP34193	*NOREF*
NOREF	CP34194	CP34170
		CP34172
		CP34173
		CP34174
		CP34181
		CP34190
		CP34191
		CP34193
NOREF	CP34200	CP34130
		CP34131
		CP34132
		VECVEC
CP34214	CP34210	DUPVEC
CP34215		ELMVEC
		VECVEC
CP34214	CP34211	ELMVEC
CP34214	CP34212	*NOREF*
CP34215		
CP34214	CP34213	*NOREF*
CP34215		
NOREF	CP34214	CP31013
		CP34018
		CP34156
		CP34210
		CP34211

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
	CP34214	CP34212
		CP34213
		DUPVEC
		ELMVEC
		INIVEC
		MATVEC
		MULVEC
		TAMVEC
		VECVEC
NOREF	CP34215	CP31013
		CP34016
		CP34210
		CP34212
		CP34213
		DUPVEC
		ELMVEC
		INIVEC
		MULVEC
		VECVEC
NOREF	CP34220	ELMVEC
		VECVEC
CP33160	CP34231	ABSMAXM
CP33191		ELMROW
CP34232		ICHCOL
CP34236		ICHROW
CP34242		MAXELMR
CP34244		ROWCST
CP34251		
CP34252		
NOREF	CP34232	CP34061
		CP34231
CP34236	CP34235	CP34053
CP34244		ICHROW
NOREF	CP34236	CP34231
		CP34235
CP34242	CP34240	MATVEC
CP34252		
CP34242	CP34241	*NOREF*
CP34244		
CP34243	CP34242	CP34231
		CP34240
		CP34241
NOREF	CP34243	CP34061
		CP34242
NOREF	CP34244	CP34231

REFD BY	PROGRAM	REFS TO
	CP34244	CP34235
		CP34241
CP34251	CP34250	CP34061
CP34253		DUPVEC
		INIVEC
		LNGMATV
NOREF	CP34251	CP34231
		CP34250
		DUPMAT
CP34254	CP34252	CP34231
		CP34240
CP34254	CP34253	CP34250
NOREF	CP34254	CP34252
		CP34253
		DUPMAT
CP34272	CP34260	ELMCOL
CP34273		ELMROW
		MATTAM
		TAMMAT
CP34273	CP34261	ELMCOL
		MATMAT
CP34273	CP34262	ELMCOL
		TAMMAT
CP34272	CP34270	*NOREF*
CP34273	CP34271	ROTCOL
NOREF	CP34272	CP34260
		CP34270
CP34281	CP34273	CP34260
CP34283		CP34261
CP34285		CP34262
CP34287		CP34271
CP34432		
CP34440		
CP34281	CP34280	MATVEC
		TAMVEC
NOREF	CP34281	CP34273
		CP34280
CP34283	CP34282	MATVEC
		TAMVEC
NOREF	CP34283	CP34273

REFD BY	PROGRAM	REFS TO
	CP34283	CP34282
CP34285	CP34284	ICHCOL
NOREF	CP34285	CP34273
		CP34284
CP34287	CP34286	MATVEC
NOREF	CP34287	CP34273
		CP34286
CP34293	CP34291	ELMROW
		ICHROW
		ICHROWC
CP34293	CP34292	ELMVECR
		MATVEC
NOREF	CP34293	CP34291
		CP34292
CP33070	CP34300	ICHROW
CP33080		MATMAT
CP33120		MATTAM
CP33130		
CP33131		
CP33132		
CP33135		
CP34301		
CP34302		
CP34444		
CP33061	CP34301	CP34051
		CP34300
NOREF	CP34302	CP34053
		CP34300
CP34702	CP34303	*NOREF*
CP34392	CP34310	TAMMAT
CP34402		
CP34393	CP34311	VECVEC
CP34403		
NOREF	CP34320	ELMVEC
		ICHVEC
		VECVEC
CP34430	CP34322	ELMVEC
		ICHVEC
		VECVEC
CP34333	CP34330	VECVEC

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP34333	CP34332	SCAPRD1
		VECVEC
CP33303	CP34333	CP34330
		CP34332
CP34501	CP34340	*NOREF*
CP34345	CP34341	*NOREF*
CP34352		
CP34353		
CP34365		
CP34366		
CP34345	CP34342	*NOREF*
CP34373		
CP34601		
CP34345	CP34343	*NOREF*
CP34501		
CP34355	CP34344	*NOREF*
CP34363		
CP34366		
CP34372	CP34345	CP34341
CP34373		CP34342
		CP34343
CP34360	CP34352	CP34341
CP34366		
CP34372		
CP34373		
CP34365	CP34353	CP34341
CP34366		
CP34367		
CP34373		
CP34373	CP34354	MATVEC
CP34366	CP34355	CP34344
		TAMMAT
CP34366	CP34356	CP34377
CP34367		TAMMAT
CP34372	CP34357	*NOREF*
CP34373		
CP34372	CP34358	*NOREF*
CP34373		
CP34374	CP34359	MATTAM
CP34375		

REFD BY	PROGRAM	REFS TO
CP34375	CP34360	CP34352
CP34374	CP34361	ICHCOL
CP34375		ICHROW
		MATTAM
		TAMMAT
CP34375	CP34362	CP34174
CP34369	CP34363	CP34344
CP34371		ELMCOL
		ELMCOLR
		ELMCOLV
		ELMROW
		ELMROWC
		ELMROWV
		ELMVECC
		ELMVECR
		MATMAT
		MATTAM
		MATVEC
		TAMMAT
		TAMVEC
CP34368	CP34364	ELMCOL
CP34370		ELMCOLR
		ELMCOLV
		ELMROW
		ELMROWC
		ELMROWV
		ELMVECC
		ELMVECR
		MATMAT
		MATTAM
		MATVEC
		TAMMAT
		TAMVEC
CP34369	CP34365	CP34341
CP34371		CP34353
		ELMCOL
		ELMCOLR
		MATMAT
		TAMMAT
CP34374	CP34366	CP34341
CP34375		CP34344
		CP34352
		CP34353
		CP34355
		CP34356
		ELMROWC
		MATMAT
CP34375	CP34367	CP34353
		CP34356

