stichting
mathematisch
centrum

MC

DEPARTMENT OF NUMERICAL MATHEMATICS NOVEMBER

NUMAL, A LIBRARY OF NUMERICAL PROCEDURES IN ALGOL 60

VOLUME 6. FUNCTION EVALUATIONS

2e boerhaavestraat 49 amsterdam

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:= 2.718281828459O5;
```

AUTHOR: D.T.WINTER.

INSTITUTE: MATHEMATICAL CENTRE.

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:

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:= 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<0 "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.

RECEIVED: 740620.

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;

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 7"-4 SEC.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE :

THE FORMULA TAN(X) = SIN(X) / COS(X) IS USED. IF COS(X) = 0 THEN
THE VALUE 1.2"322 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;

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 7"=4 SEC.

LANGUAGE: ALGOL 60.

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))= +0.9999999999990
ARCSIN(SIN(0))= +0.0000000000000
SIN(ARCSIN(0))= +0.0000000000000
SIN(ARCSIN(1))= +1.0000000000000
```

SUBSECTION: ARCCOS.

CALLING SEQUENCE:

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

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

PROCEDURES USED: NONE.

RUNNING TIME : ABOUT 7"-4 SEC.

LANGUAGE: ALGOL 60.

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))= +1.0000000000000
ARCCOS(COS(0))= +0.0000000000000
COS(ARCCOS(0))= +0.0000000000010
COS(ARCCOS(1))= +1.0000000000000
```

SOURCE TEXTS:

```
"CODE" 35120;
"REAL" "PROCEDURE" TAN(X); "VALUE" X; "REAL" X;
"BEGIN" "REAL" U; U:= COS(X);
        TAN:= "IF" U=0 "THEN" 1.2"322 "ELSE" SIN(X)/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:= "IF" U>=1 "THEN" 0 "ELSE" 2*ARCTAN(SQRT((1-U)/(1+U)));
     ARCCOS:= "IF" X>0 "THEN" V "ELSE" 3.14159265358979 - V
"END" ARCCOS;
          "EOP"
```

AUTHOR: P.W.HEMKER.

CONTRIBUTOR: F.GROEN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730921.

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;

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 5"-4 SEC.

LANGUAGE : ALGOL 60.

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) <= 741.6 THEN WE USE THE FORMULA :
SINH(X) = 0.5 * (EXP(X) - EXP(-X)).
IF ABS(X) > 741.6 THEN THE VALUE SIGN(X) * 1.2"322 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;

THE MEANING OF THE FORMAL PARAMETER IS :
X: «ARITHMETIC EXPRESSION»;
ENTRY: THE (REAL) ARGUMENT OF COSH(X).

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 5"-4 SEC.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

THE FORMULA COSH(X) = 0.5 * (EXP(X) + EXP(-X)) IS USED.
IF ABS(X) > 741.6 THEN THE VALUE 1.2"322 IS DELIVERED.
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    +0.0"+000
 -2.00    -2.3"-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;

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

PROCEDURES USED :
SINH = CP35111.

RUNNING TIME : LESS THAN ABOUT 9"=4 SEC.

LANGUAGE : ALGOL 60.

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) <= 50 WE USE THE FORMULA :
TANH(X) = (EXP(2 * X) - 1) / (EXP(2 * X) + 1).
IF ABS(X) > 50 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    -2.4"-015
  +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;

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 7"-4 SEC.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

IF ABS(X) < 0.1 A POWER SERIES IS USED (TAYLOR'S FORMULA).
IF 0.1 <= ABS(X) <= "10 WE USE THE FORMULA :
ARCSINH(X) = LN (X + SQRT (X * X + 1)).
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;

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 5"-4 SEC.

LANGUAGE : ALGOL 60.

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.

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(0.05)));
    OUTPUT(61,"("/,D.14D")",COSH(ARCCOSH(1.05)));
"END"
```

DELIVERS :

```
+0.00999999999958
+0.04999999999999
+1.00000000000000
+1.05000000000000
```

SUBSECTION : ARCTANH.

CALLING SEQUENCE :

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

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

PROCEDURES USED : NONE.

RUNNING TIME : ABOUT 5"-4 SEC.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE :

IF ABS(X) < 0.11 A POWER SERIES IS USED (TAYLOR'S FORMULA).
IF 0.11 <= ABS(X) < 1 WE USE THE FORMULA :
ARCTANH(X) = 0.5 * LN ((1 + X) / (1 - X)).
IF ABS(X) >= 1 THE VALUE SIGN(X) * 1.2"322 IS DELIVERED.
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 +
                0.008333331783 )* Y +
                0.166666666675)* Y +
                1.0            )* X ;
       SINH:= "IF" AX < 0.1 "THEN" X "ELSE"
              X * ( 1.0 + 0.148148148148 15 * X * X )
    "END" "ELSE"
    "IF" AX > 741.6 "THEN" SINH:= SIGN(X) * 1.2"322 "ELSE"
    "BEGIN" AX:= EXP( AX ); SINH:= SIGN(X) * 0.5 * ( AX - 1/AX) "END"
 "END" SINH
        "EOP"

"CODE" 35112;
 "REAL" "PROCEDURE" COSH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X)>741.6 "THEN" COSH:= +1.2"322 "ELSE"
 "BEGIN" X:= EXP( ABS(X) ); COSH:= 0.5 * ( X + 1/X) "END";
        "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 *
     (.333333333333 - Y *
     (.133333333333 - Y *
      .053968253968    )))
  "END" "ELSE" "IF" AX < 0.3 "THEN"
  "BEGIN" "REAL" SH;
     SH:= SINH(X);
     TANH:= SH/SQRT(1+SH*SH)
  "END" "ELSE"
  "IF" AX > 50 "THEN" TANH:= SIGN(X) "ELSE"
  "BEGIN" AX:= EXP(AX); TANH:= SIGN(X)*(AX-1)*(AX+1)/(AX*AX+1) "END";
"END"
        "EOP"
```

```
"CODE" 35114;
 "REAL" "PROCEDURE" ARCSINH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X) < 0.1 "THEN"
 "BEGIN" "REAL" Y; Y:= X*X; ARCSINH:= X* (1+ Y *
      (.1666666666667 +Y*
      (.0750000000000 +Y*
      (.0446428571428 6 +Y*
      (.0303819444444 4 +Y*
      (.0223721590909 1 +Y*
      (.0173527644230 8 ))))))));
 "END" "ELSE"
  ARCSINH:= SIGN(X) * ( "IF" ABS(X) > "10 "THEN"
            0.693147180559 95 + LN( ABS(X)) "ELSE"
            LN(ABS(X) + SQRT( X*X+1)));
         "EOP"

"CODE" 35115;
 "REAL" "PROCEDURE" ARCCOSH(X); "VALUE" X; "REAL" X;
 ARCCOSH:= "IF" X <= 1 "THEN" 0 "ELSE"
           "IF" X > "10 "THEN" 0.693147180559 95 + LN(X) "ELSE"
           LN(X+SQRT((X-1)*(X+1)));
         "EOP"

"CODE" 35116;
 "REAL" "PROCEDURE" ARCTANH(X); "VALUE" X; "REAL" X;
 "IF" ABS(X) < 0.11 "THEN"
 "BEGIN" "REAL" Y; Y:= X*X; ARCTANH:= X*(1+Y*
      (.3333333333333 +Y*
      (.2000000000000 +Y*
      (.1428571428571 4 +Y*
      (.1111111111111 1 +Y*
      (.0909090909090 9 +Y*
      (.0769230769230 8 +Y*
       .0666666666666 7 ))))))))
 "END" "ELSE"
 ARCTANH:= "IF" ABS(X) >= 1 "THEN" SIGN(X) * 1.2"322
           "ELSE" 0.5 * LN((1+X)/(1-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**N 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 :
+5.597359477616"-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(N0,X), WHERE N0=ENTIER(X+.5), (SEE ALSO REF[5]). THE INTEGRALS ARE THEN COMPUTED BY BACKWARD RECURRENCE IF N<N0 AND BY FORWARD RECURRENCE IF N>N0. TO OBTAIN THE STARTING VALUES E(N0,X) OF THE RECURSION THE FOLLOWING CASES ARE DISTINGUISHED:

A) N0 = 1: THE PROCEDURE EI IS USED (SECTION 6.5);
B) N0<=10: A TAYLOR EXPANSION ABOUT X=N0 IS USED, WHICH MADE IT NECESSARY TO STORE THE VALUES OF E(N,N) IN THE PROCEDURE FOR N= 2, 3,...,10;
C) N0 >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,"("48,"("E(")",DD,"(",1.1)=  ")",N/")",I,A[I]);
    NONEXPENX(50.1, 1, 1, B);
    OUTPUT(61,"("/,48,"("EXP(50.1)*E(1,50.1)=  ")",N")",B[1]);
"END"
```

THIS PROGRAM DELIVERS:

```
E(40,1.1)=  +8.295213412863 4"-003
E(41,1.1)=  +8.093658723598 2"-003
E(42,1.1)=  +7.901659978100 6"-003

EXP(50.1)*E(1,50.1)=  +1.957669632472 3"-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]:= 1.9999999924131    ;
        P[1]:=X-3.0000016782085     ;Q[2]:=-2.999964329444446  ;
        P[2]:=X-5.00140345515924    ;Q[3]:=-7.9040492298926    ;
        P[3]:=X-7.4928916779284     ;Q[4]:=-4.31325836146628   ;
        P[4]:=X-3.0833626905176"+1  ;Q[5]:= 2.959993948683"+2  ;
        P[5]:=X-1.393813603644O5    ;Q[6]:=-6.74704580465832   ;
        P[6]:=X+8.912638225737O8    ;Q[7]:= 1.047453626524680"+3;
        P[7]:=X-5.3168662349448"+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]:= 1.000838674026390   ;
        P[1]:=X-1.950223212896600   ;Q[2]:=-3.439422668998700  ;
        P[2]:=X+1.756563154696140   ;Q[3]:= 2.895167279251350"+1;
        P[3]:=X+1.796016887692520"+1;Q[4]:= 7.607611480077350"+2;
        P[4]:=X-3.234673303054030"+1;Q[5]:= 2.577763842384400"+1;
        P[5]:=X-8.285619941406410   ;Q[6]:= 5.728371938373240"+1;
        P[6]:=X-1.865454548833990"+1;Q[7]:= 6.950006558874340"+1;
        P[7]:=X-3.483346536028530   ;
   EI:=EXP(X)*JFRAC(7,Q,P)/X
"END" "ELSE" "IF" X>6 "THEN"
"BEGIN" P[0]:= +1.00443109228078    ;Q[1]:= 5.2746851962908"-1;
        P[1]:=X-4.3253113287813O"+1 ;Q[2]:= 2.7362411988932O"+3;
        P[2]:=X+6.0121799083008O"+1 ;Q[3]:= 1.4325673812193O"+1;
        P[3]:=X-3.3184253199722l"+1 ;Q[4]:= 1.0036743951672O"+3;
        P[4]:=X+2.5076281129356O"+1 ;Q[5]:=-6.2504116167187600  ;
        P[5]:=X+9.3081638566216500  ;Q[6]:= 3.0089264837291O"+2;
        P[6]:=X-2.1901023385488O"+1 ;Q[7]:= 3.9370770185271500  ;
        P[7]:=X-2.1808638152072400  ;
   EI:=EXP(X)*JFRAC(7,Q,P)/X
"END" "ELSE" "IF" X>0 "THEN"
"BEGIN" "REAL" T,R,X0,XMX0;
        P[0]:=-1.9577303690454O"+8;Q[0]:=-8.2627149862605500"+7;
        P[1]:= 3.8928042131120l"+6;Q[1]:= 8.9192576757561200"+7;
        P[2]:=-2.2174462775884500"+7;Q[2]:=-2.4903337574054000"+7;
        P[3]:=-1.1962366934924700"+5;Q[3]:= 4.2855962461174900"+6;
        P[4]:=-2.4930139345864800"+5;Q[4]:=-4.8354743616216400"+5;
        P[5]:=-4.2100161535707000"+3;Q[5]:= 3.5730029805850800"+4;
        P[6]:=-5.4914226552108500"+2;Q[6]:=-1.6070892658722100"+3;
        P[7]:=-8.6693733995107000   ;Q[7]:= 3.4171875000000000"+1;
X0:=.37250741078136700;
T:=X/3-1;
R:=CHEPOLSER(7,T,P)/CHEPOLSER(7,T,Q);
XMX0:=(X-409576229586/109951162777600)-.76717725019939400"-12;
                                                    "COMMENT"
```

