mathematisch centrum

DEPARTMENT OF NUMERICAL MATHEMATICS JUNE

NUMAL, A LIBRARY OF NUMERICAL PROCEDURES IN ALGOL 60

VOLUME 1. ELEMENTARY PROCEDURES

NU 1

amsterdam 1974

AUTHOR: P.A.BEENTJES.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES.
INIVEC INITIALIZES A (PART OF A) VECTOR WITH A CONSTANT.
INIMAT INITIALIZES A (PART OF A) MATRIX WITH A CONSTANT.
INIMATD INITIALIZES ELEMENTS A[I, I+SHIFT], I= LR(1)UR OF A MATRIX.
INISYMD INITIALIZES A (PART OF A) CODIAGONAL OF A SYMMETRIC MATRIX, WHOSE UPPERTRIANGLE IS STORED COLUMNWISE IN A ONE-DIMENSIONAL ARRAY.
INISYMROW INITIALIZES A (PART OF A) ROW OF A SYMMETRIC MATRIX,WHOSE UPPERTRIANGLE IS STORED COLUMNWISE IN A ONE-DIMENSIONAL ARRAY.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
INITIALIZATION.

SUBSECTION: INIVEC.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" INIVEC(L, U, A, X); "VALUE" L,U,X;
"INTEGER" L,U; "REAL" X; "ARRAY" A;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER INDEX OF THE VECTOR A, RESPECTIVELY;
A: <ARRAY IDENTIFIER>;
A ONE DIMENSIONAL ARRAY A[P : Q] WITH P <= L AND Q >= U;
X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: ALGOL 60.

SUBSECTION: INIMAT.

CALLING SEQUENCE:

```
HEADING;
"PROCEDURE" INIMAT(LR, UR, LC, UC, A, X); "VALUE" LR,UR,LC,UC,X;
"INTEGER" LR,UR,LC,UC; "REAL" X; "ARRAY" A;

FORMAL PARAMETERS;
LR,UR,LC,UC: <ARITHMETIC EXPRESSION>;
          LOWER AND UPPER ROW-INDEX, AND LOWER AND UPPER COLUMN-INDEX
          OF THE MATRIX A, RESPECTIVELY;
A;        <ARRAY IDENTIFIER>;
          A TWO-DIMENSIONAL ARRAY A[P : Q , R : S] WHOSE BOUNDS P, Q,
          R AND S SHOULD SATISFY: P <= LR, Q >= UR, R <= LC, S >= UC;
X;        <ARITHMETIC EXPRESSION>;
          INITIALIZATION CONSTANT.
```

LANGUAGE: ALGOL 60.

SUBSECTION: INIMATD.

CALLING SEQUENCE:

```
HEADING;
"PROCEDURE" INIMATD(LR, UR, SHIFT, A, X); "VALUE" LR,UR,SHIFT,X;
"INTEGER" LR,UR,SHIFT; "REAL" X; "ARRAY" A;

FORMAL PARAMETERS;
LR,UR:  <ARITHMETIC EXPRESSION>;
        LOWER   AND   UPPER  ROW-INDEX  OF  THE  CODIAGONAL  TO  BE
        INITIALIZED;
SHIFT:  <ARITHMETIC EXPRESSION>;
        DISTANCE BETWEEN DIAGONAL AND CODIAGONAL;
A;      <ARRAY IDENTIFIER>;
        A TWO-DIMENSIONAL ARRAY A[P : Q, R : S];
        THE SUBSCRIPTS ABOVE AND THE VALUES  OF  LR ( + SHIFT ) AND
        UR ( + SHIFT )   SHOULD   NOT   CONTRADICT   EACH   OTHER ;
   X;   <ARITHMETIC EXPRESSION>;
        INITIALIZATION CONSTANT.
```

LANGUAGE: ALGOL 60.

SUBSECTION: INISYMD.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" INISYMD(LR, UR, SHIFT, A, X); "VALUE" LR,UR,SHIFT,X;
"INTEGER" LR,UR,SHIFT; "REAL" X; "ARRAY" A;
```

FORMAL PARAMETERS:

LR,UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER ROW-INDEX OF A CODIAGONAL (OF A SYMMETRIC MATRIX OF ORDER N) TO BE INITIALIZED;
LR AND UR SHOULD SATISFY : LR >= 1, UR <= N;

SHIFT: <ARITHMETIC EXPRESSION>;
DISTANCE BETWEEN DIAGONAL AND CODIAGONAL, (-N < SHIFT < N);

A: <ARRAY IDENTIFIER>;
A ONE-DIMENSIONAL ARRAY A[1 : N * (N+1)//2] CONTAINING THE COLUMNWISE STORED UPPERTRIANGLE OF A SYMMETRIC MATRIX, SUCH THAT THE (I,J) - TH ELEMENT OF THE MATRIX IS A[(J - 1) * N + I]; J = 1, .. ,N; I = MAX(1,J-N), .. J;

X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: ALGOL 60.

SUBSECTION: INISYMROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" INISYMROW(L, U, I, A, X); "VALUE" L,U,I,X;
"INTEGER" L,U,I; "REAL" X; "ARRAY" A;
```

FORMAL PARAMETERS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER ROW-ELEMENT TO BE INITIALIZED ;

I: <ARITHMETIC EXPRESSION>;
ROW INDEX;

A: <ARRAY IDENTIFIER>;
A ONE-DIMENSIONAL ARRAY A[1 : N * (N+1)//2];
ARRAY A SHOULD CONTAIN A COLUMNWISE STORED UPPERTRIANGLE OF A SYMMETRIC MATRIX OF ORDER N,
SUCH THAT THE (I,J) - TH ELEMENT OF THE MATRIX IS A[(J - 1) * N + I]; J = 1, .. ,N; I = MAX(1,J-N), .. J.
FOR FIXED ORDER N, THE PARAMETERS L, U AND I SHOULD SATISFY THE CONDITIONS : 1 <=L<= N, 1 <=J<= N, 1 <=I<= N ;

X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: ALGOL 60.

SOURCE TEXT(S):

```
"CODE" 31010;
    "PROCEDURE" INIVEC(L, U, A, X); "VALUE" L,U,X;
    "INTEGER" L,U; "REAL" X; "ARRAY" A;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= X;
        "EOP"

"CODE" 31011;
    "PROCEDURE" INIMAT(LR, UR, LC, UC, A, X); "VALUE" LR,UR,LC,UC,X;
    "INTEGER" LR,UR,LC,UC; "REAL" X; "ARRAY" A;
    "BEGIN" "INTEGER" J;

        "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
        "FOR" J:= LC "STEP" 1 "UNTIL" UC "DO" A[LR, J]:= X
    "END" INIMAT;
        "EOP"

"CODE" 31012;
    "PROCEDURE" INIMATD(LR, UR, SHIFT, A, X); "VALUE" LR,UR,SHIFT,X;
    "INTEGER" LR,UR,SHIFT; "REAL" X; "ARRAY" A;
    "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO" A[LR, LR + SHIFT]:= X;
        "EOP"

"CODE" 31013;
    "PROCEDURE" INISYMD(LR, UR, SHIFT, A, X); "VALUE" LR,UR,SHIFT,X;
    "INTEGER" LR,UR,SHIFT; "REAL" X; "ARRAY" A;
    "BEGIN" SHIFT:= ABS(SHIFT); UR:= UR + SHIFT + 1; SHIFT:=LR + SHIFT;
        LR := (SHIFT - 3) * SHIFT // 2 + LR;
        "FOR" LR := SHIFT + LR "WHILE" SHIFT < UR "DO"
        "BEGIN" A[LR]:= X; SHIFT:= SHIFT + 1 "END"
    "END" INISYMD;
        "EOP"

"CODE" 31014;
    "PROCEDURE" INISYMROW(L, U, I, A, X); "VALUE" L,U,I,X;
    "INTEGER" L,U,I; "REAL" X; "ARRAY" A;
"BEGIN" "INTEGER" K;
    "IF" L <= 1 "THEN"
    "BEGIN" K:= (I - 1) * I//2; L := K + L;
        K := ("IF" U < 1 "THEN" U "ELSE" I) + K;
        "FOR" L:= L "STEP" 1 "UNTIL" K "DO" A[L]:= X;
        L := I + 1
    "END";
    "IF" U>1 "THEN""FOR" K:=(L-1)*L//2+I, K+L-1 "WHILE" L<= U "DO"
    "BEGIN" A[K]:= X; L:= L + 1 "END"
"END" INISYMROW;
    "EOP"
```

AUTHOR: P.A.BEENTJES.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SIX PROCEDURES.
DUPVEC COPIES THE VECTOR GIVEN IN ARRAY B[L+SHIFT : U+SHIFT] TO THE VECTOR GIVEN IN ARRAY A[L:U].
DUPVECROW COPIES THE ROW VECTOR GIVEN IN ARRAY B[I:I, L:U] TO THE VECTOR GIVEN IN ARRAY A[L:U].
DUPROWVEC COPIES THE VECTOR GIVEN IN ARRAY B[L:U] TO THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U].
DUPVECCOL COPIES THE COLUMN VECTOR GIVEN IN ARRAY B[L:U, J:J] TO THE VECTOR GIVEN IN ARRAY A[L:U].
DUPCOLVEC COPIES THE VECTOR GIVEN IN ARRAY B[L:U] TO THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, J:J].
DUPMAT COPIES THE MATRIX GIVEN IN ARRAY B[L:U, I:J] TO THE MATRIX GIVEN IN ARRAY A[L:U, I:J].

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
DUPLICATION.

SUBSECTION: DUPVEC.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" DUPVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
"INTEGER" L,U,SHIFT; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER VECTOR-INDEX, RESPECTIVELY;
SHIFT: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETER;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L (+ SHIFT) AND U(+SHIFT) SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: ALGOL 60.

SUBSECTION: DUPVECROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" DUPVECROW(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER   AND   UPPER  VECTOR ( COLUMN )-INDEX, RESPECTIVELY;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

SUBSECTION: DUPROWVEC.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" DUPROWVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER   AND   UPPER  VECTOR ( COLUMN )-INDEX, RESPECTIVELY;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR A;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

SUBSECTION: DUPVECCOL.

CALLING SEQUENCE:

```
HEADING;
"PROCEDURE" DUPVECCOL(L, U, J, A, B); "VALUE" L,U,J;
"INTEGER" L,U,J; "ARRAY" A,B;

FORMAL PARAMETERS;
L,U;     <ARITHMETIC EXPRESSION>;
         LOWER   AND   UPPER   VECTOR ( ROW )-INDEX,   RESPECTIVELY;
J;       <ARITHMETIC EXPRESSION>;
         COLUMN-INDEX OF THE COLUMN VECTOR B;
A,B;     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  J  SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

SUBSECTION: DUPCOLVEC.

CALLING SEQUENCE:

```
HEADING;
"PROCEDURE" DUPCOLVEC(L, U, J, A, B); "VALUE" L,U,J;
"INTEGER" L,U,J; "ARRAY" A,B;

FORMAL PARAMETERS;
L,U;     <ARITHMETIC EXPRESSION>;
         LOWER   AND   UPPER   VECTOR ( ROW )-INDEX,   RESPECTIVELY;
J;       <ARITHMETIC EXPRESSION>;
         COLUMN-INDEX OF THE COLUMN VECTOR A;
A,B;     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  J  SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

SUBSECTION: DUPMAT.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" DUPMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER ROW-INDEX, RESPECTIVELY;
I,J:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER COLUMN-INDEX, RESPECTIVELY;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

SOURCE TEXT(S):

```
"CODE" 31030;
    "PROCEDURE" DUPVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
    "INTEGER" L,U,SHIFT; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= B[L+SHIFT];
        "EOP"

"CODE" 31031;
    "PROCEDURE" DUPVECROW(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= B[I,L];
        "EOP"

"CODE" 31032;
    "PROCEDURE" DUPROWVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= B[L];
        "EOP"

"CODE" 31033;
    "PROCEDURE" DUPVECCOL(L, U, J, A, B); "VALUE" L,U,J;
    "INTEGER" L,U,J; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= B[L,J];
        "EOP"

"CODE" 31034;
    "PROCEDURE" DUPCOLVEC(L, U, J, A, B); "VALUE" L,U,J;
    "INTEGER" L,U,J; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,J]:= B[L];
        "EOP"

"CODE" 31035;
    "PROCEDURE" DUPMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A,B;
    "BEGIN" "INTEGER" K;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "FOR" K:= I "STEP" 1 "UNTIL" J "DO" A[L,K]:= B[L,K]
"END" DUPMAT;
    "EOP"
```

AUTHORS: P.A.BEENTJES, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FIVE PROCEDURES.
MULVEC STORES X TIMES THE VECTOR GIVEN IN ARRAY B[L+SHIFT:U+SHIFT] INTO THE VECTOR GIVEN IN ARRAY A[L:U].
MULROW STORES X TIMES THE ROW VECTOR GIVEN IN ARRAY B[J:J,L:U] INTO THE ROW VECTOR GIVEN IN ARRAY A[I:I,L:U].
MULCOL STORES X TIMES THE COLUMN VECTOR GIVEN IN ARRAY B[L:U,J:J] INTO THE COLUMN VECTOR GIVEN IN ARRAY A[L:U,I:I].
COLCST MULTIPLIES THE COLUMN VECTOR GIVEN IN ARRAY A[L:U,J:J] BY X.
ROWCST MULTIPLIES THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U] BY X.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
MULTIPLICATION.

SUBSECTION: MULVEC.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" MULVEC(L, U, SHIFT, A, B, X); "VALUE" L,U,SHIFT,X;
"INTEGER" L,U,SHIFT; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:

L,U: <ARITMETIC EXPRESSION>;
LOWER AND UPPER VECTOR-INDEX, RESPECTIVELY;
SHIFT: <ARITHMETIC EXPRESSION>;
SUBSCRIPT-SHIFTING PARAMETER;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L (+ SHIFT) AND U (+SHIFT) SHOULD NOT CONTRADICT EACH OTHER;
X: <ARITHMETIC EXPRESSION>;
MULTIPLICATION FACTOR.

LANGUAGE: ALGOL 60.

SUBSECTION: MULROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" MULROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
"INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:    <ARITMETIC EXPRESSION>;
        LOWER AND UPPER COLUMN-INDEX, RESPECTIVELY;
I,J:    <ARITHMETIC EXPRESSION>;
        ROW-INDICES OF THE ROW VECTORS A AND B;
A,B:    <ARRAY IDENTIFIER>;
        "ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
        THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
        NOT CONTRADICT EACH OTHER;
X:      <ARITHMETIC EXPRESSION>;
        MULTIPLICATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: MULCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" MULCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
"INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:    <ARITMETIC EXPRESSION>;
        LOWER AND UPPER ROW-INDEX, RESPECTIVELY,
I,J:    <ARITHMETIC EXPRESSION>;
        COLUMN-INDICES OF THE COLUMN VECTORS A AND B;
A,B:    <ARRAY IDENTIFIER>;
        "ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
        THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
        NOT CONTRADICT EACH OTHER;
X:      <ARITHMETIC EXPRESSION>;
        MULTIPLICATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: COLCST.