REFD BY	PROGRAM	REFS TO
NOREF	CP34368	CP34151
		CP34364
NOREF	CP34369	CP34151
		CP34152
		CP34363
		CP34365
NOREF	CP34370	CP34160
		CP34364
NOREF	CP34371	CP34161
		CP34363
		CP34365
CP34374	CP34372	CP34345
		CP34352
		CP34357
		CP34358
CP34375	CP34373	CP34342
		CP34345
		CP34352
		CP34353
		CP34354
		CP34357
		CP34358
		MATVEC
NOREF	CP34374	CP34359
		CP34361
		CP34366
		CP34372
NOREF	CP34375	CP34359
		CP34360
		CP34361
		CP34362
		CP34366
		CP34367
		CP34373
NOREF	CP34376	ELMVECC
CP34356	CP34377	ELMCOL
NOREF	CP34378	ELMROWV
CP34392	CP34390	MATVEC
		TAMVEC
CP34393	CP34391	SEQVEC
		VECVEC
NOREF	CP34392	CP34310
		CP34390

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
NOREF	CP34393	CP34311
		CP34391
CP34136	CP34400	DUPVECR
CP34402		MATVEC
		TAMVEC
CP34403	CP34401	CP34018
		SEQVEC
NOREF	CP34402	CP34310
		CP34400
NOREF	CP34403	CP34311
		CP34401
CP31507	CP34418	LNGSEQV
CP31509		LNGVECV
CP34422	CP34420	*NOREF*
CP34422	CP34421	*NOREF*
NOREF	CP34422	CP34420
		CP34421
CP34425	CP34423	*NOREF*
CP34425	CP34424	*NOREF*
NOREF	CP34425	CP34423
		CP34424
CP34431	CP34430	CP34322
		DUPVEC
		ELMVEC
		MULVEC
		VECVEC
NOREF	CP34431	CP34430
		CP34439
NOREF	CP34432	CONST=
		CP34273
		DUPCOLV
		DUPMAT
		DUPVEC
		ELMVECC
		ICHROWC
		INIMAT
		INIVEC
		MATTAM
		MCC
		MULCOL
		MULROW
		TAMMAT

REFD BY	PROGRAM	REFS TO
	CP34432	VECVEC
NOREF	CP34436	CONST=
CP34431	CP34439	*NOREF*
CP34444	CP34440	CP34273
		DUPVEC
		MATTAM
		MATVEC
		MULCOL
		TAMVEC
		VECVEC
NOREF	CP34441	CP34131
		CP34134
		CP34136
		DUPVEC
		ELMVEC
		VECVEC
NOREF	CP34444	CP34051
		CP34300
		CP34440
		DUPMAT
		DUPVEC
		ELMVEC
		INIMAT
		INIVEC
		MATVEC
		MULROW
		MULVEC
		VECVEC
NOREF	CP34445	VECVEC
NOREF	CP34453	CONST=
NOREF	CP34500	CP34190
NOREF	CP34501	CONST=
		CP34340
		CP34343
NOREF	CP34502	CONST=
NOREF	CP34600	CP31070
		CP31073
		CP34602
		CP34604
		CP34605
		CP34606
		CP34608
		CP34610
		CP34611
		ELMCOL

REFD BY	PROGRAM	REFS TO
NOREF	CP34601	CP34342
		CP34602
		CP34603
		CP34605
		CP34606
		CP34607
		CP34609
		CP34611
		MATMAT
CP34600	CP34602	CP31070
CP34601		TAMMAT
CP34601	CP34603	CP34605
		CP34607
CP34600	CP34604	CP34605
		CP34608
CP34600	CP34605	CP31070
CP34601		
CP34603		
CP34604		
CP34600	CP34606	CP31070
CP34601		
CP34601	CP34607	CP31073
CP34603		
CP34600	CP34608	CP31073
CP34604		
CP34601	CP34609	CP31073
CP34600	CP34610	CP31073
CP34600	CP34611	*NOREF*
CP34601		
CP34706	CP34700	TAMMAT
CP34710		
CP34707	CP34701	VECVEC
CP34711		
NOREF	CP34702	CP34303
NOREF	CP34703	CONST=
		UNDERFL
CP34706	CP34704	MATVEC
		TAMVEC
CP34707	CP34705	SEQVEC
		VECVEC

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
NOREF	CP34706	CP34700
		CP34704
NOREF	CP34707	CP34701
		CP34705
CP34710	CP34708	DUPVECR
		MATVEC
		TAMVEC
CP34711	CP34709	CP34018
		SEQVEC
NOREF	CP34710	CP34700
		CP34708
NOREF	CP34711	CP34701
		CP34709
CP35022	CP35021	CP35022
CP35021	CP35022	CP35021
NOREF	CP35023	CP31046
CP35028	CP35027	CP35028
CP35027	CP35028	CP35027
CP35040	CP35038	CP35040
CP35105		
CP35038	CP35040	CP35038
CP35103		
CP35053	CP35050	CP35061
CP35054		
NOREF	CP35051	CP35053
		CP35054
NOREF	CP35052	CP35053
		CP35054
CP35051	CP35053	CP35050
CP35052		CP35055
		CP35056
CP35051	CP35054	CP35050
CP35052		CP35055
		CP35056
CP35053	CP35055	*NOREF*
CP35054		
CP35053	CP35056	*NOREF*

REFD BY	PROGRAM	REFS TO
CP35054	CP35056	
CP35061	CP35060	*NOREF*
CP35071		
CP35075		
CP35181		
CP35191		
CP31425	CP35061	CP35060
CP31427		CP35062
CP35050		
CP35180		
CP35193		
CP35061	CP35062	*NOREF*
CP35072	CP35071	CP35060
CP35073		CP35073
		CP35111
NOREF	CP35072	CP35071
CP35071	CP35073	CP35071
CP35074		
NOREF	CP35074	CP35073
CP35076	CP35075	CP35060
		CP35077
		CP35111
NOREF	CP35076	CP35075
CP35075	CP35077	*NOREF*
CP35086	CP35080	CP31040
		CP31046
		CP35083
CP35080	CP35083	*NOREF*
CP35085	CP35084	CP31046
		CP35085
CP35084	CP35085	CP31046
		CP35084
CP35087	CP35086	CP35080
		CP35087
CP35086	CP35087	CP35086
NOREF	CP35102	CP35104
NOREF	CP35103	CP35040