```
        "IF" ABS(XMX0)>.037 "THEN" T:=LN(X/X0) "ELSE"
        "BEGIN" "REAL" Z,Z2;
                P[0]:= .8372079339760 75"+1;Q[0]:= .4186039669880 37"+1;
                P[1]:=-.6522687408371 03"+1;Q[1]:=-.4656690260808 14"+1;
                P[2]:= .5699557003067 20    ;Q[2]:= .1"+1;
               Z:=XMX0/(X+X0);Z2:=Z*Z;
               T:=Z*POL(2,Z2,P)/POL(2,Z2,Q)
            "END";
            EI:=T+XMX0*R
        "END" "ELSE"
        "IF" X>=1 "THEN"
        "BEGIN" "REAL" Y;
                P[0]:=-4.4178547172821 7"+4;Q[0]:= 7.6537332333761 4"+4;
                P[1]:= 5.7721724713944 4"+4;Q[1]:= 3.2597188129027 5"+4;
                P[2]:= 9.9383138896203 7"+3;Q[2]:= 6.1061079424575 9"+3;
                P[3]:= 1.8421108866800 0"+3;Q[3]:= 6.3541941837838 2"+2;
                P[4]:= 1.0109380616190 6"+2;Q[4]:= 3.7229835283332 7"+1;
                P[5]:= 5.0341618409756 8    ;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]:= 8.6774595483844 4"-8;Q[0]:= 1;
                P[1]:= 9.9999551930139 0"-1;Q[1]:= 1.2848193537915 7"+1;
                P[2]:= 1.1848310555494 6"+1;Q[2]:= 5.6443356956180 3"+1;
                P[3]:= 4.5593064425339 0"+1;Q[3]:= 1.0664518376991 4"+2;
                P[4]:= 6.9927945129100 3"+1;Q[4]:= 8.9731109712529 0"+1;
                P[5]:= 4.2520203476884 1"+1;Q[5]:= 3.1497184917044 1"+1;
                P[6]:= 8.8367180880384 4    ;Q[6]:= 3.7955900376212 2    ;
                P[7]:= 4.0137766494066 5"-1;Q[7]:= 9.0880456918886 9"-2;
            Y:=-1/X;
            EI:=-EXP(X)*POL(7,Y,P)/POL(7,Y,Q)
        "END" "ELSE"
        "BEGIN" "REAL" Y;
                P[0]:=-9.9999999999844 7"-1;Q[0]:= 1;
                P[1]:=-2.6627106043181 1"+1;Q[1]:= 2.8627106042219 2"+1;
                P[2]:=-2.4105582709701 5"+2;Q[2]:= 2.9231003938853 3"+2;
                P[3]:=-8.9592795777293 7"+2;Q[3]:= 1.3327853774825 7"+3;
                P[4]:=-1.2988568874648 4"+3;Q[4]:= 2.7776194950916 3"+3;
                P[5]:=-5.4537415888313 3"+2;Q[5]:= 2.4040171322590 9"+3;
                P[6]:=-5.6657520653386 9    ;Q[6]:= 6.3165748328080 0"+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]:=.375342618204 91"-1; P[11]:=.135335283236613    ;
            P[ 3]:=.893064655602 28"-2; P[12]:=.497870683678639"-1;
            P[ 4]:=.242339836865 81"-2; P[13]:=.183156388887342"-1;
            P[ 5]:=.705760693424 58"-3; P[14]:=.673794699085 47"-2;
            P[ 6]:=.214802778190 13"-3; P[15]:=.247875217666636"-2;
            P[ 7]:=.673758077810 18"-4; P[16]:=.911881965554516"-3;
            P[ 8]:=.216007301599 75"-4; P[17]:=.335462627902512"-3;
            P[ 9]:=.704115798542 92"-5; P[18]:=.123409804086680"-3;
            P[10]:=.232530265702 82"-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)= +9.460830703671 7"-001     CI(1)= +3.374039229009 7"-001
 F(1)= -2.272552531806 7"-001      G(1)= -9.784015704843 0"-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] :=+2.736870680363O"+00; A[1]:=-1.110631410789 4"+00;
        A[2] :=+1.417656219466 6"-01; A[3]:=-1.025265257917 4"-02;
        A[4] :=+4.649461561988 0"-04; A[5]:=-1.436173089664 2"-05;
        A[6] :=+3.209368494822 9"-07; A[7]:=-5.425199077016 2"-09;
        A[8] :=+7.177628863989 5"-11; A[9]:=-7.633549372348 2"-13;
        A[10]:=+6.667995834698 3"-15;
        Z:= X / 4; Z2:= Z * Z; G:= Z2 +Z2 - 1;
        SI:= Z * CHEPOLSER(10,G,A);
        A[0] :=+2.965960140072 7"+00; A[1]:=-9.429719834183 0"-01;
        A[2] :=+8.611034273816 9"-02; A[3]:=-4.777608454713 9"-03;
        A[4] :=+1.752916120514 6"-04; A[5]:=-4.544872780375 2"-06;
        A[6] :=+8.751583918006 0"-08; A[7]:=-1.299869993810 9"-09;
        A[8] :=+1.533897489883 1"-11; A[9]:=-1.472425607027 7"-13;
        A[10]:=+1.172142079842 9"-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] :=+9.657882035185"-01; A[1] :=-4.306083777859"-02;
        A[2] :=-7.314371174810"-03; A[3] :=+1.470523578986"-03;
        A[4] :=-9.865768573270"-05; A[5] :=-2.274320220465"-05;
        A[6] :=+9.824025732252"-06; A[7] :=-1.897343014871"-06;
        A[8] :=+1.006343594155"-07; A[9] :=+8.081936482224"-08;
        A[10]:=-3.897628287528"-08; A[11]:=+1.033565032549"-08;
        A[12]:=-1.410434487589"-09; A[13]:=-2.523207839968"-10;
        A[14]:=+2.569983132596"-10; A[15]:=-1.059788925394"-10;
        A[16]:=+2.897003157021"-11; A[17]:=-4.102314256308"-12;
        A[18]:=-1.043769373001"-12; A[19]:=+1.099418452054"-12;
        A[20]:=-5.221423940167"-13; A[21]:=+1.746992078783"-13;
        A[22]:=-3.847001297927"-14;
        F:= CHEPOLSER(22, 8/ABSX-1, A) / X;
        A[0] :=+2.280122063824"-01; A[1] :=-2.686972741109"-02;
        A[2] :=-3.510715728095"-03; A[3] :=+1.239800863518"-03;
        A[4] :=-1.567294511686"-04; A[5] :=-1.066414179809"-05;
        A[6] :=+1.117062934357"-05; A[7] :=-3.175401165561"-06;
        A[8] :=+4.431747352039"-07; A[9] :=+5.510869687446"-08;
        A[10]:=-5.924307871174"-08; A[11]:=+2.210257338155"-08;
        A[12]:=-5.025682754062"-09; A[13]:=+3.151916825942"-10;
        A[14]:=+3.630699084897"-10; A[15]:=-2.297476423459"-10;
        A[16]:=+8.553030942404"-11; A[17]:=-2.118306772444"-11;
        A[18]:=+1.713366264509"-12; A[19]:=+1.723887751724"-12;
        A[20]:=-1.293028136681"-12; A[21]:=+5.747233922373"-13;
        A[22]:=-1.841546826831"-13; A[23]:=+3.593725657143"-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 FURMULA 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    -2.633521515996 3"-005
  +,25    +3.625609908221 9"+000
 +1,50    +8.862269254527 6"-001
+22,00    +5.109094217170 9"+019
+50,00    +6.082818640342 2"+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.2D38, N, /")", X, LOG GAMMA(X))
"END"
```

YIELDS THE FOLLOWING RESULTS:

```
  +.25    +1.288022524698l"+000
 +1.50    -1.207822376352l"-001
+12.00    +1.750230784587l"+001
+15.00    +2.519122118273l"+001
+80.00    +2.692910976510l"+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
+2,116608667306"+000  +3,883391332693"+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" NHAX; "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,P0,I0,I1,IQ0,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,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(P); S:= P-M; P0:= "IF" S>0 "THEN" S "ELSE" S+1;
MMAX:= "IF" S>0 "THEN" M "ELSE" M-1;
I0:= INCBETA(X,P0,Q,EPS);
I1:= INCBETA(X,P0,Q+1,EPS);
"BEGIN" "ARRAY" IP[0:MMAX];
    BACKWARD(X,P0,Q,I0,MMAX,EPS,IP); IQ0:= IP[MMAX];
    BACKWARD(X,P0,Q+1,I1,MMAX,EPS,IP); IQ1:= IP[MMAX]
"END";
FORWARD(X,P,Q,IQ0,IQ1,NMAX,I)
"END" IXPFIX;
        "EOP"

"CODE" 35055;
"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;
"BEGIN" "INTEGER" M,N; "REAL" Y,R,S;
    I[0]:= I0; "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" 35056;
"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 = ")", +.5D85D85D"+D, /")",
                0,6, 0,4, INVERF1);
    OUTPUT(61,"(""("     X = ")", +D.D, "(" 1 - X = ")", +D.3D"+2ZD,
                "(" INVERF = ")", +.5D85D85D"+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

$$C(X) = 0.5 + F(X)\sin(\pi/2 \cdot X \cdot X) - G(X)\cos(\pi/2 \cdot X \cdot X)$$

AND

$$S(X) = 0.5 - F(X)\cos(\pi/2 \cdot X \cdot X) - G(X)\sin(\pi/2 \cdot X \cdot X).$$

IF $X < 0$ WE USE THE RELATIONS

$$C(-X) = -C(X),\ S(-X) = -S(X),\ F(-X) = -F(X) \text{ AND } G(-X) = -G(X).$$

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: 750207.

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 FQ1;
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.

LANGUAGE: ALGOL 60.

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 J0.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS :
"REAL" "PROCEDURE" BESS J0(X); "VALUE" X; "REAL" X;

BESS J0 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 PQ0 = 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 J0(X); "CODE" 35160;
    X:= 1; OUTPUT(61,"("/,D,6B-.14D "-ZD")",
    X , BESS J0(X))
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
1          .7651976865794" 0
```

SUBSECTION: BESS J1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" BESS J1(X); "VALUE" X; "REAL" X;

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"
```

PRINTS THE FOLLOWING RESULTS:

```
1     .44005058574492" 0
```

SUBSECTION: BESS J.