CALLING SEQUENCE:

```
HEADING;
"PROCEDURE" COLCST(L, U, J, A, X); "VALUE" L,U,J,X;
"INTEGER" L,U,J; "REAL" X; "ARRAY" A;

FORMAL PARAMETERS;
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER ROW-INDEX, RESPECTIVELY;
J:       <ARITHMETIC EXPRESSION>;
         COLUMN-INDEX;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  J  SHOULD
         NOT CONTRADICT EACH OTHER;
X:       <ARITHMETIC EXPRESSION>;
         MULTIPLICATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: ROWCST.

CALLING SEQUENCE:

```
    HEADING;
    "PROCEDURE" ROWCST(L, U, I, A, X); "VALUE" L,U,I,X;
    "INTEGER" L,U,I; "REAL" X; "ARRAY" A;

    FORMAL PARAMETERS;
    L,U:     <ARITHMETIC EXPRESSION>;
             LOWER AND UPPER COLUMN-INDEX, RESPECTIVELY;
    I:       <ARITHMETIC EXPRESSION>;
             ROW-INDEX;
    A:       <ARRAY IDENTIFIER>;
             "ARRAY" A[I1:I2, J1:J2];
             THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
             NOT CONTRADICT EACH OTHER;
    X:       <ARITHMETIC EXPRESSION>;
             MULTIPLICATION FACTOR.
```

LANGUAGE: ALGOL 60.

```
SOURCE TEXT(S);


"CODE" 31020;
    "PROCEDURE" MULVEC(L, U, SHIFT, A, B, X); "VALUE" L,U,SHIFT,X;
    "INTEGER" L,U,SHIFT; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= B[L+SHIFT]*X;
        "EOP"

"CODE" 31021;
    "PROCEDURE" MULROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
    "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= B[J,L]*X;
        "EOP"

"CODE" 31022;
    "PROCEDURE" MULCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
    "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,I]:= B[L,J]*X;
        "EOP"

"CODE" 31131;
    "PROCEDURE" COLCST(L, U, J, A, X); "VALUE" L,U,J,X;
    "INTEGER" L,U,J; "REAL" X; "ARRAY" A;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,J]:= A[L,J] * X;
        "EOP"

"CODE" 31132;
    "PROCEDURE" ROWCST(L, U, I, A, X); "VALUE" L,U,I,X;
    "INTEGER" L,U,I; "REAL" X; "ARRAY" A;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= A[I,L] * X;
        "EOP"
```

AUTHORS: T.J.DEKKER, J.C.P.BUS.

CONTRIBUTOR: P.A.BEENTJES.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS NINE PROCEDURES.
VECVEC:= SCALAR PRODUCT OF THE VECTOR GIVEN IN ARRAY A[L:U] AND ARRAY B[SHIFT + L : SHIFT + U].
MATVEC:= SCALAR PRODUCT OF THE ROW VECTOR GIVEN IN ARRAY A[I:I,L:U] AND THE VECTOR GIVEN IN ARRAY B[L:U].
TAMVEC:= SCALAR PRODUCT OF THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, I:I] AND THE VECTOR GIVEN IN ARRAY B[L:U].
MATMAT:= SCALAR PRODUCT OF THE ROW VECTOR GIVEN IN ARRAY A[I:I,L:U] AND THE COLUMN VECTOR IN ARRAY B[L:U, J:J].
TAMMAT:= SCALAR PRODUCT OF THE COLUMN VECTORS GIVEN IN ARRAY A[L:U, I:I] AND ARRAY B[L:U, J:J].
MATTAM := SCALAR PRODUCT OF THE ROW VECTORS GIVEN IN ARRAY A[I:I,L:U] AND ARRAY B[J:J, L:U].
SEQVEC := SCALAR PRODUCT OF THE VECTORS GIVEN IN ARRAY A[IL : IL + (U+L-1)*(U-L)//2] AND ARRAY B[SHIFT + L : SHIFT + U], WHERE THE ELEMENTS OF THE FIRST VECTOR ARE A[IL+(J+L-1)*(J-L)//2] FOR J = L, ..., U.
SCAPRD1:= SCALAR PRODUCT OF THE VECTORS GIVEN IN ARRAY A[MIN(LA, LA + (N - 1) * SA) : MAX(LA,LA + (N - 1) * SA)] AND ARRAY B[MIN(LB, LB + (N - 1) * SB) : MAX(LB,LB + (N - 1) * SB)] WHERE THE ELEMENTS OF THE VECTORS ARE A[LA+(J-1)*SA] AND B[LB+(J-1)*SB] FOR J = 1, ..., N.
SYMMATVEC CALCULATES THE INNERPRODUCT OF (A PART OF) A VECTOR AND (A PART OF) A ROW OF A SYMMETRIC MATRIX , WHOSE UPPERTRIANGLE IS GIVEN COLUMNWISE IN A ONE-DIMENSIONAL ARRAY.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
SCALAR PRODUCTS,

SUBSECTION: VECVEC.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
"INTEGER" L,U,SHIFT; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
SHIFT:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETER OF THE VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES  OF  L ( + SHIFT )  AND
         U(+SHIFT) SHOULD NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: MATVEC.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR A;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: TAMVEC.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;
```

FORMAL PARAMETERS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT,

I: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR A,

A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U AND I SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: MATMAT.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;
```

FORMAL PARAMETERS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;

I,J: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR A AND COLUMN-INDEX OF THE COLUMN VECTOR B;

A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: TAMMAT.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:    <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J:    <ARITHMETIC EXPRESSION>;
        COLUMN-INDICES OF THE COLUMN VECTORS A AND B, RESPECTIVELY;
A,B:    <ARRAY IDENTIFIER>;
        "ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
        THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
        NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: MATTAM.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:    <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J:    <ARITHMETIC EXPRESSION>;
        ROW-INDICES OF THE ROW VECTORS A AND B, RESPECTIVELY;
A,B:    <ARRAY IDENTIFIER>;
        "ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
        THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
        NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: SEQVEC.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);
"VALUE" L,U,IL,SHIFT; "INTEGER" L,U,IL,SHIFT; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
IL:      <ARITHMETIC EXPRESSION>;
         LOWER BOUND OF THE VECTOR A;
SHIFT:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETER OF THE VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L(+SHIFT), U(+SHIFT)
         AND IL(+(U+L-1)*(U-L)//2) SHOULD NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: SCAPRD1.

CALLING SEQUENCE:

```
HEADING:
"REAL" "PROCEDURE" SCAPRD1(LA, SA, LB, SB, N, A, B);
"VALUE" LA,SA,LB,SB,N; "INTEGER" LA,SA,LB,SB,N; "ARRAY" A,B;

FORMAL PARAMETERS:
N:       <ARITHMETIC EXPRESSION>;
         UPPER BOUND OF THE RUNNING SUBSCRIPT;
LA,LB:   <ARITHMETIC EXPRESSION>;
         LOWER BOUNDS OF THE VECTORS A AND B, RESPECTIVELY;
SA,SB:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETERS OF THE VECTORS A AND B,
         RESPECTIVELY;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF LA( +(J-1)*SA ) AND
         LB( +(J-1)*SB ),J = 1(1)N SHOULD NOT CONTRADICT EACH OTHER.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: SYMMATVEC.

CALLING SEQUENCE:

```
HEADING:
"REAL" PROCEDURE" SYMMATVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE VECTOR B, RESPECTIVELY; L >=1;
I:       <ARITHMETIC EXPRESSION>;
         ROW INDEX OF THE MATRIX A; I >= 1;
A:       <ARRAY IDENTIFIER>;
         A  ONE-DIMENSIONAL  ARRAY  A[P : Q]  WITH:  IF  I > L  THEN
         P=(I-1)*I//2 + L ELSE  P=(L-1)*L//2 + I AND IF  I > U  THEN
         Q=(I-1)*I//2 + U ELSE  Q=(U-1)*U//2 + I;
B:       <ARRAY IDENTIFIER>;
         A ONE-DIMENSIONAL ARRAY B[L:U];
SYMMATVEC: <PROCEDURE IDENTIFIER>;
         EXIT: THE VALUE OF THE SCALAR PRODUCTS OF THE VECTORS GIVEN
              IN  ARRAY A[P:Q] AND  ARRAY B[L:U] , WHERE THE ELEMENTS
              OF THE FIRST VECTOR ARE: IF L<I THEN A[(I-1)*I//2 + J],
              J=L,...., MIN(U, I-1) AND A[(J-1)*J//2 + I], J=I,...., U,
              RESPECTIVELY , OTHERWISE A[(J-1)*J//2 + I], J=L,...., U.
```

PROCEDURES USED:

VECVEC = CP34010,
SEQVEC = CP34016.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [2].

REFERENCES:

[1] .T.J.DEKKER,
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART1,
MATHEMATICAL CENTRE TRACT 22, AMSTERDAM (1970).

[2] .J.C.P.BUS,
MINIMALISERING VAN FUNKTIES VAN MEERDERE VARIABELEN,
MATHEMATICAL CENTRE, NR 29/72, AMSTERDAM (1972).

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES, EXCEPT SYMMATVEC, ARE WRITTEN IN COMPASS 3,
THE ALGOL TEXT OF THE COMPASS ROUTINES IS GIVEN.

```
"CODE" 34010;
    "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
    "INTEGER" L,U,SHIFT; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K] * B[SHIFT + K] + S;
        VECVEC:= S
    "END" VECVEC;
        "EOP"

"CODE" 34011;
    "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[K] + S;
        MATVEC:= S
    "END" MATVEC;
        "EOP"

"CODE" 34012;
    "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K,I] * B[K] + S;
        TAMVEC:= S
    "END" TAMVEC;
        "EOP"

"CODE" 34013;
    "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[K,J] + S;
        MATMAT:= S
"END" MATMAT;
        "EOP"
```

```
"CODE" 34014;
     "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
     "INTEGER" L,U,I,J; "ARRAY" A,B;
     "BEGIN" "INTEGER" K; "REAL" S;
          S:= 0;
          "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K,I] * B[K,J] + S;
          TAMMAT:= S
     "END" TAMMAT;
          "EOP"

"CODE" 34015;
     "REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "VALUE" L,U,I,J;
     "INTEGER" L,U,I,J; "ARRAY" A,B;
     "BEGIN" "INTEGER" K; "REAL" S;
          S:= 0;
          "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[J,K] + S;
          MATTAM:= S
     "END" MATTAM;
          "EOP"

"CODE" 34016;
     "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);
     "VALUE" L,U,IL,SHIFT; "INTEGER" L,U,IL,SHIFT; "ARRAY" A,B;
     "BEGIN" "INTEGER" K; "REAL" S;
          S:= 0;
          "FOR" L:=L "STEP" 1 "UNTIL" U "DO"
          "BEGIN" S:= A[IL] * B[L + SHIFT] + S; IL:= IL + L "END";
          SEQVEC:= S
     "END" SEQVEC;
          "EOP"

"CODE" 34017;
     "REAL" "PROCEDURE" SCAPRD1(LA, SA, LB, SB, N, A, B);
     "VALUE" LA,SA,LB,SB,N; "INTEGER" LA,SA,LB,SB,N; "ARRAY" A,B;
     "BEGIN" "REAL" S;"INTEGER" K;
          S:= 0;
          "FOR" K:= 1 "STEP" 1 "UNTIL" N "DO"
          "BEGIN" S:=A[LA] * B[LB] + S; LA:= LA + SA; LB:= LB + SB "END";
          SCAPRD1:= S
     "END" SCAPRD1;
          "EOP"

"CODE" 34018;
     "REAL" "PROCEDURE" SYMMATVEC(L, U, I, A, B); "VALUE" L,U,I;
     "INTEGER" L,U,I; "ARRAY" A,B;
     "BEGIN" "INTEGER" K, M;
          "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
          "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B); "CODE" 34016;
          M:= "IF" L > I "THEN" L "ELSE" I; K:= M * (M - 1) // 2;
          SYMMATVEC:= VECVEC(L, "IF" I <= U "THEN" I-1 "ELSE" U, K, B, A)
                    + SEQVEC(M, U, K + I, 0, A, B)
     "END" SYMMATVEC;
          "EOP"
```

AUTHORS: T.J.DEKKER, J.C.P.BUS, J.WOLLESWINKEL.

CONTRIBUTORS: P.A.BEENTJES, J.C.P.BJS.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 741215.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS NINE PROCEDURES.
VECVEC:= SCALAR PRODUCT OF THE VECTOR GIVEN IN ARRAY A[L:U] AND ARRAY B[SHIFT + L : SHIFT + U].
MATVEC:= SCALAR PRODUCT OF THE ROW VECTOR GIVEN IN ARRAY A[I:I,L:U] AND THE VECTOR GIVEN IN ARRAY B[L:U].
TAMVEC:= SCALAR PRODUCT OF THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, I:I] AND THE VECTOR GIVEN IN ARRAY B[L:U].
MATMAT:= SCALAR PRODUCT OF THE ROW VECTOR GIVEN IN ARRAY A[I:I,L:U] AND THE COLUMN VECTOR IN ARRAY B[L:U, J:J].
TAMMAT:= SCALAR PRODUCT OF THE COLUMN VECTORS GIVEN IN ARRAY A[L:U, I:I] AND ARRAY B[L:U, J:J].
MATTAM := SCALAR PRODUCT OF THE ROW VECTORS GIVEN IN ARRAY A[I:I,L:U] AND ARRAY B[J:J, L:U].
SEQVEC := SCALAR PRODUCT OF THE VECTORS GIVEN IN ARRAY A[IL : IL + (U+L-1)*(U-L)//2] AND ARRAY B[SHIFT + L : SHIFT + U], WHERE THE ELEMENTS OF THE FIRST VECTOR ARE A[IL+(J+L-1)*(J-L)//2] FOR J = L, ..., U.
SCAPRD1:= SCALAR PRODUCT OF THE VECTORS GIVEN IN ARRAY A[MIN(LA, LA + (N - 1) * SA) : MAX(LA,LA + (N - 1) * SA)] AND ARRAY B[MIN(LB, LB + (N - 1) * SB) : MAX(LB,LB + (N - 1) * SB)] WHERE THE ELEMENTS OF THE VECTORS ARE A[LA+(J-1)*SA] AND B[LB+(J-1)*SB] FOR J = 1, ..., N.
SYMMATVEC := THE SCALARPRODJCT OF (A PART OF) A VECTOR AND (A PART OF) A ROW OF A SYMMETRIC MATRIX , WHOSE UPPERTRIANGLE IS GIVEN COLUMNWISE IN AN ONE-DIMENSIONAL ARRAY.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
SCALAR PRODUCTS.

SUBSECTION: VECVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
"INTEGER" L,U,SHIFT; "ARRAY" A,B;
"CODE" 34010;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
SHIFT: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETER OF THE VECTOR B;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L(+SHIFT) AND U(+SHIFT) SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: MATVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL""PROCEDURE" MATVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;
"CODE" 34011;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR A;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L, U AND I SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: TAMVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;
"CODE" 34012;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR A;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L, U AND I
SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: MATMAT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;
"CODE" 34013;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR A AND COLUMN-INDEX OF THE
COLUMN VECTOR B;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L, U AND I
SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: TAMMAT.

CALLING SEQJENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;
"CODE" 34014;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J: <ARITHMETIC EXPRESSION>;
COLUMN-INDICES OF THE COLUMN VECTORS A AND B, RESPECTIVELY;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L, U AND I
SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: MATTAM.