REFD BY	PROGRAM	REFS TO
CP35102	CP35104	*NOREF*
NOREF	CP35105	CP35038
CP35071	CP35111	CONST=
CP35075		
CP35112		
CP35113		
CP35154		
CP35181		
CP35191		
NOREF	CP35112	CP35111
CP33171	CP35113	CP35111
NOREF	CP35114	CP35130
NOREF	CP35116	CONST=
		CP35130
CP33171	CP35120	CONST=
		OVERFLO
CP33171	CP35122	*NOREF*
CP35114	CP35130	*NOREF*
CP35116		
CP35145	CP35140	*NOREF*
NOREF	CP35145	CP35140
CP35180	CP35150	CP35185
NOREF	CP35152	CP35154
NOREF	CP35153	CP35155
CP35152	CP35154	CP35111
CP35190		CP35185
CP35193		
CP35153	CP35155	*NOREF*
CP35163	CP35160	CP35165
CP35165		
CP35163	CP35161	CP35166
CP35166		
CP35180	CP35162	CP35185
CP35164	CP35163	CP35160
CP35165		CP35161
CP35166		CP35165

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP35181	CP35163	CP35166
NOREF	CP35164	CP35163
CP35160	CP35165	CP35160
CP35163		CP35163
CP35183		
CP35161	CP35166	CP35161
CP35163		CP35163
CP35183		
CP35175	CP35170	CP35175
CP35176	CP35171	CP35176
CP35190	CP35172	CP35177
CP35174	CP35173	CP35179
CP35178		
CP35191		
NOREF	CP35174	CP35173
CP35170	CP35175	CP35170
CP35171	CP35176	CP35171
CP35172	CP35177	CP35185
CP35193		
CP35173	CP35178	CP35173
CP35179		
CP35194		
NOREF	CP35179	CP35178
CP35183	CP35180	CP35061
		CP35150
		CP35162
		CP35185
CP35182	CP35181	CP35060
CP35183		CP35111
		CP35163
		CP35183
NOREF	CP35182	CP35181
CP35181	CP35183	CP35165
CP35184		CP35166
		CP35180
		CP35181
NOREF	CP35184	CP35183
CP35150	CP35185	*NOREF*
CP35154		
CP35162		
CP35177		
CP35180		
CP35193		
NOREF	CP35190	CP35154
		CP35172
		CP35193
CP35192	CP35191	CP35060
CP35194		CP35111
		CP35173
		CP35194
NOREF	CP35192	CP35191
CP35190	CP35193	CP35061
		CP35154
		CP35177
		CP35185
CP35191	CP35194	CP35178
CP35195		CP35191
NOREF	CP35195	CP35194
CP36022	CP36010	*NOREF*
CP36022	CP36020	*NOREF*
CP36022	CP36021	INFNRMV
NOREF	CP36022	CP31040
		CP31050
		CP36010
		CP36020
		CP36021
		DUPVEC
		ELMVEC
NOREF	CP36401	CP34163
		CP36402
		CP36403
		CP36404
		CP36405
		INFNRMM
		LNGMATM
		LNGMATV
		LNGTAMM
		MATMAT
		TAMMAT
		VECVEC
CP36401	CP36402	ELMCOL
	CP36402	TAMMAT
CP36401	CP36403	*NOREF*
CP36401	CP36404	*NOREF*
CP36401	CP36405	*NOREF*
DUPCOLV	DIM=1	*NOREF*
DUPROWV		
DUPVEC		
DUPVECC		
DUPVECR		
ELMCOLV		
ELMROWV		
ELMVEC		
ELMVECC		
ELMVECR		
FULMATV		
FULTAMV		
ICHSEQV		
ICHVEC		
INFNRMV		
INIVEC		
LNGFULM		
LNGFULT		
LNGMATV		
LNGRESV		
LNGSCAP		
LNGSEQV		
LNGTAMV		
LNGVECV		
MATVEC		
MULVEC		
ONENRMV		
RESVEC		
SCAPRD1		
SEQVEC		
TAMVEC		
VECCST		
VECVEC		
ABSMAXM	DIM=2	*NOREF*
COLCST		
DUPCOLV		
DUPMAT		
DUPROWV		
DUPVECC		
DUPVECR		
ELMCOL		
ELMCOLR		
ELMCOLV		
ELMROW		
ELMROWC		
ELMROWV		
ELMVECC		

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
ELMVECP	DIM=2	
FULMATV		
FULTAMV		
ICHCOL		
ICHROW		
ICHROWC		
INFNRMC		
INFNRMM		
INFNRMR		
INIMAT		
INIMATD		
LNGFULM		
LNGFULT		
LNGMATM		
LNGMATT		
LNGMATV		
LNGRESV		
LNGTAMM		
LNGTAMV		
MATMAT		
MATTAM		
MATVEC		
MAXELMR		
MULCOL		
MULROW		
ONENRMC		
ONENRMM		
ONENRMR		
RESVEC		
ROTCOL		
ROTROW		
ROWCST		
TAMMAT		
TAMVEC		
FULDPI=	DPI=	*NOREF*
LNGMATM		
LNGMATT		
LNGMATV		
LNGSCAP		
LNGSEQV		
LNGTAMM		
LNGTAMV		
LNGVECV		
CP31509	DPMUL	*NOREF*
CP34053	DUPCOLV	DIM=1
CP34432		DIM=2
		DUP=
CP33066	DUPMAT	DIM=2
CP33135		DUP=
CP34251		
CP34254		
CP34432		