CALLING SEQJENCE:

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

THE MEANING OF THE FORMAL PARAMETERS IS:
X:   <ARITHMETIC EXPRESSION>;
     THE ARGUMENT OF THE BESSEL FUNCTIONS;
N:   <ARITHMETIC EXPRESSION>;
     THE JPPER BOUND OF THE INDICES OF ARRAY J;
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: NONE.

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 JSE:

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"
```

PRINTS 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,Y0,Y1); "VALUE" X; "REAL" X,Y0,Y1;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTIONS;
Y0: <VARIABLE>;
EXIT: Y0 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,Y0,Y1;
    "PROCEDURE" BESS Y01(X,Y0,Y1); "CODE" 35163;
    X:= 1; BESS Y01(X,Y0,Y1);
    OUTPUT(61,"("/,4B0.D,2(B-.14D"-ZD)")",X,Y0,Y1)
"END"
```

PRINTS THE FOLLOWING RESULTS:

```
1.0    9882569642156 76"  -1    -.781212821300 28" 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;

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;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X > 0;
P0: <VARIABLE>;
EXIT: P0 CORRESPONDS TO THE FUNCTION P(X,0) DEFINED
IN [1,FORMULAS 9.2.5 AND 9.2.6];
Q0: <VARIABLE>;
EXIT: Q0 CORRESPONDS TO 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;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT; THIS ARGUMENT SHOULD SATISFY X > 0;
P1: <VARIABLE>;
EXIT: P1 CORRESPONDS TO THE FUNCTION P(X,1) DEFINED
IN [1,FORMULAS 9.2.5 AND 9.2.6];
Q1: <VARIABLE>;
EXIT: Q1 CORRESPONDS TO 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
P0 * P1 + Q0 * Q1 = 1, WHATEVER X. IN THE FOLLOWING PROGRAM WE
VERIFY THIS RELATION.

```
"BEGIN" "REAL" X,P,Q,R,S;
    "PROCEDURE" BESS PQ0(X,P0,Q0); "CODE" 35165;
    "PROCEDURE" BESS PQ1(X,P1,Q1); "CODE" 35166;
    "FOR" X:= 1,3,5,10 "DO"
    "BEGIN" BESSPQ0(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 J0(X); "VALUE" X; "REAL" X;
"IF" X=0 "THEN" BESS J0:= 1 "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;
    B1:= B2:= 0;
    "FOR" AR:=-.75885"-15, +.4125321          "-13,
    -.194383469       "-11, +.7848696314      "-10,
    -.267925353056    "- 8, +.760816359241 9  "- 7,
    -.176194690776215"- 5, +.324603288210051"- 4,
    -.46062616620628 "- 3, +.48191800694676 "- 2,
    -.34893769411409 "- 1, +.158067102332097    ,
    -.37009499387265 "- 0, +.265178613203337    ,
    -.872344235285222"- 2 "DO"
    "BEGIN" B0:= Z2*B1-B2+AR;
      B2:= B1; B1:= B0
    "END";
  BESS J0:= Z*B1 - B2 + .15772 79714 7489
"END" "ELSE"
    "BEGIN" "REAL" C, COSX, SINX, P0, Q0;
      "PROCEDURE" BESS PQ0(X, P0, Q0); "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 PQ0(X, P0, Q0);
      BESSJ0:= C * (P0 * COSX - Q0 * SINX)
    "END" BESS J0;
        "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, +.2949707007278    "- 7,
-.76175878054003 "- 6, +.158870192399321"- 4,
-.26044389348581"- 3, +.324027018268386"- 2,
-.29175524806 1542"- 1, +.177709117239728"- 0,
-.66144393413 4543"- 0, +.128799409885768"+ 1,
-.11918011605 4122"+ 1 "DO"
"BEGIN" B0:= Z2*B1-B2+AR;
  B2:= B1; B1:= B0
"END";
```

"COMMENT"

```
      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"
    "BEGIN""REAL" X2, R, S; "INTEGER" L, M, NU, SIGNX;
      SIGNX:= SIGN(X); X:= ABS(X);
      R:="IF" N=0 "THEN" 0 "ELSE" (35.9253+N)/(1+LN(1+17.96265/N));
      S:= (52.8648+X)/(1+LN(1+26.4324/X))*1.3591;
      NU:= (6 + ENTIER("IF" R < S "THEN" S "ELSE" R))//2 * 2;
      R:= S:= 0; X2:= 2/X; L:= 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 Y01(X, Y0, Y1); "VALUE" X; "REAL" X, Y0, Y1;
    "IF" X <= 0 "THEN" Y0:= Y1:= -"+300 "ELSE"
    "IF" X< 8 "THEN"
"BEGIN" "REAL" Z, Z2, C, LNX, AR, B0, B1, B2;
  "REAL" "PROCEDURE" J0(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 Y0;
    B1:= B2:= 0;
    "FOR" AR:= +.164349 "-14,
    -.8747341          "-13, +.402633082       "-11,
    -.15837552542      "- 9, +.524879478733    "- 8,
    -.144072332740l9   "- 6, +.32065325376548  "- 5,
    -.563207914105699"- 4, +.753113593257774"- 3,
    -.72879624795521 "- 2, +.471966895957634"- 1,
    -.177302012781143"- 0, +.261567346255047"- 0,
    +.179034314077182"- 0, -.274474305529745"DO"
    "BEGIN" B0:= Z2*B1-B2+AR;
      B2:= B1; B1:= B0
    "END";
  Y0:= LNX * J0(8*X)+Z*B1-B2-.33146 11320 3285"-1;
    "COMMENT" COMPUTATION OF Y1;
    B1:= B2:= 0;
    "FOR" AR:=
    +.42773            "-15, -.2440949         "-13,
    +.121143321        "-11, -.5172121473      "-10,
    +.187547032473     "- 8, -.5688440039919   "- 7,
    +.1416624364492 35"- 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;
     "IF" X <= 0 "THEN"
      "BEGIN" "FOR" N:=N "STEP" -1 "UNTIL" 0 "DO" Y[N]:=-"+300
     "END" "ELSE"
     "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 > 1 "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" ABS(X) < 8 "THEN"
     "BEGIN" "REAL" B, COSX, SINX, J0X, Y0;
       "REAL" "PROCEDURE" J0(X); "CODE" 35160;
       "PROCEDURE" BESS Y01(X, Y0, Y1); "CODE" 35163;
       X:= ABS(X); 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,
           -.5170594537606      "- 7, +.30751847875195     "- 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,
       -.3206747 42         "-11, +.4220121905         "-10,
       -.72719159369        "- 9, +.1797245724797      "- 7,
       -.741449841 10606    "- 6, +.68385199426 1165   "- 4
       "DO"
       "BEGIN" B0:= X2 * B1 - B2 + AR;
                                                  "COMMENT"
```

```
        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" ABS(X) < 8 "THEN"
    "BEGIN" "REAL" B, COSX, SINX, J1X, Y1;
      "REAL" "PROCEDURE" J1(X); "CODE" 35161;
      "PROCEDURE" BESS Y01(X, Y0, Y1); "CODE" 35163;
      X:=ABS(X); 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, +.617763396064    "- 7,
    -.398728430048895"- 5, +.898989833085940 "- 3
    "DO"
    "BEGIN" B0:= B1 * X2 - B2 + AR;
        B2:= B1; B1:= B0
    "END";
    P1:= X * B1 - B2 + 1.000903040860013 7;
    "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, +.913861525795555 "- 6,
    -.962772354915 71"- 4 "DO"
    "BEGIN" B0:= X2 * B1 - B2 + AR;
        B2:= B1; B1:= B0
    "END";
    Q1:=(X * B1 - B2 + .467777870695 35" -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.