CALLING SEQJENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A,B;
"CODE" 34015;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J: <ARITHMETIC EXPRESSION>;
ROW-INDICES OF THE ROW VECTORS A AND B, RESPECTIVELY;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4, J3:J4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L, U AND I
SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: SEQVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);
"VALUE" L,U,IL,SHIFT; "INTEGER" L,U,IL,SHIFT; "ARRAY" A,B;
"CODE" 34016;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
IL: <ARITHMETIC EXPRESSION>;
LOWER BOUND OF THE VECTOR A;
SHIFT: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETER OF THE VECTOR B;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF L(+SHIFT)
U(+SHIFT) AND IL(+(U+L-1)*(U-L)//2) SHOULD NOT CONTRADICT
EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: SCAPRD1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" SCAPRD1(LA, SA, LB, SB, N, A, B);
"VALUE" LA,SA,LB,SB,N; "INTEGER" LA,SA,LB,SB,N; "ARRAY" A,B;
"CODE" 34017;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
UPPER BOUND OF THE RUNNING SUBSCRIPT;
LA,LB: <ARITHMETIC EXPRESSION>;
LOWER OR UPPER SUBSCRIPT BOUNDS OF THE VECTORS A AND B,
RESPECTIVELY;
SA,SB: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETERS OF THE VECTORS A AND B,
RESPECTIVELY;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTBOUNDS ABOVE AND THE VALUES OF LA,LA+(N-1)*SA,
LB, LB + (N - 1) * SB SHOULD NOT CONTRADICT EACH OTHER.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: SYMMATVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"REAL" PROCEDURE" SYMMATVEC(L, U, I, A, B); "VALUE" L,U,I;
"INTEGER" L,U,I; "ARRAY" A,B;
"CODE" 34018;
```

SYMMATVEC:= THE VALUE OF THE SCALAR PRODUCT OF THE VECTORS GIVEN IN ARRAY A[P:Q] AND ARRAY B[L:U] , WHERE THE ELEMENTS OF THE FIRST VECTOR ARE: IF L≤I THEN A[(I-1)*I//2 + J], J=L,..., MIN(U, I-1) AND A[(J-1)*J//2 + I], J=I,..., U, RESPECTIVELY , OTHERWISE A[(J-1)*J//2 + I], J=L,..., U.

THE MEANING OF THE FORMAL PARAMETERS IS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE VECTOR B, RESPECTIVELY; L >=1;

I: <ARITHMETIC EXPRESSION>;
ROW INDEX OF THE MATRIX A; I >= 1;

A: <ARRAY IDENTIFIER>;
A ONE-DIMENSIONAL ARRAY A[P : Q] WITH: IF I > L THEN P=(I-1)*I//2 + L ELSE P=(L-1)*L//2 + I AND IF I > U THEN Q=(I-1)*I//2 + U ELSE Q=(U-1)*U//2 + I;

B: <ARRAY IDENTIFIER>;
A ONE-DIMENSIONAL ARRAY B[L:U];

PROCEDURES USED:

VECVEC = CP34010,
SEQVEC = CP34016.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [2].

REFERENCES :

[1] T.J.DEKKER.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA,PART 1,
MATHEMATICAL CENTRE TRACT 22,AMSTERDAM (1970)

[2] J.C.P.BUS.
MINIMALISERING VAN FUNCTIES VAN MEERDERE VARIABELEN,
MATHEMATICAL CENTRE, NR 29/72,AMSTERDAM (1972)

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES, EXCEPT FOR SYMMATVEC, FULSYMMATVEC AND SYMRESVEC ARE WRITTEN IN COMPASS 3; AN EQUIVALENT ALGOL TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"CODE" 34010;
    "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "VALUE" L,U,SHIFT;
    "INTEGER" L,U,SHIFT; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K] * B[SHIFT + K] + S;
        VECVEC:= S
    "END" VECVEC;
        "EOP"

"CODE" 34011;
    "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[K] + S;
        MATVEC:= S
    "END" MATVEC;
        "EOP"

"CODE" 34012;
    "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K,I] * B[K] + S;
        TAMVEC:= S
    "END" TAMVEC;
        "EOP"

"CODE" 34013;
    "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S:= 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[K,J] + S;
        MATMAT:= S
"END" MATMAT;
        "EOP"
```

```
"CODE" 34014;
    "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S := 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[K,I] * B[K,J] + S;
        TAMMAT:= S
    "END" TAMMAT;
        "EOP"

"CODE" 34015;
    "REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" S;
        S := 0;
        "FOR" K:=L "STEP" 1 "UNTIL" U "DO" S:= A[I,K] * B[J,K] + S;
        MATTAM:= S
    "END" MATTAM;
        "EOP"

"CODE" 34016;
    "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);
    "VALUE" L,U,IL,SHIFT; "INTEGER" L,U,IL,SHIFT; "ARRAY" A,B;
    "BEGIN" "REAL" S;
        S := 0;
        "FOR" L:=L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" S:= A[IL] * B[L + SHIFT] + S; IL:= IL + L "END";
        SEQVEC:= S
    "END" SEQVEC;
        "EOP"

"CODE" 34017;
    "REAL" "PROCEDURE" SCAPRD1(LA, SA, LB, SB, N, A, B);
    "VALUE" LA,SA,LB,SB,N; "INTEGER" LA,SA,LB,SB,N; "ARRAY" A,B;
    "BEGIN" "REAL" S;"INTEGER" K;
        S := 0;
        "FOR" K:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" S:=A[LA] * B[LB] + S; LA:= LA + SA; LB:= LB + SB "END";
        SCAPRD1:= S
    "END" SCAPRD1;
        "EOP"

"COUE" 34018;
    "REAL" "PROCEDURE" SYMMATVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "BEGIN" "INTEGER" K, M;
        "REAL" "PROCEDURE" VECVEC(L, U, SHIFT, A, B); "CODE" 34010;
        "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B); "CODE" 34016;
        M:= "IF" L > I "THEN" L "ELSE" I; K:= M * (M - 1) // 2;
        SYMMATVEC:= VECVEC(L, "IF" I <= U "THEN" I-1 "ELSE" U, K, B, A)
                    + SEQVEC(M, U, K + I, 0, A, B)
    "END" SYMMATVEC;
        "EOP"
```

AUTHOR : D.WINTER.

INSTITUTE : MATHEMATICAL CENTRE.

RECEIVED : 741215.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS FIVE PROCEDURES :
FULMATVEC CALCULATES THE VECTOR A * B, WHERE A IS A GIVEN MATRIX AND B IS A VECTOR.
FULTAMVEC CALCULATES THE VECTOR A' * B, WHERE A' IS THE TRANSPOSED OF THE MATRIX A AND B IS A VECTOR.
FULSYMMATVEC CALCULATES THE VECTOR A * B, WHERE A IS A SYMMETRIC MATRIX WHOSE UPPERTRIANGLE IS STORED COLUMNWISE IN AN ..
ONE-DIMENSIONAL ARRAY AND B IS A VECTOR.
RESVEC CALCULATES THE RESIDUAL VECTOR A * B + X * C, WHERE A IS A GIVEN MATRIX, B AND C ARE VECTORS AND X IS A SCALAR.
SYMRESVEC CALCULATES THE RESIDUAL VECTOR A * B + X * C, WHERE A IS A SYMMETRIC MATRIX WHOSE UPPERTRIANGLE IS STORED IN A ONE-DIMENSIONAL ARRAY, B AND C ARE VECTORS AND X IS A SCALAR.

KEYWORDS :

ELEMENTARY PROCEDURE,
VECTOR OPERATION.

SUBSECTION: FULMATVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" FULMATVEC(LR, UR, LC, UC, A, B, C);
"VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
"ARRAY" A, B, C;
"CODE" 31500;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW-INDEX;
LC, UC: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOJND OF THE COLUMN-INDEX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[LC:UC]; THE VECTOR;
C: <ARRAY IDENTIFIER>;
"ARRAY" C[LR:UR];
THE RESULT A * B IS DELIVERED IN C.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: FULTAMVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" FULTAMVEC(LR, UR, LC, UC, A, B, C);
"VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
"ARRAY" A, B, C;
"CODE" 31501;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW-INDEX;
LC, UC: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE COLUMN-INDEX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[LR:UR]; THE VECTOR;
C: <ARRAY IDENTIFIER>;
"ARRAY" C[LC:UC];
THE RESULT A' * B IS DELIVERED IN C; HERE A' DENOTES THE
TRANSPOSED OF THE MATRIX A.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE: ELEMENTARY.

SUBSECTION: FULSYMMATVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" FULSYMMATVEC(LR, UR, LC, UC, A, B, C);
"VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
"ARRAY" A, B, C;
"CODE" 31502;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW-INDEX; LR >= 1;
LC, UC: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE COLUMN-INDEX; LC >= 1;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[L:U], WHERE:
L = MIN(LR * (LR - 1) // 2 + LC, LC * (LC - 1) // 2 + LR),
U = MAX(UR * (UR - 1) // 2 + UC, UC * (UC - 1) // 2 + UR)
AND THE (I,J)-TH ELEMENT OF THE SYMMETRIC MATRIX SHOULD BE
GIVEN IN A[J * (J - 1) // 2 + I];
B: <ARRAY IDENTIFIER>;
"ARRAY" B[LC:UC]; THE VECTOR;
C: <ARRAY IDENTIFIER>;
"ARRAY" C[LR:UR];
THE RESULT A * B IS DELIVERED IN C.

PROCEDURES USED:

SYMMATVEC = CP34018.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: RESVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" RESVEC(LR, UR, LC, UC, A, B, C, X);
"VALUE" LR, UR, LC, UC, B, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" A, B, C;
"CODE" 31503;


THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE ROW-INDEX;
LC, UC: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE COLUMN-INDEX;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
B:      <ARRAY IDENTIFIER>;
        "ARRAY" B[LC:UC]; THE VECTOR;
X:      <ARITHMETIC EXPRESSION>;
        THE VALUE OF THE MULTIPLYING SCALAR;
C:      <ARRAY IDENTIFIER>;
        "ARRAY" C[LR:UR];
        THE RESULT A * B + X * C IS OVERWRITTEN ON C.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: SYMRESVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" SYMRESVEC(LR, UR, LC, UC, A, B, C, X);
"VALUE" LR, UR, LC, UC, B, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" A, B, C;
"CODE" 31504;


THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOJND OF THE ROW-INDEX; LR >= 1;
LC, UC: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE COLUMN-INDEX; LC >= 1;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[L:U], WHERE:
        L = MIN(LR * (LR - 1) // 2 + LC, LC * (LC - 1) // 2 + LR),
        U = MAX(UR * (UR - 1) // 2 + UC, UC * (UC - 1) // 2 + UR)
        AND THE (I,J)-TH ELEMENT OF THE SYMMETRIC MATRIX SHOULD BE
        GIVEN IN A[J * (J - 1) // 2 + I];
    B:  <ARRAY IDENTIFIER>;
        "ARRAY" B[LC:UC]; THE VECTOR;
    X:  <ARITHMETIC EXPRESSION>;
        THE VALUE OF THE MULTIPLYING SCALAR;
    C:  <ARRAY IDENTIFIER>;
        "ARRAY" C[LR:UR];
        THE RESULT A * B  + X * C IS DELIVERED IN C.
```

PROCEDURES USED:

SYMMATVEC = CP34018.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

ELEMENTARY.

REFERENCES:

[1].T.J.DEKKER.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART1,
MATHEMATICAL CENTRE TRACT 22, AMSTERDAM (1970).

[2].J.C.P.BUS.
MINIMALISERING VAN FUNKTIES VAN MEERDERE VARIABELEN,
MATHEMATICAL CENTRE, NR 29/72, AMSTERDAM (1972).

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES, EXCEPT FOR FULSYMMATVEC AND SYMRESVEC ARE WRITTEN IN COMPASS 3, AN EQUIVALENT ALGOL TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"CODE" 31500;
    "PROCEDURE" FULMATVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, JC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
        "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
        C[LR]:= MATVEC(LC, UC, LR, A, B);
    "END" FULMATVEC;
         "EOP"

"CODE" 31501;
    "PROCEDURE" FULTAMVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;
        "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
        C[LC]:= TAMVEC(LR, UR, LC, A, B);
    "END" FULTAMVEC;
         "EOP"

"CODE" 31502;
    "PROCEDURE" FULSYMMATVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" "PROCEDURE" SYMMATVEC(L, U, I, A, B);
        "CODE" 34018;
        "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
        C[LR]:= SYMMATVEC(LC, UC, LR, A, B)
    "END" FULSYMMATVEC;
         "EOP"

"CODE" 31503;
    "PROCEDURE" RESVEC(LR, UR, LC, UC, A, B, C, X);
    "VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
    "REAL" X; "ARRAY" A, B, C;
    "BEGIN" "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
        "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
        C[LR]:= MATVEC(LC, UC, LR, A, B) + C[LR] * X
    "END" RESVEC;
         "EOP"

"CODE" 31504;
    "PROCEDURE" SYMRESVEC(LR, UR, LC, UC, A, B, C, X);
    "VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
    "REAL" X; "ARRAY" A, B, C;
    "BEGIN" "REAL" "PROCEDURE" SYMMATVEC(L, U, I, A, B);
        "CODE" 34018;
        "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
       C[LR]:= SYMMATVEC(LC, UC, LR, A, B) + C[LR] * X
    "END" SYMRESVEC;
         "EOP"
```

2-nd REVISION, 1977

MC

SECTION : 1.1.4.3 (JANUARY 1976) PAGE 1

AUTHOR: D.T.WINTER.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 751208.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS PROCEDURES THAT MULTIPLY A GIVEN MATRIX BY A (GENERALIZED) HOUSEHOLDER MATRIX, I.E. A MATRIX M = I + X * U * U', WHERE I IS THE UNIT MATRIX, X A REAL CONSTANT AND U A VECTOR (CALLED THE HOUSEHOLDER CONSTANT AND HOUSEHOLDER VECTOR, RESPECTIVELY)

HSHVECMAT	PREMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED IN A ONE-DIMENSIONAL ARRAY.
HSHCOLMAT	PREMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED AS A COLUMN IN A TWO-DIMENSIONAL ARRAY.
HSHROWMAT	PREMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED AS A ROW IN A TWO-DIMENSIONAL ARRAY.
HSHVECTAM	POSTMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED IN A ONE-DIMENSIONAL ARRAY.
HSHCOLTAM	POSTMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED AS A COLUMN IN A TWO-DIMENSIONAL ARRAY.
HSHROWTAM	POSTMULTIPLIES A MATRIX BY A HOUSEHOLDER MATRIX, THE HOUSEHOLDER VECTOR HAS BEEN STORED AS A ROW IN A TWO-DIMENSIONAL ARRAY.