REFD BY	PROGRAM	REFS TO
CP34444	DUPMAT	
CP33135	DUPROWV	DIM=1
		DIM=2
		DUP=
CP31362	DUPVEC	DIM=1
CP31363		DUP=
CP33050		
CP33061		
CP33132		
CP33135		
CP33191		
CP33314		
CP34210		
CP34214		
CP34215		
CP34250		
CP34430		
CP34432		
CP34440		
CP34441		
CP34444		
CP36022		
NOREF	DUPVECC	DIM=1
		DIM=2
		DUP=
CP34400	DUPVECR	DIM=1
CP34708		DIM=2
		DUP=
DUPCOLV	DUP=	*NOREF*
DUPMAT		
DUPROWV		
DUPVEC		
DUPVECC		
DUPVECR		
CP31071	ELMCOL	DIM=2
CP33066		ELM=
CP34130		
CP34134		
CP34137		
CP34140		
CP34141		
CP34142		
CP34260		
CP34261		
CP34262		
CP34363		
CP34364		
CP34365		
CP34377		
CP34600		

REFD BY	PROGRAM	REFS TO
CP36402	ELMCOL	
CP31072	ELMCOLR	DIM=2
CP34363		ELM=
CP34364		
CP34365		
CP31070	ELMCOLV	DIM=1
CP34140		DIM=2
CP34363		ELM=
CP34364		
CP31075	ELMROW	DIM=2
CP33135		ELM=
CP33191		
CP34231		
CP34260		
CP34291		
CP34363		
CP34364		
CP31074	ELMROWC	DIM=2
CP34363		ELM=
CP34364		
CP34366		
CP31073	ELMROWV	DIM=1
CP34363		DIM=2
CP34364		ELM=
CP34378		
CP31050	ELMVEC	DIM=1
CP33050		ELM=
CP33061		
CP33070		
CP33132		
CP33135		
CP33191		
CP34071		
CP34143		
CP34144		
CP34210		
CP34211		
CP34214		
CP34215		
CP34220		
CP34320		
CP34322		
CP34430		
CP34441		
CP34444		
CP36022		
CP34131	ELMVECC	DIM=1
CP34138		DIM=2
CP34140		ELM=

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP34152	ELMVECC	
CP34363		
CP34364		
CP34376		
CP34432		
CP34292	ELMVECP	DIM=1
CP34363		DIM=2
CP34364		ELM=
ELMCOL	ELM=	*NOREF*
ELMCOLP		
ELMCOLV		
ELMROW		
ELMROWC		
ELMROWV		
ELMVEC		
ELMVECC		
ELMVECP		
MAXELMR	EMX=	*NOREF*
LNGFULM	FULDPI=	DPI=
LNGFULT		
LNGRESV		
FULMATV	FULINP=	INP=
FULTAMV		
RESVEC		
NOREF	FULMATV	DIM=1
		DIM=2
		FULINP=
		INI=
NOREF	FULTAMV	DIM=1
		DIM=2
		FULINP=
		INI=
CP34053	ICHCOL	DIM=2
CP34130		ICH=
CP34134		
CP34136		
CP34137		
CP34138		
CP34170		
CP34173		
CP34231		
CP34284		
CP34361		
CP34050	ICHROW	DIM=2
CP34136		ICH=
CP34170		
CP34172		

REFD BY	PROGRAM	REFS TO
CP34173	ICHROW	
CP34174		
CP34231		
CP34235		
CP34291		
CP34300		
CP34361		
CP34136	ICHROWC	DIM=2
CP34291		ICH=
CP34432		
NOREF	ICHSEQ	ICH=
		SEQMAT=
NOREF	ICHSEQV	DIM=1
		ICH=
		SEQMAT=
CP34320	ICHVEC	DIM=1
CP34322		ICH=
ICHCOL	ICH=	*NOREF*
ICHROW		
ICHROWC		
ICHSEQ		
ICHSEQV		
ICHVEC		
NOREF	INFNRMC	AMY=
		DIM=2
CP36401	INFNRMM	DIM=2
		MATNRM=
NOREF	INFNRMR	AMX=
		DIM=2
CP31363	INFNRMV	AMX=
CP36021		DIM=1
CP33135	INIMAT	DIM=2
CP34432		INI=
CP34444		
NOREF	INIMATD	DIM=2
		INI=
CP33050	INIVEC	DIM=1
CP33061		INI=
CP33132		
CP33135		
CP34214		
CP34215		
CP34250		
CP34432		

REFD BY	PROGRAM	REFS TO
CP34444	INIVEC	
FULMATV	INI=	*NOREF*
FULTAMV		
INIMAT		
INIMATD		
INIVEC		
LNGFULM		
LNGFULT		
FULINP=	INP=	*NOREF*
MATMAT		
MATTAM		
MATVEC		
SCAPRD1		
SEQVEC		
TAMMAT		
TAMVEC		
VECVEC		
NOREF	LNGADD	PAR=DP
CP31110	LNGDIV	PAR=DP
NOREF	LNGFULM	DIM=1
		DIM=2
		FULDPI=
		INI=
NOREF	LNGFULT	DIM=1
		DIM=2
		FULDPI=
		INI=
CP36401	LNGMATM	DIM=2
		DPI=
		PAR=DP
NOREF	LNGMATT	DIM=2
		DPI=
		PAR=DP
CP34138	LNGMATV	DIM=1
CP34250		DIM=2
CP36401		DPI=
		PAR=DP
CP31100	LNGMUL	PAR=DP
CP31110		
NOREF	LNGRESV	CML=
		DIM=1
		DIM=2
		FULDPI=
NOREF	LNGSCAP	DIM=1