LANGUAGE: ALGOL 60.

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 SEQJENCE:

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

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 FUNCTIJN.

PROCEDURE S 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 REQJIRED;
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: SEE [6].

EXAMPLE OF JSE:

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       .1266065877752O"  1
```

SUBSECTION: BESS I1.

CALLING SEQUENCE:

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

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: 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       .565159103992520"   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;

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:

AT FIRST THE RATIOS OF ANY TWO SUBSEQUENT ARRAY ELEMENTS ARE
COMPUTED, WHILE THE ZEROETH ELEMENT IS CALCULATED
AS A CONTINUED FRACTION; HENCE ALL THE ARRAY ELEMENTS
ARE COMPUTED; SEE [5].

PROCEDURES USED :

NONEXP BESS I = CP 35177.

REQUIRED CENTRAL MEMORY:

NO AUXILIARY ARRAYS ARE USED.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO THE MAXIMJM 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;

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   .9244190712276 6"   0   .1656441120003 3"   1
1.5   .2138055626475 4"   0   .2773878004568 3"   0
2.5   .6234755320036 6" -1   .7389081634774 6" -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;

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;
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: K0 = CP35173.

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 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(B8.12D"-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;

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       .4657596075936 4"   0
```

SUBSECTION: NONEXP BESS I1.

CALLING SEQUENCE:

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

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 JSE:

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"
```

PRINTS THE FOLLOWING RESULTS:

```
1       .207910415349 72"   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;
```

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 (L=0,...,N) MULTIPLIED BY EXP (- X).

PROCEDURES USED: NONE.

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;

THE MEANING OF THE FORMAL PARAMETERS IS:
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT OF THE BESSEL FUNCTIONS;
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     .152410938577 39"  1     .273100970821 18"  1
1.0     .114463079806 9"   1     .163615348626 33"  1
1.5     .958210053294 96"  0     .124316587355 25"  1
2.0     .841568215070 78"  0     .103347684706 87"  1
2.5     .759548690328 10"  0     .900174423907 88"  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;

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   .152410938577 39"1   .273100970821 18"1   .124481486218 62"2
1.0   .114446307980 69"1   .163615348626 33"1   .441677005233 34"1
1.5   .958210053294 96"0   .124316587355 25"1   .261576455136 49"1
2.0   .841568215070 78"0   .103347684706 87"0   .187504506213 95"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 K01(X, K0, K1); "VALUE" X; "REAL" X, K0, K1;
     "IF"X <= 0 "THEN" K0:= K1:= "+ 300 "ELSE"
     "IF" X <= 1.5 "THEN"
     "BEGIN" "INTEGER" K; "REAL" C, D, R, S, SUM0, SUM1, T,
     TERM, T0, T1;
     SUM0:= D:= LN(2/X) -.5772156649015328606;
     SUM1:=    C:= -1 -2    * D; R:= TERM:= 1; T:= X * X/4;
     "FOR" K:= 1,K+1 "WHILE" ABS(T0/SUM0) + ABS(T1/SUM1) >
     "-15 "DO"
     "BEGIN"  TERM:= T * TERM * R * R; D:= D + R;
          C:= C - R; R:= 1/(K+1); C:= C - R;
          T0:= TERM * D; T1:= TERM * C * R;
          SUM0:= SUM0 + T0; SUM1:= SUM1 + T1
     "END";
     K0:= SUM0; K1:= (1 + T * SUM1) / X
"END" "ELSE"
"BEGIN" "REAL" EXPX;
     "PROCEDURE" NONEXP BESS K01(X, K0, K1); "CODE" 35178;
     EXPX:= EXP(- X);
     NONEXP BESS K01(X, K0, K1); K1:= EXPX * K1; K0:= K0 * EXPX
"END" BESS K01;
          "EOP"

"CODE" 35174;
     "PROCEDURE" BESS K(X, N, K); "VALUE" X, N;
     "REAL" X; "INTEGER" N; "ARRAY" K;
     "IF" X <= 0 "THEN"
     "BEGIN" "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" K[N]:= +"+300 "END"
     "ELSE"
     "BEGIN"  "INTEGER" I; "REAL" K0, K1, K2;
          "PROCEDURE" BESS K01(X, K0, K1); "CODE" 35173;
          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" BESS K;
          "EOP"

"CODE" 35175;
     "REAL" "PROCEDURE" NONEXP BESS I0(X);
     "VALUE" X; "REAL" X;
     "IF" X= 0 "THEN"
     NONEXP BESS I0:=1 "ELSE"
     "IF" ABS(X) <= 15.0 "THEN"
     "BEGIN" "REAL" "PROCEDURE" BESS I0(X); "CODE" 35170;
       NONEXP BESS I0:= EXP(-ABS(X))*BESS I0(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;
     NEGATIVE:= X < 0; X:= ABS(X);
     R:= 1.3591 * ("IF" X > 35.9253 "THEN" X "ELSE"
     (52.8648-.4715 * X) /(1+LN(.26424+26.4324/X)));
     X2:= 2 / X; S:= "IF" N = 0 "THEN" 0 "ELSE"
     (N + 36) / (1 + LN(1 + 18 / N));
     K:= ENTIER("IF" R < S "THEN" S "ELSE" R); R:= S:= 0;
     "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 <= 0 "THEN" K0:= K1:= " + 300 "ELSE"
     "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:= .904837418O3596,
     .670320046O3564,      .4065696597406O,      .2018965179946 6,
     .820849986238 99"-1,  .2732372244729 3"-1,  .7446583070924 3"-2,
     .1661557273173 9"-2,  .3035391380788 7"-3,  .4539992976248 5"-4,
     .5595132416500"-5,    .5573903692694 4"-6,  .4575338769445 9"-7,
     .3074879879586 50"-8, .1691897922615 1"-9,  .7621865194512 7"-11,
     .2811185298789 1"-12, .8489044033872 9"-14, .2098791048793"-15,
     .4248354255291 6"-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, .63257510850049 " - 6,
    -.111066851966665" - 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;
    "IF" X <= 0 "THEN"
    "BEGIN" "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" K[N]:= +"+300 "END"
    "ELSE"
    "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.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 741221.

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.

BESS YA01:

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

KEYWORDS:

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

LANGUAGE: ALGOL 60.

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].LUKE, Y.L., MATH. COMP. 26 (1972), 237-240.

[3].TEMME, N.M. ON THE NUMERICAL EVALUATION OF THE
MODIFIED BESSEL FUNCTION OF THE THIRD KIND.
REPORT TN 72/73, MATHEMATICAL CENTRE, AMSTERDAM, 1973.

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

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

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;

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 VALJE 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.

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:

IN ALL THE CASES THE BESSEL FUNCTIONS ARE COMPUTED ACCORDING THE MILLER METHOD DESCRIBED IN [5, P.46-52].

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
.5730612692836 4"0   .4152947512442 4" 0   .1661633879311 1" 0
```

SUBSECTION: BESS YA01.