KEYWORDS:

HOUSEHOLDER MATRIX
ORTHOGONAL TRANSFORMATION

LANGUAGE: COMPASS

SUBSECTION: HSHVECMAT

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHVECMAT(LR, UR, LC, UC, X, U, A);
"VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" U, A;
"CODE" 31070;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,UR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,UC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[LR:UR];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:UC];
THE MATRIX TO BE PREMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
TAMVEC = CP34012
ELMCOLVEC = CP34022

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SUBSECTION: HSHCOLMAT

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHCOLMAT(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"CODE" 31071;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,UR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,JC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
I: <ARITHMETIC EXPRESSION>;
THE COLUMN INDEX OF THE HOUSEHOLDER VECTOR;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[LR:UR,I:I];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:JC];
THE MATRIX TO BE PREMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
TAMMAT = CP34014
ELMCOL = CP34023

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SUBSECTION: HSHROWMAT

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHROWMAT(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"CODE" 31072;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,UR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,UC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
I: <ARITHMETIC EXPRESSION>;
THE ROW INDEX OF THE HOUSEHOLDER VECTOR;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[I:I,LR:UR];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:UC];
THE MATRIX TO BE PREMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
MATMAT = CP34013
ELMCOLROW = CP34027

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SUBSECTION: HSHVECTAM

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHVECTAM(LR, UR, LC, UC, X, U, A);
"VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" U, A;
"CODE" 31073;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,JR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,JC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[LC:JC];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:JR,LC:JC];
THE MATRIX TO BE POSTMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
MATVEC = CP34011
ELMROWVEC = CP34027

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SUBSECTION: HSHCOLTAM

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHCOLTAM(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"CODE" 31074;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,UR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,UC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
I: <ARITHMETIC EXPRESSION>;
THE COLUMN INDEX OF THE HOUSEHOLDER VECTOR;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[LC:UC,I:I];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:UC];
THE MATRIX TO BE POSTMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
MATMAT = CP34013
ELMROWCOL = CP34028

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SUBSECTION: HSHROWTAM

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHROWTAM(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"CODE" 31075;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR,UR: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER ROW INDICES;
LC,UC: <ARITHMETIC EXPRESSIONS>;
THE LOWER AND UPPER COLUMN INDICES;
I: <ARITHMETIC EXPRESSION>;
THE ROW INDEX OF THE HOUSEHOLDER VECTOR;
X: <ARITHMETIC EXPRESSION>;
THE HOUSEHOLDER CONSTANT;
U: <ARRAY IDENTIFIER>; "ARRAY" U[I:I,LC:UC];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:UC];
THE MATRIX TO BE POSTMULTIPLIED BY THE HOUSEHOLDER MATRIX.

PROCEDURES USED:
MATTAM = CP34015
ELMROW = CP34024

RUNNING TIME: PROPORTIONAL TO (UC-LC+1)*(UR-LR+1)

SOURCE TEXTS:

```
"CODE" 31070;
"PROCEDURE" HSHVECMAT(LR, UR, LC, UC, X, U, A);
"VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" TAMVEC(L, U, I, A, B); "CODE" 34012;
    "PROCEDURE" ELMCOLVEC(L, U, I, A, B, X); "CODE" 34022;
    "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
    ELMCOLVEC(LR, UR, LC, A, U, TAMVEC(LR, UR, LC, A, U) * X)
"END";
        "EOP"

"CODE" 31071;
"PROCEDURE" HSHCOLMAT(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" TAMMAT(L, U, I, J, A, B); "CODE" 34014;
    "PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "CODE" 34023;
    "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
    ELMCOL(LR, UR, LC, I, A, U, TAMMAT(LR, UR, LC, I, A, U) * X)
"END";
        "EOP"
```

```
"CODE" 31072;
"PROCEDURE" HSHROWMAT(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "CODE" 34013;
    "PROCEDURE" ELMCOLROW(L, U, I, J, A, B, X); "CODE" 34029;
    "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
    ELMCOLROW(LR, UR, LC, I, A, U, MATMAT(LR, UR, I, LC, U, A) * X)
"END";
          "EOP"

"CODE" 31073;
"PROCEDURE" HSHVECTAM(LR, UR, LC, UC, X, U, A);
"VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" MATVEC(L, U, I, A, B); "CODE" 34011;
    "PROCEDURE" ELMROWVEC(L, U, I, A, B, X); "CODE" 34027;
    "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
    ELMROWVEC(LC, UC, LR, A, U, MATVEC(LC, UC, LR, A, U) * X)
"END";
          "EOP"

"CODE" 31074;
"PROCEDURE" HSHCOLTAM(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "CODE" 34013;
    "PROCEDURE" ELMROWCOL(L, U, I, J, A, B, X); "CODE" 34028;
    "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
    ELMROWCOL(LC, UC, LR, I, A, U, MATMAT(LC, UC, LR, I, A, U) * X)
"END";
          "EOP"

"CODE" 31075;
"PROCEDURE" HSHROWTAM(LR, UR, LC, UC, I, X, U, A);
"VALUE" LR, UR, LC, UC, I, X; "INTEGER" LR, UR, LC, UC, I;
"REAL" X; "ARRAY" U, A;
"BEGIN" "REAL" "PROCEDURE" MATTAM(L, U, I, J, A, B); "CODE" 34015;
    "PROCEDURE" ELMROW(L, U, I, J, A, B, X); "CODE" 34024;
    "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
    ELMROW(LC, UC, LR, I, A, U, MATTAM(LC, UC, LR, I, A, U) * X)
"END";
          "EOP"
```

AUTHORS: T.J.DEKKER, C.G. VAN DER LAAN,

CONTRIBUTOR: P.A.BEENTJES,

INSTITUTE: MATHEMATICAL CENTRE,

RECEIVED: 730715,

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TEN PROCEDURES.
ELMVEC ADDS X TIMES THE VECTOR GIVEN IN ARRAY B[SHIFT+L : SHIFT+U] TO THE VECTOR GIVEN IN ARRAY A[L:U].
ELMCOL ADDS X TIMES THE COLUMN VECTOR GIVEN IN ARRAY B[L:U, J:J] TO THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, I:I].
ELMROW ADDS X TIMES THE ROW VECTOR GIVEN IN ARRAY B[J:J, L:U] TO THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U].
ELMVECCOL ADDS X TIMES THE COLUMN VECTOR GIVEN IN ARRAY B[L:U, I:I] TO THE VECTOR GIVEN IN ARRAY A[L:U].
ELMCOLVEC ADDS X TIMES THE VECTOR GIVEN IN ARRAY B[L:U] TO THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, I:I].
ELMVECROW ADDS X TIMES THE ROW VECTOR GIVEN IN ARRAY B[I:I, L:U] TO THE VECTOR GIVEN IN ARRAY A[L:U].
ELMROWVEC ADDS X TIMES THE VECTOR GIVEN IN ARRAY B[L:U] TO THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U].
ELMCOLROW ADDS X TIMES THE ROW VECTOR GIVEN IN ARRAY B[J:J, L:U] TO THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, I:I].
ELMROWCOL ADDS X TIMES THE COLUMN VECTOR GIVEN IN ARRAY B[L:U, J:J] TO THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U].
MAXELMROW ADDS X TIMES THE ROW VECTOR GIVEN IN ARRAY B[J:J, L:U] TO THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U].
MOREOVER, MAXELMROW:= THE VALUE OF THE SECOND SUBSCRIPT OF AN ELEMENT OF THE NEW ROW VECTOR IN ARRAY A WHICH IS OF MAXIMUM ABSOLUTE VALUE.
IF, HOWEVER, L > U, THEN MAXELMROW:= L.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
ELIMINATION,

SUBSECTION: ELMVEC.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" ELMVEC(L, U, SHIFT, A, B, X); "VALUE" L,U,SHIFT,X;
"INTEGER" L,U,SHIFT; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
SHIFT: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETER OF THE VECTOR B;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L (+ SHIFT) AND
U(+SHIFT) SHOULD NOT CONTRADICT EACH OTHER;
X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ELMCOL.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
"INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR A;
J: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR B;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4,J3:J4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
NOT CONTRADICT EACH OTHER;
X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:
SEE REFERENCE [1].

SUBSECTION: ELMROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ELMROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
"INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
```

FORMAL PARAMETERS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR A;

J: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR B;

A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2], B[I3:I4,J3:J4];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD NOT CONTRADICT EACH OTHER;

X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ELMVECCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ELMVECCOL(L, U, I, A, B, X); "VALUE" L,U,I,X;
"INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;
```

FORMAL PARAMETERS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR B;

A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2], B[I3:I4, J1:J2];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U AND I SHOULD NOT CONTRADICT EACH OTHER;

X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ELMCOLVEC.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ELMCOLVEC(L, U, I, A, B, X); "VALUE" L,U,I,X;
"INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:    <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:      <ARITHMETIC EXPRESSION>;
        COLUMN-INDEX OF THE COLUMN VECTOR A;
A,B:    <ARRAY IDENTIFIER>;
        "ARRAY" A[I1:I2, J1:J2], B[I3:I4];
        THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
        NOT CONTRADICT EACH OTHER;
X:      <ARITHMETIC EXPRESSION>;
        ELIMINATION FACTOR.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ELMVECROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ELMVECROW(L, U, I, A, B, X); "VALUE" L,U,I,X;
"INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2], B[I3:I4, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
         NOT CONTRADICT EACH OTHER;
X:       <ARITHMETIC EXPRESSION>;
         ELIMINATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: ELMROWVEC.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ELMROWVEC(L, U, I, A, B, X); "VALUE" L,U,I,X;
    "INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;

    FORMAL PARAMETERS:
    L,U:     <ARITHMETIC EXPRESSION>;
             LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
    I:       <ARITHMETIC EXPRESSION>;
             ROW-INDEX OF THE ROW VECTOR A;
    A,B:     <ARRAY IDENTIFIER>;
             "ARRAY" A[I1:I2, J1:J2], B[I3:I4];
             THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U  AND  I  SHOULD
             NOT CONTRADICT EACH OTHER;
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: ELMCOLROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ELMCOLROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
"INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:       <ARITHMETIC EXPRESSION>;
         COLUMN-INDEX OF THE COLUMN VECTOR A;
J:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2], B[I3:I4,J3:J4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER, WHEN A = B THEN CORRECT
         ELIMINATION IS GUARANTEED ONLY WHEN THE ROW AND COLUMN ARE
         DISJUNCT;
X:       <ARITHMETIC EXPRESSION>;
         ELIMINATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: ELMROWCOL.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ELMROWCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
    "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

    FORMAL PARAMETERS:
    L,U:     <ARITHMETIC EXPRESSION>;
             LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
    I:       <ARITHMETIC EXPRESSION>;
             ROW-INDEX OF THE ROW VECTOR A;
    J:       <ARITHMETIC EXPRESSION>;
             COLUMN-INDEX OF THE COLUMN VECTOR B;
    A,B:     <ARRAY IDENTIFIER>;
             "ARRAY" A[I1:I2, J1:J2], B[I3:I4,J3:J4];
             THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
             NOT CONTRADICT EACH OTHER, WHEN A = B THEN CORRECT
             ELIMINATION IS GUARANTEED ONLY WHEN THE ROW AND COLUMN ARE
             DISJUNCT;
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: ALGOL 60.

SUBSECTION: MAXELMROW.

CALLING SEQUENCE:

```
HEADING:
"INTEGER" "PROCEDURE" MAXELMROW(L, U, I, J, A, B, X);
"VALUE" L,U,I,J,X; "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR A;
J:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR B;
A,B:     <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J21, B[I3:I4,J3:J4];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER;
X:       <ARITHMETIC EXPRESSION>;
         ELIMINATION FACTOR.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

REFERENCES:

[1].T.J.DEKKER,
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART1,
MATHEMATICAL CENTRE TRACT 22, AMSTERDAM (1970).

SOURCE TEXT(S):

```
IN THIS SECTION THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS 3,
     ELMVEC
     ELMROW
     ELMVECCOL
     ELMCOLVEC
     MAXELMROW
THE ALGOL SOURCE TEXT OF THE COMPASS ROUTINES IS GIVEN,

"CODE" 34020;
     "PROCEDURE" ELMVEC(L, U, SHIFT, A, B, X); "VALUE" L,U,SHIFT,X;
     "INTEGER" L,U,SHIFT; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= A[L] + B[L + SHIFT] * X;
          "EOP"

"CODE" 34023;
     "PROCEDURE" ELMCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
     "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,I]:= A[L,I] + B[L,J] * X;
          "EOP"

"CODE" 34024;
     "PROCEDURE" ELMROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
     "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= A[I,L] + B[J,L] * X;
          "EOP"

"CODE" 34021;
     "PROCEDURE" ELMVECCOL(L, U, I, A, B, X); "VALUE" L,U,I,X;
     "INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= A[L] + B[L,I] * X;
          "EOP"

"CODE" 34022;
     "PROCEDURE" ELMCOLVEC(L, U, I, A, B, X); "VALUE" L,U,I,X;
     "INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,I]:= A[L,I] + B[L] * X;
          "EOP"

"CODE" 34026;
     "PROCEDURE" ELMVECROW(L, U, I, A, B, X); "VALUE" L,U,I,X;
     "INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L]:= A[L] + B[I,L] * X;
          "EOP"
```

```
"CODE" 34027;
    "PROCEDURE" ELMROWVEC(L, U, I, A, B, X); "VALUE" L,U,I,X;
    "INTEGER" L,U,I; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= A[I,L] + B[L] * X;
        "EOP"

"CODE" 34029;
    "PROCEDURE" ELMCOLROW(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
    "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[L,I]:= A[L,I] + B[J,L] * X;
        "EOP"

"CODE" 34028;
    "PROCEDURE" ELMROWCOL(L, U, I, J, A, B, X); "VALUE" L,U,I,J,X;
    "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" A[I,L]:= A[I,L] + B[L,J] * X;
        "EOP"