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
	LNGSCAP	DPI=
		PAR=DP
CP34418	LNGSEQV	DIM=1
		DPI=
		PAR=DP
		SEQMAT=
CP31100	LNGSUB	PAR=DP
CP34138		
CP36401	LNGTAMM	DIM=2
		DPI=
		PAR=DP
CP34138	LNGTAMV	DIM=1
		DIM=2
		DFI=
		PAR=DP
CP34418	LNGVECV	DIM=1
		DPI=
		PAR=DP
CP31072	MATMAT	DIM=2
CP31074		INP=
CP33131		
CP33135		
CP33160		
CP33191		
CP34050		
CP34053		
CP34137		
CP34140		
CP34170		
CP34261		
CP34300		
CP34363		
CP34364		
CP34365		
CP34366		
CP34601		
CP36401		
INFNRMM	MATNRM=	ASM=
UNENRMM		
CP31075	MATTAM	DIM=2
CP34050		INP=
CP34173		
CP34260		
CP34300		
CP34359		
CP34361		
CP34363		
CP34364		
CP34432	MATTAM	
CP34440		
CP31073	MATVEC	DIM=1
CP33080		DIM=2
CP33120		INP=
CP33131		
CP33132		
CP33135		
CP33160		
CP33191		
CP34051		
CP34131		
CP34138		
CP34170		
CP34171		
CP34181		
CP34186		
CP34191		
CP34214		
CP34240		
CP34280		
CP34282		
CP34286		
CP34292		
CP34354		
CP34363		
CP34364		
CP34373		
CP34390		
CP34400		
CP34440		
CP34444		
CP34704		
CP34708		
CP34231	MAXELMR	DIM=2
		EMX=
CP34432	MCC	*NOREF*
ALGERR	MESSAGE	PPCALL=
CP34432	MULCOL	DIM=2
CP34440		MUL=
CP33132	MULROW	DIM=2
CP33135		MUL=
CP34432		
CP34444		
CP33061	MULVEC	DIM=1
CP33132		MUL=
CP33135		
CP33191		
CP34214		
CP34215	MULVEC	
CP34430		
CP34444		
MULCOL	MUL=	*NOREF*
MULROW		
MULVEC		
NOREF	ONENRMC	ASM=
		DIM=2
NOREF	ONENRMM	DIM=2
		MATNRM=
NOREF	ONENRMR	ASM=
		DIM=2
NOREF	ONENRMV	ASM=
		DIM=1
CP35120	OVERFLO	*NOREF*
LNGADD	PAR=DP	ALGERR
LNGDIV		
LNGMATM		
LNGMATT		
LNGMATV		
LNGMUL		
LNGSCAP		
LNGSEQV		
LNGSUB		
LNGTAMM		
LNGTAMV		
LNGVECV		
MESSAGE	PPCALL=	*NOREF*
NOREF	RESVEC	CML=
		DIM=1
		DIM=2
		FULINP=
CP34161	ROTCOL	DIM=2
CP34180		ROT=
CP34186		
CP34271		
CP34190	ROTROW	DIM=2
CP34186		ROT=
ROTCOL	ROT=	*NOREF*
ROTROW		
CP34231	ROWCST	CML=
		DIM=2

CROSS REFERENCE TABLE

REFD BY	PROGRAM	REFS TO
CP34332	SCAPRD1	DIM=1
		INP=
ICHSEQ	SEQMAT=	*NOREF*
ICHSEQV		
LNGSEQV		
SEQVEC		
CP34018	SEQVEC	DIM=1
CP34143		INP=
CP34391		SEQMAT=
CP34401		
CP34705		
CP34709		
CP31071	TAMMAT	DIM=2
CP34130		INP=
CP34134		
CP34137		
CP34140		
CP34141		
CP34142		
CP34173		
CP34260		
CP34262		
CP34310		
CP34355		
CP34356		
CP34361		
CP34363		
CP34364		
CP34365		
CP34432		
CP34602		
CP34700	TAMMAT	
CP36401		
CP36402		
CP31070	TAMVEC	DIM=1
CP34131		DIM=2
CP34132		INP=
CP34138		
CP34140		
CP34152		
CP34172		
CP34191		
CP34214		
CP34280		
CP34282		
CP34363		
CP34364		
CP34390		
CP34400		
CP34440		
CP34704		
CP34708		
CP34703	UNDERFL	*NOREF*
NOREF	VECCST	CML=
		DIM=1
CP33040	VECVEC	DIM=1
CP33050		INP=
CP33061		
CP33120		
CP33130		
CP33131		
CP33132	VECVEC	
CP33135		
CP33160		
CP33191		
CP34018		
CP34071		
CP34132		
CP34138		
CP34143		
CP34144		
CP34152		
CP34181		
CP34191		
CP34200		
CP34210		
CP34214		
CP34215		
CP34220		
CP34311		
CP34320		
CP34322		
CP34330		
CP34332		
CP34391		
CP34430		
CP34432		
CP34440		
CP34441		
CP34444		
CP34445		
CP34701		
CP34705		
CP36401		

UITGAVEN IN DE SERIE MC SYLLABUS

Onderstaande uitgaven zijn verkrijgbaar bij het Mathematisch Centrum, 2e Boerhaavestraat 49 te Amsterdam-1005, tel. 020-947272.

MCS 1.1 F. GÖBEL & J. VAN DE LUNE, *Leergang Besliskunde, deel 1: Wiskundige basiskennis*, 1965. ISBN 90 6196 014 2.

MCS 1.2 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 2: Kansberekening*, 1965. ISBN 90 6196 015 0.

MCS 1.3 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 3: Statistiek*, 1966. ISBN 90 6196 016 9.

MCS 1.4 G. DE LEVE & W. MOLENAAR, *Leergang Besliskunde, deel 4: Markovketens en wachttijden*, 1966. ISBN 90 6196 017 7.

MCS 1.5 J. KRIENS & G. DE LEVE, *Leergang Besliskunde, deel 5: Inleiding tot de mathematische besliskunde*, 1966. ISBN 90 6196 018 5.

MCS 1.6a B. DORHOUT & J. KRIENS, *Leergang Besliskunde, deel 6a: Wiskundige programmering 1*, 1968. ISBN 90 6196 032 0.

MCS 1.6b B. DORHOUT, J. KRIENS & J.TH. VAN LIESHOUT, *Leergang Besliskunde, deel 6b: Wiskundige programmering 2*, 1977. ISBN 90 6196 150 5.

MCS 1.7a G. DE LEVE, *Leergang Besliskunde, deel 7a: Dynamische programmering 1*, 1968. ISBN 90 6196 033 9.

MCS 1.7b G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7b: Dynamische programmering 2*, 1970. ISBN 90 6196 055 X.