CALLING SEQJENCE:

THE HEADING OF THE PROCEDURE READS:

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

THE MEANING OF THE FORMAL PARAMETERS IS:

```
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER. THIS PARAMETER SHOULD SATISFY A>=0;
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]. APART FROM SOME MINOR MODIFICATIONS, THE METHOD IS THE SAME FOR THE NEUMANN FUNCTIONS.FOR X>=3 THE PROCEDURE CALLS FOR THE PROCEDURE BESS PQA01 (SEE SUBSECTION BESS PQA01).IT IS STRONGLY ADVISED TO TAKE A>=0. IF THE PROCEDURE IS CALLED WITH NEGATIVE A THE COMPUTED FUNCTIONS HAVE THE ORDER ABS(A) AND ABS(A)+1 RESPECTIVELY. 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

```
+8.8256964215680"-002  -7.8121282130031"-001.
```

SUBSECTION: BESS YAPLUSN.

CALLING SEQJENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDJRE" BESS YAPLUSN(A, X, NMAX, YAN); "VALUE" A, X, NMAX;
"REAL" A, X; "INTEGER" NMAX; "ARRAY" YAN;

THE MEANING OF THE FORMAL PARAMETERS IS:
A:      <ARITHMETIC EXPRESSION>;
        THE ORDER; THIS PARAMETER SHOULD SATISFY 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 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.

EXAMPLE OF JSE:

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

```
+8.8256964215680"-002 -7.8121282130031"-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;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARITHMETIC EXPRESSION>;
THE ORDER. THIS PARAMETER SHOULD SATISFY A-ENTIER(A)^=.5;
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 COMPUTATIONS ARE BASED ON CHEBYSHEV EXPANSIONS. SEE REFERENCE [2]; IN FORMULA (12) OF THAT PAPER WE TAKE LAMBDA EQUAL TO Z. THEORETICALLY, THE ALGORITHM CONVERGES FOR ALL X>0, BUT FOR SMALL VALUES OF X THE CONVERGENCE IS POOR. 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:= 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:

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

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"
"BEGIN" "REAL" X2, R, S, L, LABDA; "INTEGER" M, NU;
  "REAL" "PROCEDURE" GAMMA(Y); "CODE" 35061;
  R:= "IF" N = 0 "THEN" 0 "ELSE" (35.9253 + N) /
  (1 + LN(1 + 17.96265 / N)); X:= ABS(X);
  S:= (52.8548 + X) / (1 + LN(1 + 26.4324 / X)) * 1.3591; L:= 1;
  NU:=    (6 + ENTIER("IF" R <= S "THEN" S "ELSE" R)) // 2 ;
  "IF" A ^= 1 "THEN"
  "FOR" M:= 1 "STEP" 1 "UNTIL" NU "DO"
  L:= L * (M + A) / (M + 1); R:= S:= 0; X2:= 2 / X;
  "FOR" M:= NU+NU "STEP" - 1 "UNTIL" 1 "DO"
  "BEGIN" R:= 1 / (X2 * (A + M)    - R);
    "IF" M // 2 ^= M / 2 "THEN" LABDA:= 0 "ELSE"
    "BEGIN" "IF" A ^= 1 "THEN" L:= L * (M + 2) / (M + A + A);
      LABDA:= L * (M + A) "END";
    S:= R * (LABDA + S);
    "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;
  "BEGIN" "REAL" A2, B, C, D, E, F, G, H, P, Q, S0, S1, PI, T;
    "INTEGER" N, NA;
"REAL" "PROCEDURE" GAM(X, ODD, EVEN);                    "CODE" 35060;
"REAL" "PROCEDURE" SINH(X);                              "CODE" 35111;
"PROCEDURE" BESS PQA01(A, X, PA, QA, PA1, QA1); "CODE" 35183;
A:= ABS(A); NA:= ENTIER(A); A:= A-NA;
"IF" A > .5 "THEN" A:= A - 1; A2:= A/2; PI:= 3.1415 92653 58979;
"IF" A = .5 "THEN"
"BEGIN"
    C:= COS(X); F:= -SQRT(2/(X*PI))*C; G:= F*(1/X+SIN(X)/C)
"END" "ELSE"
"BEGIN" "IF" X < 3 "THEN"
    "BEGIN" B:= X/2; D:= -LN(B); E:= A*D; C:= A*PI;
        T:= 2*PI*SIN(C/2)/COS(C/2);
        C :="IF" ABS(C) < "-15 "THEN" 1 "ELSE" C/SIN(C);
        S0:= "IF" ABS(E) < "-15 "THEN" 1 "ELSE" SINH(E)/E;
        E:= EXP(E); S1:= (E+1/E)/2; G:= -GAM(A, P, Q)*E;
        F:= C*(P*S1+Q*S0*D); E:= A*A; D:= -B*B; P:= .25*G*C;
        Q:= .25/G; T:= -T*Q/C; C:= 1; S0:= F+T; S1:= P+Q+T*A2;
        "FOR" N:= 1, N+1 "WHILE"
        ABS(H/S0) > "-16 "OR" ABS(G/S1) > "-16 "DO"
        "BEGIN" C:= C*D/N; F:= (F*N-2*(P+Q))/(N*N-E);
            P:= P/(N-A); Q:= Q/(N+A); T:= T/(N+A); H:= C*(F+T);
            G:= C*(N*(F+T)+P+Q+T*A2); S0:= S0+H; S1:= S1+G
        "END";
        F:= -2*S0/PI; G:= (A2*F+2*S1/PI)/B
    "END" "ELSE"
    "BEGIN" B:= X-PI*(A2+.25); C:= SQRT(2/(X*PI));
        D:= COS(B); E:= SIN(B);
        BESS PQA01(A, X, P, Q, S0, S1);
        F:= C*(P*E+Q*D);  G:= C*(S1*E-S0*D)
    "END"
"END";
X:= 2/X; NA:= NA + ("IF" A<0 "THEN" 2 "ELSE" 1); A:= A - 1;
"FOR" N:= 2 "STEP" 1 "UNTIL" NA "DO"
"BEGIN" H:= -F+(A+N)*X*G; F:= G; G:= H "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" X, A, PA, QA, PA1, QA1;
"IF" ABS(X) >= 3 "THEN"
"BEGIN" "REAL" CK, EK, RK, RK1, RK2, RK3, RKD, RKD1, RKJ2, IK,
  IK1, IK2, IK3, IKD, IKD1, IKD2, P0, P1, Q0, Q1, A1, A2, A5, A6,
  RS, IS, B, C, NK, S;
  "INTEGER" K, M, NU;
  X:= AES(X);
  B:= A*A; IK1:= RK1:= 1; RS:= IS:= RKD1:= IKD1:=
  P0:= P1:= Q0:= Q1:= RK2:= RK3:= RKD2:= IK2:= IK3:= IKJ2:= 0;
  A1:= .5-A; A2:= .5+A; A5:= 2.5-A; A6:= 2.5+A;
  NU:= ENTIER(5+35/SQRT(X));
  S:= "IF" NU//2*2=NU "THEN" -1 "ELSE" 1; C:= 8*X;
  "FOR" K:= NU "STEP" -1 "UNTIL" 0 "DO"
  "BEGIN" CK:= -(B+5.75+3*K*(K+3)); M:= K+1;
      EK:= -(K+A5)*(K+A6); NK:= 1/((K+A1)*(K+A2));
      RK:= (M*((CK*RK1+EK*RK3)/(K+2)+C*(IK2-IK1))+CK*RK2)*NK;
      IK:= (M*((CK*IK1+EK*IK3)/(K+2)-C*(RK2-RK1))+CK*IK2)*NK;
      RKD:= RKD2+2*M*RK1; IKD:= IKD2+2*M*IK1;
      P0:= P0+RK; P1:= P1+RKD; Q0:= Q0+IK; Q1:= Q1+IKD;
      "IF" K > 0 "THEN"
      "BEGIN"
          RK3:= RK2; RK2:= RK1; RK1:= RK; RKD2:= RKD1; RKD1:= RKD;
          IK3:= IK2; IK2:= IK1; IK1:= IK; IKD2:= IKD1; IKD1:= IKD
      "END" "ELSE" S:= S/2;
      S:= -S; RS:= RS+S*RK; IS:= IS+S*IK
  "END";
  B:= RS*RS+IS*IS; RS:= RS/B; IS:= IS/B;
  P0:= P0-RK/2; P1:= P1-RKD/2; Q0:= Q0-IK/2; Q1:= Q1-IKD/2;
  PA:= P0*RS+Q0*IS; QA:= Q0*RS-P0*IS;
  P0:= P1*RS+Q1*IS; Q0:= Q1*RS-P1*IS;
  PA1:= PA-(2*Q0+(A+.5)*QA)/X; QA1:= QA+(2*P0+(A+.5)*PA)/X
"END" "ELSE"
"BEGIN" "REAL" PI, C, S, B, CHI, YA, YA1; "ARRAY" JA[0:1];
  "PROCEDURE" BESSJAPLUSN(A, X, N, JA); "CODE" 35180;
  "PROCEDURE" BESSYA01(A, X, YA, YA1); "CODE" 35181;
  X:= ABS(X);
  PI:= 3.14159 26535 8979; B:= SQRT(PI * X / 2);
  CHI:= X - PI * (A / 2 + .25); C:= COS(CHI); S:= SIN(CHI);
  BESSYA01(A, X, YA, YA1); BESSJAPLUSN(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" BESS PQA01;
      "EOP"
```

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

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 741221.

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, A> = 0 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, A > = 0 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, A>=0, 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, A>=0, AND ARGUMENT X>0.
THE MODIFIED BESSEL FUNCTIONS CORRESPOND TO THE FJNCTION 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, A >= 0, 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, A>=0, AND ARGUMENT X>0 MULTIPLIED BY THE FACTOR EXP(X). THJS, APART FROM THE EXPONENTIAL FACTOR, THE FJNCTIONS ARE THE SAME AS THOSE COMPUTED BY THE PROCEDURE BESS KAPLUSN.

KEYWORDS:

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

LANGUAGE: ALGOL 60.

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].LUKE, Y.L., MATH. COMP. 25 (1971), 323-330.

[3].TEMME, N.M., ON THE NUMERICAL EVALUATION OF THE
MODIFIED BESSEL FUNCTION OF THE THIRD KIND.
REPORT TN 72/73, MATHEMATICAL CENTRE, AMSTERDAM, (1973).

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

SUBSECTION: BESS IAPLUSN.

CALLING SEQJENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" BESS IAPLUSN(A, X, N, IA);
"VALUE" X, N, A; "REAL" X, A;
"INTEGER" N; "ARRAY" IA;

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 35191.
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:

```
.25    2.00
.4875922316231"  1    .2965793890724"  1    .1168679868217 7"  1
```

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;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

A: < ARITHMETIC EXPRESSION >;
THE NONINTEGER PART OF THE ORDER A+N;

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

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

REQUIRED CENTRAL MEMORY: NO AUXILIARY ARRAYS ARE DECLARED.

METHOD AND PERFORMANCE:
IN ALL THE CASES THE BESSEL FUNCTIONS ARE COMPUTED ACCORDING THE MILLER METHOD DESCRIBED IN [4, P.46-52].

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" NE BESS IAPLUSN(A, X, N, IA); "CODE" 35193;
    A:= .25; X:= 2; NE 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:

```
.25    2.00
.659884316256 4"  0    .401376556222 60"  0    .158163620978 1"  0
```

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;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARITHMETIC EXPRESSION>;
THE ORDER. THIS PARAMETER SHOULD SATISFY A>=0;

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<1.5 THE BESSEL FUNCTIONS ARE COMPUTED BY USING TAYLOR SERIES. THE METHOD IS DESCRIBED IN REFERENCE [3]. FOR X>=1.5 THE PROCEDURE CALLS FOR THE PROCEDURE NONEXP BESS KA (SEE SUBSECTION NONEXP BESS KA). IT IS STRONGLY ADVISED TO TAKE A>=0. THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND IS AN EVEN FUNCTION WITH RESPECT TO ITS ORDER. IF THE PROCEDURE IS CALLED WITH A<0, THE VALUE OF BESS KA01 ITSELF IS CORRECT, BUT THE OTHER FUNCTION WILL BE OF ORDER 1+ABS(A). 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

```
+4.210244382407 1"-001   +6.019072301972 4"-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;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARITHMETIC EXPRESSION>;
THE ORDER. THIS PARAMETER SHOULD SATISFY 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. THE RECURRENCE RELATION 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

```
+4.210244382407 1"-001 +6.019072301972 4"-001 +1.624838898635 2"+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;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARITHMETIC EXPRESSION>;
THE ORDER. THIS PARAMETER SHOULD SATISFY A>=0;
X: <ARITHMETIC EXPRESSION>;
THE ARGUMENT. THIS ARGUMENT SHOJLD 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<1.5 THE PROCEDURE NONEXP BESS KA CALLS FOR THE PROCEDURE BESS KA01. FOR X>=1.5 THE BESSEL FUNCTIONS ARE COMPUTED FROM THEIR EXPANSION IN CHEBYSHEV POLYNOMIALS. THE METHOD IS DESCRIBED IN REFERENCE [2]. IN FORMULA (1) OF THAT PAPER WE TAKE LAMBDA EQUAL TO Z. 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. IT IS STRONGLY ADVISED TO TAKE A>=0. THE MODIFIED BESSEL FUNCTION OF THE THIRD KIND IS AN EVEN FUNCTION WITH RESPECT TO ITS ORDER. IF THE PROCEDURE IS CALLED WITH A<0, THE VALUE OF NONEXP BESS KA ITSELF IS CORRECT, BUT THE OTHER FUNCTION WILL BE OF ORDER 1+ABS(A).

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:

```
8.4156821507076"-001 +1.0334768470687"+000.
```

SUBSECTION: NONEXP BESS KAPLUSN.

CALLING SEQJENCE:

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;
```

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. THIS PARAMETER SHOULD SATISFY 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.
RECURRENCE RELATION 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 JSE:

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:

```
+5.478075643135O"-001 +6.002738587882 9"-001 +7.879171078288 2"-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" X = 0 "THEN" KA:= KA1:= + "+ 300 "ELSE"
"IF" ABS(X) >= 1.5 "OR" A - ENTIER(A) = .5 "THEN"
"BEGIN" "REAL" EXPON;
    "PROCEDURE" NONEXP BESS KA(A, X, KA, KA1); "CODE" 35194;
    EXPON:= EXP(-X); NONEXP BESS KA  (A, X, KA, KA1);
    KA:= EXPON*KA; KA1:= EXPON*KA1
"END" "ELSE"
"BEGIN" "REAL" B, C, D, E, EPS, F, G, H, P, Q, S0, S1, PI;
    "INTEGER" N, NA;
  "REAL" "PROCEDURE" GAM(X, ODD, EVEN); "VALUE" X;
    "REAL" X, ODD, EVEN; "CODE" 35060;
    "REAL" "PROCEDURE" SINH(X);  "CODE" 35111;
    A:= ABS(A); NA:= ENTIER(A); A:= A-NA; "IF" A>.5 "THEN" A:= A-1;
    PI:= 3.1415 92653 58979; EPS:= "-16;
    B:= X/2; D:= -LN(B); E:= A*D; C:= A*PI;
    C := "IF" ABS(C) < "-15 "THEN" 1 "ELSE" C/SIN(C);
    S0:= "IF" ABS(E) < "-15 "THEN" 1 "ELSE" SINH(E)/E;
    E:= EXP(E); S1:= (E+1/E)/2; G:= -GAM(A, P, Q)*E;
    S0:= F:= C*(P*S1+Q*S0*D); E:= A*A;
    P:= .25*G*C; Q:= .25/G; S1:= P+Q; C:= 1; D:= B*B;
    "FOR" N:= 1, N+1 "WHILE" ABS(H/S0)>EPS"OR"ABS(G/S1)>EPS "DO"
    "BEGIN" C:= C*D/N; F:= (F*N-2*(P+Q))/(N*N-E);
        P:= P/(N-A); Q:= Q/(N+A); G:= C*(N*F+P+Q);
        H:= C*F; S0:= S0+H; S1:= S1+G
    "END";
    F:= S0; G:= (A*F/2-S1)/B;
    X:= 2/X; NA:= NA + ("IF" A<0 "THEN" 2 "ELSE" 1); A:= A - 1;
    "FOR" N:= 2 "STEP" 1 "UNTIL" NA "DO"
    "BEGIN" H:= F+(A+N)*X*G; F:= G; G:= H "END";
    KA:= F; KA1:= G
  "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;
  BESSKA01(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 FRACTIOAL 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;
  A2:= A+A; X:= ABS(X); X2:= 2/X;
  S:= 1.3591 * ("IF" X > 35.9253 "THEN" X "ELSE"
  (52.8648 - .47152 * X)/(1+LN(.26424+26.4324/X)));
  R:= "IF" N= 0 "THEN" 0 "ELSE"
  (35.9253 + N)/(1 + LN (1 + 17.96265/N));
  NU:= 6 + ENTIER ("IF" R < S "THEN" S "ELSE" R); L:= 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" ABS(X) < 1.5 "AND" A = ENTIER(A) "NOTEQUAL" .5 "THEN"
"BEGIN" "REAL" EXPON;
    "PROCEDURE" BESS KA01(A, X, KA, KA1);"CODE" 35191;
    EXPON:= EXP(X); BESSKA01(A, X, KA, KA1);
    KA:= E XPON*KA; KA1:= EXPON*KA1
"END" "ELSE"
"BEGIN" "REAL" A1, A2, A5, A6, B, C, CN, CN1, CN2, CN3, CND, CND1,
    F, G, CND2, H, PI, S, S0, S1, S2; "INTEGER" M, N, NA, NU;
    A:= ABS(A); NA:= ENTIER(A); A:= A-NA; PI:= 3.1415 92653 58979;
    "IF" A > .5 "THEN" A:= A-1; "IF" A=.5 "THEN"
    "BEGIN" F:= SQRT(PI/(X+X)); G:= F*(1+1/X); "GOTO" HALF "END";
    H:= 8*X; G:= A*A; CN2:= CN3:= CND1:= CND2:= S0:= S1:= S2:= 0;
    A1:= .5-A; A2:= .5+A; A5:= 2.5-A; A6:= 2.5+A; CN1:= 1;
    NU:= ENTIER(5+30/SQRT(X));
    S:= "IF" NU//2*2=NU "THEN" -1 "ELSE" 1;
    "FOR" N:= NU "STEP" -1 "UNTIL" 0 "DO"
    "BEGIN" C:= -(G+5.75+3*N*(N+3)); M:= N+1;
        CN:= (M*((C*CN1-(N+A5)*(N+A6)*CN3)/(N+2)+
        H*(CN2-CN1))+C*CN2)/((N+A1)*(N+A2));
        CND:= CND2+2*M*CN1; S1:= S1+CN; S2:= S2+CND;
        "IF" N > 0 "THEN"
        "BEGIN" CN3:= CN2; CN2:= CN1; CN1:= CN;
            CND2:= CND1; CND1:= CND
        "END" "ELSE" S:= S/2;
        S:= -S; S0:= S0+S*CN
    "END";
    S1:= S1-CN/2; S2:= S2-CND/2; H:= SQRT(PI/(X+X))/S0;
    F:= H*S1; G:= H*((A+X+.5)*S1+2*S2)/X;
HALF:   X:= 2/X; NA:= NA + ("IF" A<0 "THEN" 2 "ELSE" 1);
    A:= A - 1;
    "FOR" N:= 2 "STEP" 1 "UNTIL" NA "DO"
    "BEGIN" H:= F+(A+N)*X*G; F:= G; G:= H "END";
    KA:= F; KA1:= G
"END" NONEXP BESS KA;
    "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; OF COURSE 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 FIST 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.

LANGUAGE: ALGOL 60.

REFERENCES:

[1] M. ABRAMOWITZ AND I. STEGUN,
HANDBOOK OF MATHEMATICAL FUNCTIONS.

[2] W. GAUTSCHI,
COMPUTATIONAL ASPECTS OF THREE TERM RECURRENCE RELATIONS
S.I.A.M. REVIEW, VOLUME 9(1967), NUMBER 1.

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;

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[0: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: NONE.

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.

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
.664996657736O3"0   .396172970712 2"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;
```

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 VALJE 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 ACCORDING 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(48-.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;
```

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   .1175201193643 8" 1     .1175201193643 8" 1
1   .3678794411714 4" 0     .3678794411714 4" 0
2   .7156287012947 4"-1     .7156287012947 4"-1
3   .1006509052407 0"-1     .1006509052407 0"-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;

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.

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.

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;

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" N:= 0, 1, 2, 3 "DO"
    OUTPUT(61, "("/D, 2(5B-.14D"-ZD)")",
    N, K1[N], K2[N] * EXPX)
"END"
```

RESULTS:

```
0     .106292082896910"0     .106292082896910"0
1     .159438124345360"0     .159438124345360"0
2     .345449269414950"0     .345449269414940"0
3     .102306129788280"1     .102306129788280"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;
```

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 FJNCTION
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 ACCORDING 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; X:= ABS(X);
  R:= (35.9253 + N) / (1 + LN(1 + 17.96265 / N));
  S:= (52.8648 + X) / (1 + LN(1 + 26.4324 / X)) * 1.3591;
  M:= 6 + ENTIER("IF" R < S "THEN" S "ELSE" R); R:= 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" X= 0 "THEN"
  "BEGIN"
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" Y[N]:= - " +300
  "END" "ELSE" "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 (ABS (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;
"IF" X= 0 "THEN"
"BEGIN"
  "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" K[N]:= "+300 "END" "ELSE"
"BEGIN" "REAL" EXPX;
  "PROCEDURE" NONEXP SPHER BESS K(X, N, K); "CODE" 35155;
  EXPX:= EXP(-ABS(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;
"REAL" "PROCEDURE" SINH(X); "CODE" 35111;
"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;
  X:= ABS(X); X2:= X+X;
  I[0]:= X2:= "IF" X2 < .013 "THEN"
  1-X2*(1-X2*(1-X2*(1-X2/5)/4)/3)/2
  "ELSE" "IF" X2 < 0.7 "THEN" SINH(X) / ( X * EXP(X) )
  "ELSE" (1-EXP(-X2))/X2;
  "IF" N= 0 "THEN" "GO TO" EXIT;
  R:= (35.9253+N)/(1+LN(1+17.96265/N));
  S:= 1.3591*("IF" X > 35.9253 "THEN" X "ELSE"
  (52.8648-.4715*X)/(1+LN(.26424+26.4324/X)));
  M:= 6 + ENTIER("IF" R < S "THEN" S "ELSE" R); R:= 0;
  "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;
  "IF" X= 0 "THEN"
  "BEGIN"
    "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO" K[N]:= "+300 "END" "ELSE"
  "BEGIN" "INTEGER" I; "REAL" KI, KI1, KI2;
    X:= 1/ABS(X); K[0]:= KI2:= X*1.5707963267949;
    "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) = +3.287352554916 5"-010
AID(9.654894) = -1.029799932348 2"-009
BI (9.654894) = +1.558388704967 0"+008
BID(9.654894) = +4.801037468265 4"+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
+8.369910126198 6"-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:= SQRT1DPI/SQRT(SQRTZ);
    "IF" Z<0 "THEN"
    "BEGIN" Z:= -Z; CO:= COS(ZT-PIO4); SI:= SIN(ZT-PIO4);
        "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"
```