"CODE" 34025;
    "INTEGER" "PROCEDURE" MAXELMROW(L, U, I, J, A, B, X);
    "VALUE" L,U,I,J,X; "INTEGER" L,U,I,J; "REAL" X; "ARRAY" A,B;
    "BEGIN" "INTEGER" K; "REAL" R, S;
        S:= 0;
        "FOR" K:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[I,K]:= A[I,K] + B[J,K] * X;"IF" ABS(R) > S "THEN"
            "BEGIN" S:= ABS(R); L:= K "END"
        "END";
        MAXELMROW:= L
    "END" MAXELMROW;
        "EOP"
```

AUTHOR: T.J.DEKKER.

CONTRIBUTOR: P.A.BEENTJES.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SIX PROCEDURES.
ICHVEC INTERCHANGES THE ELEMENTS OF THE VECTOR GIVEN IN ARRAY A[L:U] AND ARRAY A[SHIFT + L : SHIFT + U].
ICHCOL INTERCHANGES THE ELEMENTS OF THE COLUMN VECTORS GIVEN IN ARRAY A[L:U, I:I] AND ARRAY A[L:U, J:J].
ICHROW INTERCHANGES THE ELEMENTS OF THE ROW VECTORS GIVEN IN ARRAY A[I:I, L:U] AND ARRAY A[J:J, L:U].
ICHROWCOL INTERCHANGES THE ELEMENTS OF THE ROW VECTOR GIVEN IN ARRAY A[I:I, L:U] AND THE COLUMN VECTOR GIVEN IN ARRAY A[L:U, J:J].
ICHSEQVEC INTERCHANGES THE ELEMENTS OF THE VECTORS GIVEN IN ARRAY A[IL : IL + (U + L - 1)*(U - L)//2] AND ARRAY A[SHIFT+L : SHIFT+U], WHERE THE ELEMENTS OF THE FIRST VECTOR ARE A[IL+(J+L-1)*(J-L)//2] FOR J = L,...., U.
ICHSEQ INTERCHANGES THE ELEMENTS OF THE VECTORS GIVEN IN ARRAY A[IL : IL + (U + L - 1) * (U - L) // 2] AND ARRAY A[SHIFT + IL : SHIFT + IL + (U + L - 1) * (U - L) // 2] WHERE THE ELEMENTS OF THE VECTORS ARE A[IL + (J + L - 1) * (J - L) // 2] AND A[SHIFT + IL + (J + L - 1) * (J - L) // 2] FOR J = L ,...., U .

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
INTERCHANGING.

SUBSECTION: ICHVEC.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHVEC(L, U, SHIFT, A); "VALUE" L,U,SHIFT;
"INTEGER" L,U,SHIFT; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
SHIFT:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETER;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2];
         THE SUBSCRIPTS ABOVE AND THE VALUES  OF  L ( + SHIFT )  AND
         U(+SHIFT) SHOULD NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ICHCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHCOL(L, U, I, J, A); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J:     <ARITHMETIC EXPRESSION>;
         COLUMN-INDICES   OF   THE   COLUMN   VECTORS   OF   ARRAY   A;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ICHROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHROW(L, U, I, J, A); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J:     <ARITHMETIC EXPRESSION>;
         ROW-INDICES OF THE ROW VECTORS OF ARRAY A;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: ICHROWCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHROWCOL(L, U, I, J, A); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:       <ARITHMETIC EXPRESSION>;
         ROW-INDEX OF THE ROW VECTOR OF ARRAY A;
J:       <ARITHMETIC EXPRESSION>;
         COLUMN-INDEX OF THE COLUMN VECTOR OF ARRAY A;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER, FURTHERMORE THE ROW AND COLUMN
         TO BE INTERCHANGED SHOULD BE DISJUNCT;
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE REFERENCE [1].

SUBSECTION: ICHSEQVEC.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHSEQVEC(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
"INTEGER" L,U,IL,SHIFT; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
IL:      <ARITHMETIC EXPRESSION>;
         LOWER BOUND OF THE VECTOR A;
SHIFT:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETER;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L(+SHIFT), U(+SHIFT)
         AND IL(+(U+L-1)*(U-L)//2) SHOULD NOT CONTRADICT EACH OTHER.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ICHSEQ.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHSEQ(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
"INTEGER" L,U,IL,SHIFT; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
IL:      <ARITHMETIC EXPRESSION>;
         LOWER BOUND OF THE VECTOR A;
SHIFT:   <ARITHMETIC EXPRESSION>;
         INDEX-SHIFTING PARAMETER;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF  IL+(J+L-1)*(J-L)//2
         ( +SHIFT ),J = L(1)U, SHOULD  NOT  CONTRADICT  EACH  OTHER.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

REFERENCES:

[1].T.J.DEKKER.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART1,
MATHEMATICAL CENTRE TRACT 22, AMSTERDAM (1970).

SOURCE TEXT(S):

```
"CODE" 34030;
    "PROCEDURE" ICHVEC(L, U, SHIFT, A); "VALUE" L,U,SHIFT;
    "INTEGER" L,U,SHIFT; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[L]; A[L]:= A[L + SHIFT]; A[L + SHIFT]:= R "END"
    "END" ICHVEC;
        "EOP"

"CODE" 34031;
    "PROCEDURE" ICHCOL(L, U, I, J, A); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[L,I]; A[L,I]:= A[L,J]; A[L,J]:= R "END"
    "END" ICHCOL;
        "EOP"

"CODE" 34032;
    "PROCEDURE" ICHROW(L, U, I, J, A); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[I,L]; A[I,L]:= A[J,L]; A[J,L]:= R "END"
    "END" ICHROW;
        "EOP"
```

```
"CODE" 34033;
    "PROCEDURE" ICHROWCOL(L, U, I, J, A); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[I,L]; A[I,L]:= A[L,J]; A[L,J]:= R "END"
    "END" ICHROWCOL;
        "EOP"

"CODE" 34034;
    "PROCEDURE" ICHSEQVEC(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
    "INTEGER" L,U,IL,SHIFT; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[IL]; A[IL]:= A[L + SHIFT]; A[L + SHIFT]:= R;
            IL:= IL + L
        "END"
    "END" ICHSEQVEC;
        "EOP"

"CODE" 34035;
    "PROCEDURE" ICHSEQ(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
    "INTEGER" L,U,IL,SHIFT; "ARRAY" A;
    "BEGIN" "REAL" R;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" R:= A[IL]; A[IL]:= A[IL + SHIFT]; A[IL + SHIFT]:= R;
            IL:= IL + L
        "END"
    "END" ICHSEQ;
        "EOP"
```

AUTHOR: T.J.DEKKER.

CONTRIBUTOR: P.A.BEENTJES.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730715.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES.
ROTCOL REPLACES THE COLUMN VECTOR X GIVEN IN ARRAY A[L:U, I:I] AND THE COLUMN VECTOR Y GIVEN IN ARRAY A[L:U, J:J] BY THE VECTORS CX + SY AND CY - SX.
ROTROW REPLACES THE ROW VECTOR X GIVEN IN ARRAY A[I:I, L:U] AND THE ROW VECTOR Y GIVEN IN ARRAY A[J:J, L:U] BY THE VECTORS CX + SY AND CY - SX.

KEYWORDS:

ELEMENTARY PROCEDURE,
VECTOR OPERATIONS,
ROTATION.

SUBSECTION: ROTCOL.

CALLING SEQUENCE:

HEADING:
"PROCEDURE" ROTCOL(L, U, I, J, A, C, S); "VALUE" L,U,I,J,C,S;
"INTEGER" L,U,I,J; "REAL" C,S; "ARRAY" A;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J: <ARITHMETIC EXPRESSION>;
COLUMN-INDICES OF THE COLUMN VECTORS OF ARRAY A;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[I1:I2, J1:J2];
THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD NOT CONTRADICT EACH OTHER;
C,S: <ARITHMETIC EXPRESSION>;
ROTATION FACTORS.

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

SUBSECTION: ROTROW.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ROTROW(L, U, I, J, A, C, S); "VALUE" L,U,I,J,C,S;
"INTEGER" L,U,I,J; "REAL" C,S; "ARRAY" A;

FORMAL PARAMETERS:
L,U:     <ARITHMETIC EXPRESSION>;
         LOWER AND UPPER BOUND OF THE RUNNING SUBSCRIPT;
I,J:     <ARITHMETIC EXPRESSION>;
         ROW-INDICES OF THE ROW-VECTORS OF ARRAY A;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[I1:I2, J1:J2];
         THE SUBSCRIPTS ABOVE AND THE VALUES OF L, U, I AND J SHOULD
         NOT CONTRADICT EACH OTHER;
C,S:     <ARITHMETIC EXPRESSION>;
         ROTATION FACTORS.
```

LANGUAGE: COMPASS 3.

METHOD AND PERFORMANCE:

SEE REFERENCE [1].

REFERENCES:

[1].T.J.DEKKER.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART1.
MATHEMATICAL CENTRE TRACT 22, AMSTERDAM (1970).

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS 3,
THE ALGOL SOURCE TEXT OF THE COMPASS ROUTINES IS GIVEN.

```
"CODE" 34040;
    "PROCEDURE" ROTCOL(L, U, I, J, A, C, S); "VALUE" L,U,I,J,C,S;
    "INTEGER" L,U,I,J; "REAL" C,S; "ARRAY" A;
    "BEGIN" "REAL" X, Y;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" X:= A[L,I]; Y:= A[L,J]; A[L,I]:= X * C + Y * S;
            A[L,J]:= Y * C - X * S
        "END"
    "END" ROTCOL;
        "EOP"

"CODE" 34041;
"PROCEDURE" ROTROW(L, U, I, J, A, C, S); "VALUE" L,U,I,J,C,S;
"INTEGER" L,U,I,J; "REAL" C,S; "ARRAY" A;
"BEGIN" "REAL" X, Y;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    "BEGIN" X:= A[I,L]; Y:= A[J,L]; A[I,L]:= X * C + Y * S;
        A[J,L]:= Y * C - X * S
    "END"
"END" ROTROW;
    "EOP"
```

AUTHORS: C.G. VAN DER LAAN AND J.C.P. BUS.

CONTRIBUTOR: J.C.P. BUS.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 740921.

BRIEF DESCRIPTION:

INFNRMVEC CALCULATES THE INFINITY-NORM OF A VECTOR;
INFNRMROW CALCULATES THE INFINITY-NORM OF A ROW VECTOR;
INFNRMCOL CALCULATES THE INFINITY-NORM OF A COLUMN VECTOR;
INFNRMMAT CALCULATES THE INFINITY-NORM OF A MATRIX;
ONENRMVEC CALCULATES THE ONE-NORM OF A VECTOR;
ONENRMROW CALCULATES THE ONE-NORM OF A ROW VECTOR;
ONENRMCOL CALCULATES THE ONE-NORM OF A COLUMN VECTOR;
ONENRMMAT CALCULATES THE ONE-NORM OF A MATRIX;
ABSMAXMAT CALCULATES FOR A GIVEN MATRIX THE MODULUS OF AN ELEMENT WHICH IS OF MAXIMUM ABSOLUTE VALUE;

KEYWORDS:

VECTOR NORMS,
MATRIX NORMS.

SUBSECTION: INFNRMVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" INFNRMVEC(L, U, K, A);
"VALUE" L, U; "INTEGER" L, U, K; "ARRAY" A;

INFNRMVEC := MAX( ABS(A[I]), I= L, ..., U );

THE MEANING OF THE FORMAL PARAMETERS IS:
L, U:    <ARITHMETIC EXPRESSION>;
         ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE
               VECTOR A RESPECTIVELY;
K:       <VARIABLE>;
         EXIT:THE FIRST INDEX FOR WHICH ABS(A[I]), I = L, ..., U,
              IS MAXIMAL;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[L:U].
```

PROCEDURES USED: NONE.

SUBSECTION: INFNRMROW.

CALLING SEQJENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL" "PROCEDURE" INFNRMROW(L, U, I, K, A);
    "VALUE" L, U, I; "INTEGER" L, U, I, K; "ARRAY" A;

    INFNRMROW := MAX( ABS(A[I,J]), J= L, ..., U );

    THE MEANING OF THE FORMAL PARAMETERS IS:
    L, U:    <ARITHMETIC EXPRESSION>;
             ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE
                   ROW VECTOR A RESPECTIVELY;
    I:       <ARITHMETIC EXPRESSION>;
             ENTRY:THE ROW INDEX;
    K:       <VARIABLE>;
             EXIT:THE FIRST INDEX FOR WHICH ABS(A[I,J]), J = L, ..., U,
                  IS MAXIMAL;
    A:       <ARRAY IDENTIFIER>;
             "ARRAY" A[I:I,L:U].
```

PROCEDURES USED: NONE.

SUBSECTION: INFNRMCOL.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" INFNRMCOL(L, U, J, K, A);
"VALUE" L, U, J; "INTEGER" L, U, J, K; "ARRAY" A;

INFNRMCOL := MAX( ABS(A[I,J]), I= L, ..., U );

THE MEANING OF THE FORMAL PARAMETERS IS:
L, U:    <ARITHMETIC EXPRESSION>;
         ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE
               COLUMN VECTOR A RESPECTIVELY;
J:       <ARITHMETIC EXPRESSION>;
         ENTRY:THE COLUMN INDEX;
K:       <VARIABLE>;
         EXIT:THE FIRST INDEX FOR WHICH ABS(A[I,J]), I = L, ..., U,
              IS MAXIMAL;
A:       <ARRAY IDENTIFIER>;
         "ARRAY" A[L:U,J:J].
```

PROCEDURES USED: NONE.

SUBSECTION: INFNRMMAT.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL" "PROCEDURE" INFNRMMAT(LR, UR, LC, UC, KR, A);
    "VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, KR; "ARRAY" A;

    INFNRMMAT := MAX( ONENRMROW(LC, UC, I, A), I=LR, ..., UR );

    THE MEANING OF THE FORMAL PARAMETERS IS:
    LR, UR: <ARITHMETIC EXPRESSION>;
            ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE ROW INDEX,
                  RESPECTIWELY;
    LC, UC: <ARITHMETIC EXPRESSION>;
            ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE COLUMN INDEX,
                  RESPECTIWELY;
    KR:     <VARIABLE>;
            EXIT:THE FIRST ROW INDEX FOR WHICH THE ONE-NORM IS MAXIMAL;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[LR:UR,LC:UC].
```

PROCEDURES JSED: ONENRMROW.

SUBSECTION: ONENRMVEC.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL" "PROCEDURE" ONENRMVEC(L, U, A);
    "VALUE" L, U; "INTEGER" L, U; "ARRAY" A;

    ONENRMVEC := SUM( ABS(A[I]), I= L, ..., U );

    THE MEANING OF THE FORMAL PARAMETERS IS:
    L, U:    <ARITHMETIC EXPRESSION>;
             ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE
                   VECTOR A RESPECTIVELY;
    A:       <ARRAY IDENTIFIER>;
             "ARRAY" A[L:U].
```

PROCEDURES USED: NONE.

SUBSECTION: ONENRMROW.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL" "PROCEDURE" ONENRMROW(L, U, I, A);
    "VALUE" L, U, I; "INTEGER" L, U, I; "ARRAY" A;

    ONENRMROW := SUM( ABS(A[I,J]), J= L, ..., U );

    THE MEANING OF THE FORMAL PARAMETERS IS:
    L, U:    <ARITHMETIC EXPRESSION>;
             ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE
                   ROW VECTOR A RESPECTIVELY;
    I:       <ARITHMETIC EXPRESSION>;
             ENTRY: THE ROW INDEX;
    A:       <ARRAY IDENTIFIER>;
             "ARRAY" A[I:I,L:U].
```

PROCEDURES USED: NONE.

SUBSECTION: ONENRMCOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" ONENRMCOL(L, U, J, A);
"VALUE" L, U, J; "INTEGER" L, U, J; "ARRAY" A;

ONENRMCOL := SUM(ABS(A[I,J]), I= L, ..., U);

THE MEANING OF THE FORMAL PARAMETERS IS:
L, U: <ARITHMETIC EXPRESSION>;
ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE INDICES OF THE COLUMN VECTOR A RESPECTIVELY;
J: <ARITHMETIC EXPRESSION>;
ENTRY: THE COLUMN INDEX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[L:U,J:J].

PROCEDURES USED: NONE.

SUBSECTION: ONENRMMAT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" ONENRMMAT(LR, UR, LC, UC, KC, A);
"VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, KC; "ARRAY" A;

ONENRMMAT := MAX(ONENRMCOL(LR, UR, J, A), J=LC, ..., UC);

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE ROW INDEX, RESPECTIVELY;
LC, UC: <ARITHMETIC EXPRESSION>;
ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE COLUMN INDEX, RESPECTIVELY;
KC: <VARIABLE>;
EXIT:THE FIRST COLUMN INDEX FOR WHICH THE ONE-NORM IS MAXIMAL;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC].

PROCEDURES USED: ONENRMCOL.