MCS 1.7c G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7c: Dynamische programmering 3*, 1971. ISBN 90 6196 066 5.

MCS 1.8 J. KRIENS, F. GÖBEL & W. MOLENAAR, *Leergang Besliskunde, deel 8: Minimaxmethode, netwerkplanning, simulatie*, 1968. ISBN 90 6196 034 7.

MCS 2.1 G.J.R. FÖRCH, P.J. VAN DER HOUWEN & R.P. VAN DE RIET, *Colloquium Stabiliteit van differentieschema's, deel 1*, 1967. ISBN 90 6196 023 1.

MCS 2.2 L. DEKKER, T.J. DEKKER, P.J. VAN DER HOUWEN & M.N. SPIJKER, *Colloquium Stabiliteit van differentieschema's, deel 2*, 1968. ISBN 90 6196 035 5.

MCS 3.1 H.A. LAUWERIER, *Randwaardeproblemen, deel 1*, 1967. ISBN 90 6196 024 X.

MCS 3.2 H.A. LAUWERIER, *Randwaardeproblemen, deel 2*, 1968. ISBN 90 6196 036 3.

MCS 3.3 H.A. LAUWERIER, *Randwaardeproblemen, deel 3*, 1968. ISBN 90 6196 043 6.

MCS 4 H.A. LAUWERIER, *Representaties van groepen*, 1968. ISBN 90 6196 037 1.

MCS 5 J.H. VAN LINT, J.J. SEIDEL & P.C. BAAYEN, *Colloquium Discrete wiskunde*, 1968.
ISBN 90 6196 044 4.

MCS 6 K.K. KOKSMA, *Cursus ALGOL 60*, 1969. ISBN 90 6196 045 2.

MCS 7.1 *Colloquium Moderne rekenmachines, deel 1*, 1969. ISBN 90 6196 046 0.

MCS 7.2 *Colloquium Moderne rekenmachines, deel 2*, 1969. ISBN 90 6196 047 9.

MCS 8 H. BAVINCK & J. GRASMAN, *Relaxatietrillingen*, 1969.
ISBN 90 6196 056 8.

MCS 9.1 T.M.T. COOLEN, G.J.R. FÖRCH, E.M. DE JAGER & H.G.J. PIJLS, *Elliptische differentiaalvergelijkingen, deel 1*, 1970.
ISBN 90 6196 048 7.

MCS 9.2 W.P. VAN DEN BRINK, T.M.T. COOLEN, B. DIJKHUIS, P.P.N. DE GROEN, P.J. VAN DER HOUWEN, E.M. DE JAGER, N.M. TEMME & R.J. DE VOGELAERE, *Colloquium Elliptische differentiaalvergelijkingen, deel 2*, 1970.
ISBN 90 6196 049 5.

MCS 10 J. FABIUS & W.R. VAN ZWET, *Grondbegrippen van de waarschijnlijkheidsrekening*, 1970. ISBN 90 6196 057 6.

MCS 11 H. BART, M.A. KAASHOEK, H.G.J. PIJLS, W.J. DE SCHIPPER & J. DE VRIES, *Colloquium Halfalgebra's en positieve operatoren*, 1971.
ISBN 90 6196 067 3.

MCS 12 T.J. DEKKER, *Numerieke algebra*, 1971. ISBN 90 6196 068 1.

MCS 13 F.E.J. KRUSEMAN ARETZ, *Programmeren voor rekenautomaten; De MC ALGOL 60 vertaler voor de EL X8*, 1971. ISBN 90 6196 069 X.

MCS 14 H. BAVINCK, W. GAUTSCHI & G.M. WILLEMS, *Colloquium Approximatietheorie*, 1971. ISBN 90 6196 070 3.

MCS 15.1 T.J. DEKKER, P.W. HEMKER & P.J. VAN DER HOUWEN, *Colloquium Stijve differentiaalvergelijkingen, deel 1*, 1972. ISBN 90 6196 078 9.

MCS 15.2 P.A. BEENTJES, K. DEKKER, H.C. HEMKER, S.P.N. VAN KAMPEN & G.M. WILLEMS, *Colloquium Stijve differentiaalvergelijkingen, deel 2*, 1973. ISBN 90 6196 079 7.

MCS 15.3 P.A. BEENTJES, K. DEKKER, P.W. HEMKER & M. VAN VELDHUIZEN, *Colloquium Stijve differentiaalvergelijkingen, deel 3*, 1975.
ISBN 90 6196 118 1.

MCS 16.1 L. GEURTS, *Cursus Programmeren, deel 1: De elementen van het programmeren*, 1973. ISBN 90 6196 080 0.

MCS 16.2 L. GEURTS, *Cursus Programmeren, deel 2: De programmeertaal ALGOL 60*, 1973. ISBN 90 6196 087 8.

MCS 17.1 P.S. STOBBE, *Lineaire algebra, deel 1*, 1974. ISBN 90 6196 090 8.

MCS 17.2 P.S. STOBBE, *Lineaire algebra, deel 2*, 1974. ISBN 90 6196 091 6.

MCS 17.3 N.M. TEMME, *Lineaire algebra , deel 3*, 1976. ISBN 90 6196 123 8.

MCS 18 F. VAN DER BLIJ, H. FREUDENTHAL, J.J. DE IONGH, J.J. SEIDEL & A. VAN WIJNGAARDEN, *Een kwart eeuw wiskunde 1946-1971, Syllabus van de Vakantiecursus 1971*, 1974. ISBN 90 6196 092 4.

MCS 19 A. HORDIJK, R. POTHARST & J.Th. RUNNENBURG, *Optimaal stoppen van Markovketens*, 1974. ISBN 90 6196 093 2.

UITGAVEN IN DE SERIE MC SYLLABUS

Onderstaande uitgaven zijn verkrijgbaar bij het Mathematisch Centrum, 2e Boerhaavestraat 49 te Amsterdam-1005, tel. 020-947272.

MCS 1.1 F. GÖBEL & J. VAN DE LUNE, *Leergang Besliskunde, deel 1: Wiskundige basiskennis*, 1965. ISBN 90 6196 014 2.