SUBSECTION: ABSMAXMAT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" ABSMAXMAT(LR, UR, LC, UC, KR, KC, A);
"VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, KR, KC;
"ARRAY" A;

ABSMAXMAT := MAX(ABS(A[I,J]), I= LR, ..., UR, J= LC, ..., JC);

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE ROW INDEX,
RESPECTIVELY;
LC, UC: <ARITHMETIC EXPRESSION>;
ENTRY:THE LOWER BOUND AND UPPER BOUND OF THE COLUMN INDEX,
RESPECTIVELY;
KR, KC: <VARIABLE>;
EXIT:THE ROW AND COLUMN INDEX OF AN ELEMENT FOR WHICH THE
MODULUS IS MAXIMAL;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC].

PROCEDURES USED: INFNRMCOL.

LANGUAGE: COMPASS.

METHOD AND PERFORMANCE:

ELEMENTARY.

SOURCE TEXT(S):

```
"CODE" 31061;
 "REAL" "PROCEDURE" INFNRMVEC(L, U, K, A); "VALUE" L, U;
 "INTEGER" L, U, K; "ARRAY" A;
 "BEGIN" "REAL" R, MAX;
    MAX:= 0; K:= L;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    "BEGIN" R:= ABS(A[L]); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; K:= L "END"
    "END";
    INFNRMVEC:= MAX
"END" INFNRMVEC;
        "EOP"

"CODE" 31062;
 "REAL" "PROCEDURE" INFNRMROW(L, U, I, K, A); "VALUE" L, U, I;
 "INTEGER" L, U, I, K; "ARRAY" A;
 "BEGIN" "REAL" R, MAX;
    MAX:= 0; K:= L;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    "BEGIN" R:= ABS(A[I,L]); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; K:= L "END"
    "END";
    INFNRMROW:= MAX
"END" INFNRMROW;
        "EOP"

"CODE" 31063;
 "REAL" "PROCEDURE" INFNRMCOL(L, U, J, K, A); "VALUE" L, U, J;
 "INTEGER" L, U, J, K; "ARRAY" A;
 "BEGIN" "REAL" R, MAX;
    MAX:= 0; K:= L;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    "BEGIN" R:= ABS(A[L,J]); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; K:= L "END"
    "END";
    INFNRMCOL:= MAX
"END" INFNRMCOL;
        "EOP"
```

```
"CODE" 31064;
 "REAL" "PROCEDURE" INFNRMMAT(LR, UR, LC, UC, KR, A);
 "VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, KR; "ARRAY" A;
 "BEGIN" "REAL" R, MAX;
    "REAL" "PROCEDURE" ONENRMROW(L, U, I, A); "CODE" 31066;
    MAX:= 0; KR:= LR;
    "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
    "BEGIN" R:= ONENRMROW(LC, UC, LR, A); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; KR:= LR "END"
    "END";
    INFNRMMAT:= MAX
"END" INFNRMMAT;
        "EOP"

"CODE" 31065;
 "REAL" "PROCEDURE" ONENRMVEC(L, U, A); "VALUE" L, U;
 "INTEGER" L, U; "ARRAY" A;
 "BEGIN" "REAL" SUM;
    SUM:= 0; "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    SUM:= SUM + ABS(A[L]);
    ONENRMVEC:= SUM
"END" ONENRMVEC;
        "EOP"

"CODE" 31066;
 "REAL" "PROCEDURE" ONENRMROW(L, U, I, A); "VALUE" L, U, I;
 "INTEGER" L, U, I; "ARRAY" A;
 "BEGIN" "REAL" SUM;
    SUM:= 0; "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    SUM:= SUM + ABS(A[I,L]);
    ONENRMROW:= SUM
"END" ONENRMROW;
        "EOP"
```

```
"CODE" 31067;
 "REAL" "PROCEDURE" ONENRMCOL(L, U, J, A); "VALUE" L, U, J;
 "INTEGER" L, U, J; "ARRAY" A;
 "BEGIN" "REAL" SUM;
    SUM:= 0; "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
    SUM:= SUM + ABS(A[L,J]);
    ONENRMCOL:= SUM
"END" ONENRMCOL;
        "EOP"

"CODE" 31068;
 "REAL" "PROCEDURE" ONENRMMAT(LR, UR, LC, UC, KC, A);
 "VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, KC; "ARRAY" A;
 "BEGIN" "REAL" MAX, R;
    "REAL" "PROCEDURE" ONENRMCOL(L, U, J, A); "CODE" 31067;
    MAX:= 0; KC:= LC;
    "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
    "BEGIN" R:= ONENRMCOL(LR, UR, LC, A); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; KC:= LC "END"
    "END";
    ONENRMMAT:= MAX
"END" ONENRMMAT;
        "EOP"

"CODE" 31069;
 "REAL" "PROCEDURE" ABSMAXMAT(LR, UR, LC, UC, I, J, A);
 "VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, I, J; "ARRAY" A;
 "BEGIN" "INTEGER" II; "REAL" MAX, R;
    "REAL" "PROCEDURE" INFNRMCOL(L, U, I, K, A); "CODE" 31063;
    MAX:= 0; I:= LR; J:= LC;
    "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
    "BEGIN" R:= INFNRMCOL(LR, UR, LC, II, A); "IF" R > MAX "THEN"
        "BEGIN" MAX:= R; I:= II; J:= LC "END"
    "END";
    ABSMAXMAT:= MAX
"END" ABSMAXMAT;
        "EOP"
```

AUTHORS : T.J. DEKKER, W. HOFFMANN.

CONTRIBUTORS: W. HOFFMANN, S.P.N. VAN KAMPEN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731030.

BRIEF DESCRIPTION:

THE PROCEDURE REASCL NORMALIZES THE (NON-NULL) COLUMNS OF A TWO-DIMENSIONAL ARRAY IN SUCH A WAY THAT, IN EACH COLUMN, AN ELEMENT OF MAXIMUM ABSOLUTE VALUE EQUALS 1. THE NORMALIZED VECTORS ARE DELIVERED IN THE CORRESPONDING COLUMNS OF THE ARRAY.

KEYWORDS:

NORMALIZATION,
VECTOR SCALING.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" REASCL(A, N, N1, N2); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A;

THE MEANING OF THE FORMAL PARAMETERS IS:
A: <ARRAY IDENTIFIER>;
A TWO-DIMENSIONAL ARRAY A[1:N,N1:N2];
ENTRY: THE N2 - N1 + 1 COLUMN VECTORS MUST BE GIVEN IN A;
EXIT: THE NORMALIZED VECTORS (I.E. IN EACH VECTOR AN ELEMENT OF MAXIMUM ABSOLUTE VALUE EQUALS 1) ARE DELIVERED IN THE CORRESPONDING COLUMNS OF A;
N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF ARRAY A;
N1, N2: <ARITHMETIC EXPRESSION>;
THE LOWER AND UPPER BOUND OF THE COLUMN INDICES OF ARRAY A.

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N * (N2 - N1 + 1).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE REF [1].

REFERENCES:

[1].T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

EXAMPLE OF USE:

THE PROCEDURE REASCL IS USED IN REAEIG1, SECTION 3.3.1.2.2.

SOURCE TEXT(S) :

```
"CODE" 34183;
    "COMMENT" MCA 2413;
    "PROCEDURE" REASCL(A, N, N1, N2); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" A;
    "BEGIN" "INTEGER" I, J; "REAL" S;
        "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO"
        "BEGIN" S:= 0;
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            "IF" ABS(A[I,J]) > ABS(S) "THEN" S:= A[I,J];
            "IF" S "=" 0 "THEN"
            "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO" A[I,J]:= A[I,J] / S
        "END"
    "END" REASCL;
        "EOP"
```

AUTHOR: J. C. P. BUS.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730816.

BRIEF DESCRIPTION:

MAXMAT CALCULATES FOR A GIVEN MATRIX THE INDICES AND MODULUS OF AN ELEMENT, WHICH IS OF MAXIMUM ABSOLUTE VALUE.

KEYWORDS:

MATRIX NORMS.

CALLING SEQUENCE:

```
THE HEADING OF THIS PROCEDURE IS:
"REAL" "PROCEDURE" MAXMAT(LR, UR, LC, UC, I, J, A);
"VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, I, J; "ARRAY" A;

MAXMAT: DELIVERS FOR THE MATRIX, GIVEN IN ARRAY A, THE MODULUS OF
        AN ELEMENT, WHICH IS OF MAXIMUM ABSOLUTE VALUE; IF, HOWEVER
        , LR > UR OR LC > UC, THEN MAXMAT:= 0;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR:     <ARITHMETIC EXPRESSION>;
        THE LOWER BOUND OF THE FIRST SUBSCRIPT OF THE ARRAY A;
UR:     <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE FIRST SUBSCRIPT OF THE ARRAY A;
LC:     <ARITHMETIC EXPRESSION>;
        THE LOWER BOUND OF THE SECOND SUBSCRIPT OF THE ARRAY A;
UC:     <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE SECOND SUBSCRIPT OF THE ARRAY A;
I:      <VARIABLE>;
        EXIT: THE VALUE OF THE FIRST SUBSCRIPT OF THE ELEMENT,
            WHOSE MODULUS IS DELIVERED BY MAXMAT; IF, HOWEVER,
            LR > UR OR LC > UC, THEN I = LR;
    J:  <VARIABLE>;
        EXIT: THE VALUE OF THE SECOND SUBSCRIPT OF THE ELEMENT,
            WHOSE MODULUS IS DELIVERED BY MAXMAT; IF, HOWEVER,
            LR > UR OR LC > UC, THEN J = LC;
    A:  <ARRAY IDENTIFIER>;
        A TWO - DIMENSIONAL ARRAY A[LR : UR, LC : UC];
        THE MATRIX MUST BE GIVEN IN A.
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH: MAXMAT DECLARES NO AUXILIARY ARRAYS.

RUNNING TIME:

PROPORTIONAL TO (UR - LR + 1) * (UC - LC + 1).

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:

```
"BEGIN"
     "REAL" "PROCEDURE" MAXMAT(LR, UR, LC, UC, I, J, A);
     "CODE" 34230;
     "INTEGER" I, J; "REAL" MAX; "ARRAY" A[2:5, 3:4];
     MAX:= 1;
     "FOR" I:= 2, 3, 4 "DO" "FOR" J:= 3, 4 "DO"
     "BEGIN" A[I, J]:= I * J * MAX; MAX:= -MAX "END";
     A[5, 3]:= 16; A[5, 4]:= -10; I:= J:= 0;
     MAX:= MAXMAT(2, 5, 3, 4, I, J, A);
     OUTPUT(61, "("/,+ZZD,/D,3BD")", MAX, I, J)
"END"
```

DELIVERS:

```
 +16
4   4
```

SOURCE TEXT(S):

```
"CODE" 34230;
     "REAL" "PROCEDURE" MAXMAT(LR, UR, LC, UC, I, J, A);
     "VALUE" LR, UR, LC, UC; "INTEGER" LR, UR, LC, UC, I, J; "ARRAY" A;
     "BEGIN" "INTEGER" P, Q, IA, JA; "REAL" MAX, AID;
          MAX:= 0; IA:= LR; JA:= LC;
     "FOR" P:= LR "STEP" 1 "UNTIL" UR "DO"
     "BEGIN" "FOR" Q:= LC "STEP" 1 "UNTIL" UC "DO"
          "BEGIN" AID:= ABS(A[P, Q]); "IF" AID > MAX "THEN"
               "BEGIN" MAX:= AID; IA:= P; JA:= Q "END"
          "END"
     "END";
     MAXMAT:= MAX; I:= IA; J:= JA
"END" MAXMAT;
     "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730928.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES COMCOLCST AND COMROWCST.
COMCOLCST MULTIPLIES THE COMPLEX COLUMN-VECTOR GIVEN IN ARRAY
AR,AI[L:U,J:J] BY XR+I*XI.
COMROWCST MULTIPLIES THE COMPLEX ROW-VECTOR GIVEN IN ARRAY
AR,AI[I:I,L:U] BY XR+I*XI.

KEYWORDS :

COMPLEX VECTOR OPERATIONS,
MULTIPLICATION.

SUBSECTION: COMCOLCST.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);
"VALUE" L,U,J,XR,XI;"INTEGER" L,U,J;"REAL" XR,XI;
"ARRAY" AR,AI;

THE MEANING OF THE FORMAL PARAMETERS IS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE COLUMN VECTOR;

J: <ARITHMETIC EXPRESSION>;
COLUMN-INDEX OF THE COLUMN VECTOR;

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[L:U,J:J]
ENTRY:
AR : REAL PART,
AI : IMAGINARY PART OF THE COLUMN VECTOR
EXIT:
THE TRANSFORMED COMPLEX COLUMN;

XR,XI: <ARITHMETIC EXPRESSION>;
ENTRY:
XR: REAL PART OF THE MULTIPLICATION FACTOR;
XI: IMAGINARY PART OF THE MULTIPLICATION FACTOR.

PROCEDURES USED: COMMUL = CP34341.

RUNNING TIME: ROUGHLY PROPORTIONAL TO (U-L+1).

LANGUAGE: ALGOL 60.

SUBSECTION: COMROWCST.

CALLING SEQJENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" COMROWCST(L, U, I, AR, AI, XR, XI);
"VALUE" L, U, I, XR, XI; "INTEGER" L, U, I; "REAL" XR, XI;
"ARRAY" AR, AI;

THE MEANING OF THE FORMAL PARAMETERS IS:

L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW VECTOR;

I: <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTOR;

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY"AR,AI[I:I,L:U];
ENTRY:
AR : REAL PART,
AI : IMAGINARY PART OF THE ROW VECTOR
EXIT:
THE TRANSFORMED COMPLEX ROW;

XR,XI: <ARITHMETIC EXPRESSION>;
XR: REAL PART OF THE MULTIPLICATION FACTOR;
XI: IMAGINARY PART OF THE MULTIPLICATION FACTOR.

PROCEDURES USED: COMMUL = CP34341.

RUNNING TIME: ROUGHLY PROPORTIONAL TO (U-L).

LANGUAGE: ALGOL 60.

SOURCE TEXT(S) :

```
"CODE" 34352;
    "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);
    "VALUE" L,U,J,XR,XI;"INTEGER" L,U,J;"REAL" XR,XI;
    "ARRAY" AR,AI;
    "BEGIN"
    "PROCEDURE" COMMUL(AR,AI,BR,BI,RR,RI);"CODE" 34341;
    "FOR" L:=L "STEP" 1 "UNTIL" U "DO"
    COMMUL(AR[L,J],AI[L,J],XR,XI,AR[L,J],AI[L,J]);
    "END" COMCOLCST;
        "EOP"

"CODE" 34353;
    "PROCEDURE" COMROWCST(L, U, I, AR, AI, XR, XI);
    "VALUE" L, U, I, XR, XI; "INTEGER" L, U, I; "REAL" XR, XI;
    "ARRAY" AR, AI;
    "BEGIN"
        "PROCEDURE" COMMUL(AR,AI,BR,BI,RR,RI);"CODE" 34341;
    "FOR" L:= L "STEP" 1 "UNTIL" U "DO" COMMUL(AR[I,L], AI[I,L], XR,
    XI, AR[I,L], AI[I,L]);
"END" COMROWCST;
    "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED : 731016.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THREE PROCEDURES:

COMMATVEC CALCULATES THE SCALAR PRODUCT OF A COMPLEX ROWVECTOR GIVEN IN ARRAY AR,AI[I:I,L:U] AND THE COMPLEX VECTOR GIVEN IN ARRAY BR,BI[L:U].
HSHCOMCOL TRANSFORMS A COMPLEX VECTOR INTO A VECTOR PROPORTIONAL TO A UNIT VECTOR;
HSHCOMPRD PREMULTIPLIES A COMPLEX MATRIX WITH A COMPLEX HOUSEHOLDER MATRIX.
HSHCOMCOL AND HSHCOMPRD ARE AUXILIARY PROCEDURES FOR PREMULTIPLYING A COMPLEX MATRIX OR VECTOR WITH A COMPLEX HOUSEHOLDER MATRIX;

KEYWORDS:

COMPLEX SCALAR PRODUCTS,
HOUSEHOLDER TRANSFORMATION

SUBSECTION: COMMATVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" COMMATVEC(L, U, I, AR, AI, BR, BI, RR, RI);
"VALUE" L, U, I; "INTEGER" L, U, I; "REAL" RR, RI;
"ARRAY" AR, AI, BR, BI;

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U : <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE VECTORS;
I : <ARITHMETIC EXPRESSION>;
ROW-INDEX OF THE ROW VECTORS AR AND AI;
AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[I:I,L:U];
ENTRY:
AR: REAL PART AND
AI: IMAGINARY PART OF THE MATRIX;

BR,BI : <ARRAY IDENTIFIER>;
"ARRAY" BR,BI[L:U];
ENTRY:
BR: REAL PART AND
BI: IMAGINARY PART OF THE VECTOR;
RR,RI: <VARIABLE>;
EXIT:
RR: THE REAL PART AND
RI: THE IMAGINARY PART OF THE SCALAR PRODUCT.

PROCEDURES USED:MATVEC=CP34011.

RUNNING TIME: PROPORTIONAL TO U-L.

LANGUAGE: ALGOL 60.

SUBSECTION: HSHCOMCOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"BOOLEAN" "PROCEDURE" HSHCOMCOL(L, U, J, AR, AI, TOL, K, C, S, T);
"VALUE" L, U, J, TOL; "INTEGER" L, U, J; "REAL" TOL, K, C, S, T;
"ARRAY" AR, AI;

HSHCOMCOL DELIVERS THE FOLLOWING BOOLEAN VALUE:
IF AR[L+1,J]**2+AI[L+1,J]**2+...+AR[U,J]**2+AI[U,J]**2>TOL THEN
A TRANSFORMATION IS PERFORMED AND HSHCOMCOL:="TRUE",OTHERWISE
HSHCOMCOL:="FALSE" AND THE VECTOR TO BE TRANSFORMED IS CONSIDERED
TO BE PROPORTIONAL TO THE DESIRED UNIT VECTOR AND NO
TRANSFORMATION IS PERFORMED.

THE MEANING OF THE FORMAL PARAMETERS IS:
L,U,J: <ARITHMETIC EXPRESSION>;
THE COMPLEX VECTOR TO BE TRANSFORMED, MUST BE GIVEN IN
THE J-TH COLUMN FROM ROW L UNTIL ROW U OF A COMPLEX
MATRIX;
AR,AI: <ARRAY IDENTIIER>;
"ARRAY" AR,AI[L:U,J:J];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE VECTOR TO BE
TRANSFORMED MUST BE GIVEN IN THE ARRAYS AR AND AI,
RESPECTIVELY;
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE VECTOR U,
OF THE HOUSEHOLDER MATRIX I-UU"/T (WHERE " DENOTES
CONJUGATING AND TRANSPOSING)ARE DELIVERED IN THE ARRAYS
AR AND AI,RESPECTIVELY,PROVIDED A TRANSFORMATION IS
PERFORMED,IF NO TRANSFORMATION IS PERFORMED THE ARRAYS
AR AND AI ARE UNALTERED;

TOL: <ARITHMETIC EXPRESSION>;
ENTRY: A TOLERANCE;
(E.G. THE SQUARE OF THE MACHINE PRECISION TIMES A NORM
OF THE MATRIX IN CONSIDERATION);
T: <ARITHMETIC EXPRESSION>;
EXIT:
INFORMATION CONCERNING THE TRANSFORMATION,I.E. THE SCALAR
T OF THE HOUSEHOLDER MATRIX ,PROVIDED A TRANSFORMATION IS
PERFORMED.OTHERWISE,T:=-1;
K,C,S: <VARIABLE>;
EXIT:
THE MODULUS , COSINE AND SINE OF THE ARGUMENT OF THE
FIRST ELEMENT OF THE TRANSFORMED VECTOR ARE DELIVERED IN
K,C AND S,RESPECTIVELY,PROVIDED A TRANSFORMATION IS
PERFORMED.OTHERWISE THE MODULUS,COSINE AND SINE OF THE
COMPLEX NUMBER AR[L,J]+AI[L,J]*I ARE DELIVERED.

PROCEDURES USED:

CARPOL=CP34344,
TAMMAT=CP34014.

RUNNING TIME: PROPORTIONAL TO U-L.

METHOD AND PERFORMANCE:

SEE WILKINSON(1965,P.49,50).

LANGUAGE: ALGOL 60.

SUBSECTION: HSHCOMPRD.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" HSHCOMPRD(I, II, L, U, J, AR, AI, BR, BI, T);
"VALJE" I, II, L, U, J, T; "INTEGER" I, II, L, U, J; "REAL" T;
"ARRAY" AR, AI, BR, BI;

THE MEANING OF THE FORMAL PARAMETERS IS:

I,II,L,J: <ARITHMETIC EXPRESSION>;
ENTRY:
THE COMPLEX MATRIX TO BE PREMULTIPLIED, MUST BE GIVEN IN THE L-TH TO U-TH COLUMN FROM ROW I TO ROW II OF A COMPLEX MATRIX;

J: <ARITHMETIC EXPRESSION>;
ENTRY:
THE COMPLEX VECTOR V OF THE HOUSEHOLDER MATRIX I-VV"/T,WHERE " DENOTES TRANSPOSING AND CONJUGATING, MUST BE GIVEN IN THE J-TH COLUMN FROM ROW I TO ROW II OF A COMPLEX MATRIX GIVEN IN (BR,BI);

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[I:II,L:U];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX TO BE PREMULTIPLIED,MUST BE GIVEN IN THE ARRAYS AR AND AI,RESPECTIVELY;
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE RESULTING MATRIX ARE DELIVERED IN THE ARRAYS AR AND AI,RESPECTIVELY;

BR,BI: <ARRAY IDENTIFIER>;
"ARRAY" BR,BI[I:II,J:J];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX VECTOR V OF THE HOUSEHOLDER MATRIX MUST BE GIVEN IN THE ARRAYS BR AND BI,RESPECTIVELY;
(E.G. AS DELIVERED BY HS1COMCOL);

T: <ARITHMETIC EXPRESSION>;
ENTRY:
THE SCALAR T OF THE HOUSEHOLDER MATRIX;
(E.G. AS DELIVERED BY HSHCOMCOL);

PROCEDURES USED:

TAMMAT =CP34014,
ELMCOMCOL=CP34377,

RUNNING TIME: PROPORTIONAL TO (U-L)*(II-I).

LANGUAGE: ALGOL 60.

REFERENCE:

WILKINSON,J.H(1965):
THE ALGEBRAIC EIGENVALUE PROBLEM,
CLARENDON PRESS,OXFORD.

EXAMPLE OF USE:

AS A FORMAL TEST OF THE PROCEDURES HSHCOMCOL AND HSHCOMPRD THE
FOLLOWING MATRIX:
3 4*I
4*I 5
IS TRANSFORMED INTO UPPER TRIANGULAR FORM.

```
"BEGIN""INTEGER"I;"REAL"K,C,S,T;
  "ARRAY"AR,AI[1:2,1:2];
  "BOOLEAN""PROCEDURE" HSHCOMCOL(L,U,J,AR,AI,TOL,K,C,S,T);"CODE"34355;
  "PROCEDURE"HSHCOMPRD(I,II,L,U,J,AR,AI,BR,BI,T);"CODE"34356;
  AR[1,1]:=3;AR[1,2]:=AR[2,1]:=0;AR[2,2]:=5;
  AI[1,1]:=0;AI[1,2]:=AI[2,1]:=4;AI[2,2]:=0;
  "IF"HSHCOMCOL(1,2,1,AR,AI,("-14*5)**2,K,C,S,T)"THEN"
  HSHCOMPRD(1,2,2,2,1,AR,AI,AR,AI,T);
  OUTPUT(61,"("""("AFTER USE HSHCOMCOL,HSHCOMPRD:")",/,
   2(2(-D.D,+D.D,"("*I")",BB),/)")",
   AR[1,1],AI[1,1],AR[1,2],AI[1,2],AR[2,1],AI[2,1],AR[2,2],AI[2,2]);
  OUTPUT(61,"("""("K,   C,   S,   T,")",/,3(-D.DB),-DD.D,/,")",K,C,S,
   T);
"END"
```

OUTPUT:

```
AFTER USE HSHCOMCOL,HSHCOMPRD:
 8.0+0.0*I    0.0+1.6*I
 0.0+4.0*I    6.2+0.0*I
K,    C,    S,    T,
 5.0 -1.0   0.0   40.0
```

```
SOURCE TEXT(S) :

"CODE" 34354;
     "PROCEDURE" COMMATVEC(L, U, I, AR, AI, BR, BI, RR, RI);
     "VALUE" L, U, I; "INTEGER" L, U, I; "REAL" RR, RI;
     "ARRAY" AR, AI, BR, BI;
     "BEGIN" "REAL" "PROCEDURE" MATVEC(L,U,I,A,B);"CODE" 34011;
          "REAL" MV;
          MV:= MATVEC(L, U, I, AR, BR) - MATVEC(L, U, I, AI, BI);
          RI:= MATVEC(L, U, I, AI, BR) + MATVEC(L, U, I, AR, BI);
          RR:=MV
     "END" COMMATVEC;
          "EOP"

"CODE" 34355;
     "BOOLEAN" "PROCEDURE" HSHCOMCOL(L, U, J, AR, AI, TOL, K, C, S, T);
     "VALUE" L, U, J, TOL; "INTEGER" L, U, J; "REAL" TOL, K, C, S, T;
     "ARRAY" AR, AI;
     "BEGIN" "REAL" VR, DEL, MOD, H, ARLJ, AILJ;
          "PROCEDURE" CARPOL(AR,AI,R,C,S);"CODE" 34344;
          "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
          VR:= TAMMAT(L + 1, U, J, J, AR, AR) + TAMMAT(L + 1, U,
          J, J, AI, AI); ARLJ:= AR[L,J]; AILJ:= AI[L,J];
          CARPOL(ARLJ, AILJ, MOD, C, S); "IF" VR > TOL "THEN"
          "BEGIN" VR:= VR + ARLJ ** 2 + AILJ ** 2; H:= K:= SQRT(VR);
               T:= VR + MOD * H;
               "IF" ARLJ = 0 "AND" AILJ = 0 "THEN" AR[L,J]:= H "ELSE"
               "BEGIN" AR[L,J]:= ARLJ + C * K; AI[L,J]:= AILJ + S * K;
                    S:= - S
               "END";
               C:= - C; HSHCOMCOL:= "TRUE"
          "END"
          "ELSE"
          "BEGIN" HSHCOMCOL:= "FALSE"; K:= MOD; T:= - 1 "END"
     "END" HSHCOMCOL;
          "EOP"

"CODE" 34356;
     "PROCEDURE" HSHCOMPRD(I, II, L, U, J, AR, AI, BR, BI, T);
     "VALUE" I, II, L, U, J, T; "INTEGER" I, II, L, U, J; "REAL" T;
     "ARRAY" AR, AI, BR, BI;
     "BEGIN"
          "PROCEDURE" ELMCOMCOL(L,U,I,J,AR,AI,BR,BI,XR,XI);"CODE" 34377;
          "REAL" "PROCEDURE" TAMMAT(L,U,I,J,A,B);"CODE" 34014;
     "FOR" L:= L "STEP" 1 "UNTIL" U "DO" ELMCOMCOL(I, II, L, J, AR, AI,
     BR, BI, ( - TAMMAT(I, II, J, L, BR, AR) - TAMMAT(I, II, J,
     L, BI, AI)) / T, (TAMMAT(I, II, J, L, BI, AR) - TAMMAT(I,
     II, J, L, BR, AI)) / T);
     "END" HSHCOMPRD;
          "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET , C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED : 730813.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES ELMCOMVECCOL, ELMCOMCOL AND ELMCOMROWVEC.
ELMCOMVECCOL ADDS XR+I*XI TIMES THE COMPLEX COLUMN VECTOR GIVEN IN ARRAY BR,BI[L:U,J:J] TO THE COMPLEX VECTOR GIVEN IN ARRAY AR,AI[L:U].
ELMCOMCOL ADDS XR+I*XI TIMES THE COMPLEX COLUMN VECTOR GIVEN IN ARRAY BR,BI[L:U,J:J] TO THE COMPLEX COLUMN VECTOR GIVEN IN ARRAY AR,AI[L:U,I:I].
ELMCOMROWVEC ADDS XR+I*XI TIMES THE COMPLEX VECTOR GIVEN IN ARRAY BR,BI[L:U] TO THE COMPLEX ROW VECTOR GIVEN IN ARRAY AR,AI[I:I,L:U].

KEYWORDS :

COMPLEX VECTOR OPERATIONS ,
ELIMINATION.

SUBSECTION : ELMCOMVECCOL.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" ELMCOMVECCOL(L,U,J,AR,AI,BR,BI,XR,XI);
"VALUE" L,U,J,XR,XI;
"INTEGER" L,U,J;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;

THE MEANING OF THE FORMAL PARAMETERS IS :

L,U:	<ARITHMETIC EXPRESSION>; LOWER AND UPPER BOUND OF THE VECTORS;
J:	<ARITHMETIC EXPRESSION>; COLUMN-INDEX OF THE COLUMN VECTORS BR AND BI;
AR,AI:	<ARRAY IDENTIFIER>; "ARRAY" AR,AI[L:U] ENTRY: AR : REAL PART OF THE VECTOR, AI : IMAGINARY PART OF THE VECTOR. EXIT: THE RESULTING VECTOR (SEE ALSO BRIEF DESCRIPTION);

BR,BI: <ARRAY IDENTIFIER>;
"ARRAY" BR,BI[L:U,J:J];
ENTRY:
BR : REAL PART OF THE COLUMN VECTOR,
BI : IMAGINARY PART OF THE COLUMN VECTOR.
XR,XI: <ARITHMETIC EXPRESSION>;
ENTRY:
XR: REAL PART OF THE ELIMINATION FACTOR;
XI: IMAGINARY PART OF THE ELIMINATION FACTOR .

PROCEDURES JSED : ELMVECCOL = CP34021 .