MCS 1.2 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 2: Kansberekening*, 1965. ISBN 90 6196 015 0.

MCS 1.3 J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 3: Statistiek*, 1966. ISBN 90 6196 016 9.

MCS 1.4 G. DE LEVE & W. MOLENAAR, *Leergang Besliskunde, deel 4: Markovketens en wachttijden*, 1966. ISBN 90 6196 017 7.

MCS 1.5 J. KRIENS & G. DE LEVE, *Leergang Besliskunde, deel 5: Inleiding tot de mathematische besliskunde*, 1966. ISBN 90 6196 018 5.

MCS 1.6a B. DORHOUT & J. KRIENS, *Leergang Besliskunde, deel 6a: Wiskundige programmering 1*, 1968. ISBN 90 6196 032 0.

MCS 1.6b B. DORHOUT, J. KRIENS & J.TH. VAN LIESHOUT, *Leergang Besliskunde, deel 6b: Wiskundige programmering 2*, 1977. ISBN 90 6196 150 5.

MCS 1.7a G. DE LEVE, *Leergang Besliskunde, deel 7a: Dynamische programmering 1*, 1968. ISBN 90 6196 033 9.

MCS 1.7b G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7b: Dynamische programmering 2*, 1970. ISBN 90 6196 055 X.

MCS 1.7c G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7c: Dynamische programmering 3*, 1971. ISBN 90 6196 066 5.

MCS 1.8 J. KRIENS, F. GÖBEL & W. MOLENAAR, *Leergang Besliskunde, deel 8: Minimaxmethode, netwerkplanning, simulatie*, 1968. ISBN 90 6196 034 7.

MCS 2.1 G.J.R. FÖRCH, P.J. VAN DER HOUWEN & R.P. VAN DE RIET, *Colloquium Stabiliteit van differentieschema's, deel 1*, 1967. ISBN 90 6196 023 1.

MCS 2.2 L. DEKKER, T.J. DEKKER, P.J. VAN DER HOUWEN & M.N. SPIJKER, *Colloquium Stabiliteit van differentieschema's, deel 2*, 1968. ISBN 90 6196 035 5.

MCS 3.1 H.A. LAUWERIER, *Randwaardeproblemen, deel 1*, 1967. ISBN 90 6196 024 X.

MCS 3.2 H.A. LAUWERIER, *Randwaardeproblemen, deel 2*, 1968. ISBN 90 6196 036 3.

MCS 3.3 H.A. LAUWERIER, *Randwaardeproblemen, deel 3*, 1968. ISBN 90 6196 043 6.

MCS 4 H.A. LAUWERIER, *Representaties van groepen*, 1968. ISBN 90 6196 037 1.

MCS 5 J.H. VAN LINT, J.J. SEIDEL & P.C. BAAYEN, *Colloquium Discrete wiskunde*, 1968.
ISBN 90 6196 044 4.

MCS 6 K.K. KOKSMA, *Cursus ALGOL 60*, 1969. ISBN 90 6196 045 2.

MCS 7.1 *Colloquium Moderne rekenmachines, deel 1*, 1969. ISBN 90 6196 046 0.

MCS 7.2 *Colloquium Moderne rekenmachines, deel 2*, 1969. ISBN 90 6196 047 9.

MCS 8 H. BAVINCK & J. GRASMAN, *Relaxatietrillingen*, 1969.
ISBN 90 6196 056 8.

MCS 9.1 T.M.T. COOLEN, G.J.R. FÖRCH, E.M. DE JAGER & H.G.J. PIJLS, *Elliptische differentiaalvergelijkingen, deel 1*, 1970.
ISBN 90 6196 048 7.

MCS 9.2 W.P. VAN DEN BRINK, T.M.T. COOLEN, B. DIJKHUIS, P.P.N. DE GROEN, P.J. VAN DER HOUWEN, E.M. DE JAGER, N.M. TEMME & R.J. DE VOGELAERE, *Colloquium Elliptische differentiaalvergelijkingen, deel 2*, 1970.
ISBN 90 6196 049 5.

MCS 10 J. FABIUS & W.R. VAN ZWET, *Grondbegrippen van de waarschijnlijkheidsrekening*, 1970. ISBN 90 6196 057 6.

MCS 11 H. BART, M.A. KAASHOEK, H.G.J. PIJLS, W.J. DE SCHIPPER & J. DE VRIES, *Colloquium Halfalgebra's en positieve operatoren*, 1971.
ISBN 90 6196 067 3.

MCS 12 T.J. DEKKER, *Numerieke algebra*, 1971. ISBN 90 6196 068 1.

MCS 13 F.E.J. KRUSEMAN ARETZ, *Programmeren voor rekenautomaten; De MC ALGOL 60 vertaler voor de EL X8*, 1971. ISBN 90 6196 069 X.

MCS 14 H. BAVINCK, W. GAUTSCHI & G.M. WILLEMS, *Colloquium Approximatietheorie*, 1971. ISBN 90 6196 070 3.

MCS 15.1 T.J. DEKKER, P.W. HEMKER & P.J. VAN DER HOUWEN, *Colloquium Stijve differentiaalvergelijkingen, deel 1*, 1972. ISBN 90 6196 078 9.

MCS 15.2 P.A. BEENTJES, K. DEKKER, H.C. HEMKER, S.P.N. VAN KAMPEN & G.M. WILLEMS, *Colloquium Stijve differentiaalvergelijkingen, deel 2*, 1973. ISBN 90 6196 079 7.

MCS 15.3 P.A. BEENTJES, K. DEKKER, P.W. HEMKER & M. VAN VELDHUIZEN, *Colloquium Stijve differentiaalvergelijkingen, deel 3*, 1975.
ISBN 90 6196 118 1.

MCS 16.1 L. GEURTS, *Cursus Programmeren, deel 1: De elementen van het programmeren*, 1973. ISBN 90 6196 080 0.

MCS 16.2 L. GEURTS, *Cursus Programmeren, deel 2: De programmeertaal ALGOL 60*, 1973. ISBN 90 6196 087 8.

MCS 17.1 P.S. STOBBE, *Lineaire algebra, deel 1*, 1974. ISBN 90 6196 090 8.

MCS 17.2 P.S. STOBBE, *Lineaire algebra, deel 2*, 1974. ISBN 90 6196 091 6.

MCS 17.3 N.M. TEMME, *Lineaire algebra, deel 3*, 1976. ISBN 90 6196 123 8.

MCS 18 F. VAN DER BLIJ, H. FREUDENTHAL, J.J. DE IONGH, J.J. SEIDEL & A. VAN WIJNGAARDEN, *Een kwart eeuw wiskunde 1946-1971, Syllabus van de Vakantiecursus 1971*, 1974. ISBN 90 6196 092 4.

MCS 19 A. HORDIJK, R. POTHARST & J.Th. RUNNENBURG, *Optimaal stoppen van Markovketens*, 1974. ISBN 90 6196 093 2.

MCS 20 T.M.T. COOLEN, P.W. HEMKER, P.J. VAN DER HOUWEN & E. SLAGT, *ALGOL 60 procedures voor begin- en randwaardeproblemen*, 1976. ISBN 90 6196 094 0.

MCS 21 J.W. DE BAKKER (red.), *Colloquium Programmacorrectheid*, 1975. ISBN 90 6196 103 3.

MCS 22 R. HELMERS, F.H. RUYMGAART, M.C.A. VAN ZUYLEN & J. OOSTERHOFF, *Asymptotische methoden in de toetsingstheorie; Toepassingen van naburigheid*, 1976. ISBN 90 6196 104 1.

MCS 23.1 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 1*, 1976. ISBN 90 6196 105 X.

MCS 23.2 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 2*, 1976. ISBN 90 6196 115 7.

MCS 24.1 P.J. VAN DER HOUWEN, *Numerieke integratie van differentiaalvergelijkingen, deel 1: Eenstapsmethoden*, 1974. ISBN 90 6196 106 8.

MCS 25 *Colloquium Structuur van programmeertalen*, 1976. ISBN 90 6196 116 5.

MCS 26.1 N.M. TEMME (ed.), *Nonlinear analysis, volume 1*, 1976. ISBN 90 6196 117 3.

MCS 26.2 N.M. TEMME (ed.), *Nonlinear analysis, volume 2*, 1976. ISBN 90 6196 121 1.

MCS 27 M. BAKKER, P.W. HEMKER, P.J. VAN DER HOUWEN, S.J. POLAK & M. VAN VELDHUIZEN, *Colloquium Discretiseringsmethoden*, 1976. ISBN 90 6196 124 6.

MCS 28 O. DIEKMANN, N.M. TEMME (EDS), *Nonlinear Diffusion Problems*, 1976. ISBN 90 6196 126 2.

MCS 29.1 J.C.P. BUS (red.), *Colloquium Numerieke programmatuur, deel 1A, deel 1B*, 1976. ISBN 90 6196 128 9.

MCS 29.2 H.J.J. TE RIELE (red.), *Colloquium Numerieke programmatuur, deel 2*, 1976. ISBN 90 6196 144 0

* MCS 30 P. GROENEBOOM, R. HELMERS, J. OOSTERHOFF & R. POTHARST, *Efficiency begrippen in de statistiek*, . ISBN 90 6196 149 1.

MCS 31 J.H. VAN LINT (red.), *Inleiding in de coderingstheorie*, 1976. ISBN 90 6196 136 X.

MCS 32 L. GEURTS (red.), *Colloquium Bedrijfssystemen*, 1976. ISBN 90 6196 137 8.

MCS 33 P.J. VAN DER HOUWEN, *Differentieschema's voor de berekening van waterstanden in zeeën en rivieren*, 1977. ISBN 90 6196 138 6.

MCS 34 J. HEMELRIJK, *Oriënterende cursus mathematische statistiek*, ISBN 90 6196 139 4.

MCS 35 P.J.W. TEN HAGEN (red.), *Colloquium Computer Graphics*, 1977. ISBN 90 6196 142 4.

MCS 36 J.M. AARTS, J. DE VRIES, *Colloquium Topologische Dynamische Systemen*, 1977. ISBN 90 6196 143 2.

MCS 37 J.C. van Vliet (red.), *Colloquium Capita Datastructuren*, 1978. ISBN 90 6196 159 9.

MCS 38.1 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 161 0.

MCS 38.2 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 181 5.

MCS 39 O.J. VRIEZE & G.L. Wanrooij, *Colloquium Stochastische Spelen*, 1978. ISBN 90 6196 167 X.

MCS 40 J. VAN TIEL, *Convexe Analyse*, 1979. ISBN 90 6196 187 4.

MCS 41 H.J.J. TE RIELE (ED.), *Colloquium Numerical Treatment of Integral Equations*, 1979. ISBN 90 6196 189 0.

MCS 42 J.C. VAN VLIET (RED.), *Colloquium Capita Implementatie van Programmeertalen*, 1980. ISBN 90 6196 191 2.

MCS 43 A.M. COHEN & H.A. WILBRINK, *Eindige groepen (Een inleidende cursus)*, 1980. ISBN 90 6196 203 X

MCS 44 J.G. VERWER (ED.), *Numerical solution of partial differential equations*, 1980. ISBN 90 6196 205 6.

MCS 45 P. KLINT (red.), *Colloquium hogere programmeertalen en computerarchitectuur*, 1980. ISBN 90 6196 206 4.

MCS 46.1 P.M.G. APERS (RED.), *Colloquium Databankorganisatie*, 1981. ISBN 90 6196 212 9.

MCS 47.1 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part I: *General information and indices*, 1981. ISBN 90 6196 217 X.

MCS 47.2 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part II: *Elementary procedures, algebraic evaluation*, 1981, ISBN 90 6196 217 X.

MCS 47.3 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part III: *Linear algebra, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.4 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part IV: *Linear algebra, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.5 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part V: *Analytical evaluations, analytical problems, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.6 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VI: *Analytical problems, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.7 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VII: *Special functions and constants, interpolation and approximation*, 1981. ISBN 90 6196 217 X.

De met een * gemerkte uitgaven moeten nog verschijnen.