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

SUBSECTION : ELMCOMCOL.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" ELMCOMCOL(L,U,I,J,AR,AI,BR,BI,XR,XI);
"VALUE" L,U,I,J,XR,XI;
"INTEGER" L,U,I,J;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;

THE MEANING OF THE FORMAL PARAMETERS IS :
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE VECTORS;
I,J: <ARITHMETIC EXPRESSION>;
I: COLUMN-INDEX OF THE COLUMN VECTORS AR AND AI;
J: COLUMN-INDEX OF THE COLUMN VECTORS BR AND BI;
AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[L:U,I:I]
ENTRY:
AR : REAL PART OF THE COLUMN VECTOR,
AI : IMAGINARY PART OF THE COLUMN VECTOR.
EXIT:
THE RESULTING VECTOR (SEE ALSO BRIEF DESCRIPTION);
BR,BI: <ARRAY IDENTIFIER>;
"ARRAY" BR,BI[L:U,J:J]
ENTRY:
BR : REAL PART OF THE COLUMN VECTOR,
BI : IMAGINARY PART OF THE COLUMN VECTOR GIVEN IN THE
XR,XI: <ARITHMETIC EXPRESSION>;
ENTRY:
XR: REAL PART OF THE ELIMINATION FACTOR;
XI: IMAGINARY PART OF THE ELIMINATION FACTOR .

PROCEDURES USED : ELMCOL = CP34023 .

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

SUBSECTION : ELMCOMROWVEC .

CALLING SEQUENCE :

```
    THE HEADING OF THE PROCEDURE READS :
    "PROCEDURE" ELMCOMROWVEC(L,U,I,AR,AI,BR,BI,XR,XI);
    "VALUE" L,U,I,XR,XI;
    "INTEGER" L,U,I;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;

    THE MEANING OF THE FORMAL PARAMETERS IS :
    L,U:        <ARITHMETIC EXPRESSION>;
                LOWER AND UPPER BOUND OF THE VECTORS;
    I:          <ARITHMETIC EXPRESSION>;
                ROW-INDEX OF THE ROW VECTORS AR AND AI;
    AR,AI:      <ARRAY IDENTIFIER>;
                "ARRAY" AR,AI[I:I,L:U]
                ENTRY:
                AR : REAL PART OF THE ROW VECTOR,
                AI : IMAGINARY PART OF THE ROW VECTOR,
                EXIT:
                THE RESULTING VECTOR (SEE ALSO BRIEF DESCRIPTION);
    BR,BI:      <ARRAY IDENTIFIER>;
                "ARRAY" BR,BI[L:U]
                ENTRY:
                BR : REAL PART OF THE VECTOR,
                BI : IMAGINARY PART OF THE VECTOR
    XR,XI:      <ARITHMETIC EXPRESSION>;
                ENTRY:
                XR: REAL PART OF THE ELIMINATION FACTOR;
                XI: IMAGINARY PART OF THE ELIMINATION FACTOR .
```

PROCEDURES USED : ELMROWVEC = CP34027 .

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

EXAMPLE OF USE :

```
      "BEGIN"
"COMMENT" EXAMPLE OF USE ELMCOMCOL;
"PROCEDURE" ELMCOMCOL(L,U,I,J,AR,AI,BR,BI,XR,XI);"CODE" 34377;
"REAL" "ARRAY" AR,AI[1:2,1:2];
"INTEGER" I,J;
"PROCEDURE" OUT(K);"INTEGER" K;
OUTPUT(61,"("2(=D,+D,"("*I  ")"),/")",
           AR[K,1],AI[K,1],AR[K,2],AI[K,2]);
AR[1,1]:=+1;AR[1,2]:=-9;AR[2,1]:=-1;AR[2,2]:=-1;
AI[1,1]:=+2;AI[1,2]:=+2;AI[2,1]:=+2;AI[2,2]:=-2;
OUTPUT(61,"(""("INPUT MATRIX:")",/")");
"FOR" I:=1,2 "DO" OUT(I);
ELMCOMCOL(1,2,2,1,AR,AI,AR,AI,1,-4);
OUTPUT(61,"("/,"("MATRIX AFTER ELIMINATION:")",/")");
OUTPUT(61,"(-D,+D,"("*I")",48,Z,D/")",
           AR[1,1],AI[1,1],AR[1,2],AI[1,2]);
OUT(2)
"END"
```

```
OUTPUT:
INPUT MATRIX:
 1+2*I   -9+2*I
-1+2*I   -1-2*I

MATRIX AFTER ELIMINATION:
 1+2*I      0
-1+2*I    6+4*I     .
```

```
SOURCE TEXT(S) :

"CODE" 34376;
     "PROCEDURE" ELMCOMVECCOL(L,U,J,AR,AI,BR,BI,XR,XI);
     "VALUE" L,U,J,XR,XI;
     "INTEGER" L,U,J;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;
     "BEGIN"
     "PROCEDURE" ELMVECCOL(L,U,I,A,B,X);"CODE" 34021;
          ELMVECCOL(L,U,J,AR,BR,XR);
          ELMVECCOL(L,U,J,AR,BI,-XI);
          ELMVECCOL(L,U,J,AI,BR,XI);
          ELMVECCOL(L,U,J,AI,BI,XR)
     "END" ELMCOMVECCOL;
          "EOP"

"CODE" 34377;
     "PROCEDURE" ELMCOMCOL(L,U,I,J,AR,AI,BR,BI,XR,XI);
     "VALUE" L,U,I,J,XR,XI;
     "INTEGER" L,U,I,J;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;
     "BEGIN"
     "PROCEDURE" ELMCOL(L,U,I,J,A,B,X);"CODE" 34023;
          ELMCOL(L,U,I,J,AR,BR,XR);
          ELMCOL(L,U,I,J,AR,BI,-XI);
          ELMCOL(L,U,I,J,AI,BR,XI);
          ELMCOL(L,U,I,J,AI,BI,XR)
     "END" ELMCOMCOL;
          "EOP"

"CODE" 34378;
     "PROCEDURE" ELMCOMROWVEC(L,U,I,AR,AI,BR,BI,XR,XI);
     "VALUE" L,U,I,XR,XI;
     "INTEGER" L,U,I;"REAL" XR,XI;"ARRAY" AR,AI,BR,BI;
     "BEGIN"
     "PROCEDURE" ELMROWVEC(L,U,I,A,B,X);"CODE" 34027;
          ELMROWVEC(L,U,I,AR,BR,XR);
          ELMROWVEC(L,U,I,AR,BI,-XI);
          ELMROWVEC(L,U,I,AI,BR,XI);
          ELMROWVEC(L,U,I,AI,BI,XR)
     "END" ELMCOMROWVEC;
          "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED : 730817.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES ROTCOMCOL AND ROTCOMROW.
ROTCOMCOL REPLACES THE COLUMN VECTOR VR+I*VI GIVEN IN THE ARRAYS AR,AI[L:U,I:I] AND THE COLUMN VECTOR YR+I*YI GIVEN IN THE ARRAYS AR,AI[L:U,J:J] BY THE VECTORS (VR+I*VI)*(CR-I*CI)-(YR+I*YI)*S AND (YR+I*YI)*(CR+I*CI)+(VR+I*VI)*S, RESPECTIVELY.
ROTCOMROW REPLACES THE ROW VECTOR VR+I*VI GIVEN IN THE ARRAYS AR,AI[I:I,L:U] AND THE ROW VECTOR YR+I*YI GIVEN IN THE ARRAYS AR,AI[J:J,L:U] BY THE VECTORS (VR+I*VI)*(CR-I*CI)-(YR+I*YI)*S AND (YR+I*YI)*(CR+I*CI)-(VR+I*VI)*S, RESPECTIVELY.

KEYWORDS :

COMPLEX VECTOR OPERATIONS,
ROTATION .

SUBSECTION : ROTCOMCOL .

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE" ROTCOMCOL(L, U, I, J, AR, AI, CR, CI, S);
"VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
"REAL" CR, CI, S; "ARRAY" AR, AI;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

L,U,I,J: <ARITHMETIC EXPRESSION>;
THE ROTATION IS PERFORMED ON THE COLUMN VECTORS AR,AI[L:U,I:I] AND AR,AI[L:U,J:J];

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[L:U,I:J]
ENTRY:
AR:THE REAL PARTS OF THE COLUMN VECTORS
AI:THE IMAGINARY PARTS OF THE COLUMN VECTORS
EXIT:
THE RESULTING VECTORS ,OF WHICH THE REAL AND IMAGINARY PARTS ARE GIVEN IN AR AND AI RESPECTIVELY;
(SEE ALSO BRIEF DESCRIPTION);

CR,CI,S: <ARITHMETIC EXPRESSION>;
ENTRY:
ROTATION FACTORS; SEE ALSO BRIEF DESCRIPTION.

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

SUBSECTION : ROTCOMROW .

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" ROTCOMROW(L, U, I, J, AR, AI, CR, CI, S);
"VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
"REAL" CR, CI, S; "ARRAY" AR, AI;

THE MEANING OF THE FORMAL PARAMETERS IS :

L,U,I,J: <ARITHMETIC EXPRESSION>;
THE ROTATION IS PERFORMED ON THE ROW VECTORS
AR,AI[I:I,L:U] AND AR,AI[J:J,L:U];

AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[I:J,L:U]
ENTRY:
AR:THE REAL PARTS OF THE ROW VECTORS
AI:THE IMAGINARY PARTS OF THE ROW VECTORS
EXIT:
THE RESULTING VECTORS, OF WHICH THE REAL AND IMAGINARY
PARTS ARE GIVEN IN AR AND AI RESPECTIVELY;
(SEE ALSO BRIEF DESCRIPTION);

CR,CI,S: <ARITHMETIC EXPRESSION>;
ENTRY:
ROTATION FACTORS; SEE ALSO BRIEF DESCRIPTION.

PROCEDURES USED : NONE .

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

EXAMPLE OF USE :

```
    "BEGIN"
"COMMENT" EXAMPLE OF USE ROTCOMCOL;
"PROCEDURE" ROTCOMCOL(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34357;
"REAL" "ARRAY" AR,AI[1:2,1:2];
"INTEGER" I,J;
AR[1,1]:=+4;AR[1,2]:=+5;AR[2,1]:=-5;AR[2,2]:=+4;
AI[1,1]:=+3;AI[1,2]:= 0;AI[2,1]:= 0;AI[2,2]:=-3;
OUTPUT(61,"("""("INPUT MATRIX:")",/")");
OUTPUT(61,"("-D,+D,"("*I")",4B,-D,Z/,B8-D,Z,3B,-D,+D,"("*I")",/")",
AR[1,1],AI[1,1],AR[1,2],AI[1,2],AR[2,1],AI[2,1],AR[2,2],AI[2,2]);
OUTPUT(61,"("//,"("AFTER POSTMULTIPLICATION WITH:")",/")");
OUTPUT(61,"("""(",08-.06*I      -.1")",/,
          "("     .1      .08+.06*I")",//")");
ROTCOMCOL(1,2,1,2,AR,AI,.08,.06,-.1);
OUTPUT(61,"("""("DELIVERS:")",/")");
OUTPUT(61,"("-D,Z,2BD,Z/,BD,Z,B-D,Z")",
AR[1,1],AI[1,1],AR[1,2],AI[1,2],AR[2,1],AI[2,1],AR[2,2],AI[2,2]);
"END"
```

```
OUTPUT:
INPUT MATRIX:
 4+3*I      5
  -5      4-3*I

AFTER POSTMULTIPLICATION WITH:
.08-.06*I      -.1
    .1       .08+.06*I
DELIVERS:
 1    0
 0    1
```

```
SOURCE TEXT(S) ;
"CODE" 34357;
    "PROCEDURE" ROTCOMCOL(L, U, I, J, AR, AI, CR, CI, S);
    "VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
    "REAL" CR, CI, S; "ARRAY" AR, AI;
    "BEGIN" "REAL" ARLI, AILI, ARLJ, AILJ;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" ARLI:= AR[L,I]; AILI:= AI[L,I]; ARLJ:= AR[L,J];
            AILJ:= AI[L,J];
            AR[L,I]:= CR * ARLI + CI * AILI - S * ARLJ;
            AI[L,I]:= CR * AILI - CI * ARLI - S * AILJ;
            AR[L,J]:= CR * ARLJ - CI * AILJ + S * ARLI;
            AI[L,J]:= CR * AILJ + CI * ARLJ + S * AILI;
        "END"
    "END" ROTCOMCOL;
        "EOP"

"CODE" 34358;
    "PROCEDURE" ROTCOMROW(L, U, I, J, AR, AI, CR, CI, S);
    "VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
    "REAL" CR, CI, S; "ARRAY" AR, AI;
    "BEGIN" "REAL" ARIL, AIIL, ARJL, AIJL;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" ARIL:= AR[I,L]; AIIL:= AI[I,L]; ARJL:= AR[J,L];
            AIJL:= AI[J,L];
            AR[I,L]:= CR * ARIL + CI * AIIL + S * ARJL;
            AI[I,L]:= CR * AIIL - CI * ARIL + S * AIJL;
            AR[J,L]:= CR * ARJL - CI * AIJL - S * ARIL;
            AI[J,L]:= CR * AIJL + CI * ARJL - S * AIIL;
    "END"
"END" ROTCOMROW;
    "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731016.

BRIEF DESCRIPTION:

COMEUCNRM CALCULATES THE EUCLIDEAN NORM OF A COMPLEX MATRIX WITH LW LOWER CODIAGONALS.

KEYWORDS:

EUCLIDEAN NORM,
COMPLEX MATRIX.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" COMEUCNRM(AR, AI, LW, N); "VALUE" N, LW;
"INTEGER" N, LW; "ARRAY" AR, AI;

COMEUCNRM DELIVERS THE EUCLIDEAN NORM OF A COMPLEX MATRIX WITH LW LOWER CODIAGONALS;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX;
LW: <ARITHMETIC EXPRESSION>;
THE NUMBER OF LOWER CODIAGONALS;
AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX MATRIX,WITH LW LOWER CODIAGONALS,MUST BE GIVEN IN THE ARRAYS AR AND AI,RESPECTIVELY.

PROCEDURES USED: MATTAM = CP34015.

RUNNING TIME: PROPORTIONAL TO N**2.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:SEE EIGVALCOM OR EIGCOM (SECTION 3.3.2.2.2).

SOURCE TEXT(S) :

```
"CODE" 34359;
     "REAL" "PROCEDURE" COMEUCNRM(AR, AI, LW, N); "VALUE" N, LW;
     "INTEGER" N, LW; "ARRAY" AR, AI;
     "BEGIN" "INTEGER" I, L;
     "REAL" "PROCEDURE" MATTAM(L,U,I,J,A,B);"CODE" 34015;
     "REAL" R;
     R:= 0;
     "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
     "BEGIN" L:= "IF" I > LW "THEN" I - LW "ELSE" 1;
          R:= MATTAM(L, N, I, I, AR, AR) + MATTAM(L, N, I,
          I, AI, AI) + R;
     "END";
     COMEUCNRM:= SQRT(R)
"END" COMEUCNRM;
     "EOP"
```

AUTHORS : J.J.G. ADMIRAAL, C.G. VAN DER LAAN.

CONTRIBUTORS : J.J.G. ADMIRAAL, H. FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE, UNIVERSITY OF AMSTERDAM.

RECEIVED : 730817.

BRIEF DESCRIPTION :

THIS SECTION CONTAINS THE PROCEDURES ROTCOMCOL, ROTCOMROW AND CHSH2
ROTCOMCOL REPLACES THE COLUMN VECTOR VR+I*VI GIVEN IN THE ARRAYS
AR,AI[L:U,I:I] AND THE COLUMN VECTOR YR+I*YI GIVEN IN THE ARRAYS
AR,AI[L:J,J:J] BY THE VECTORS (VR+I*VI)*(CR-I*CI)-(YR+I*YI)*S AND
(YR+I*YI)*(CR+I*CI)+(VR+I*VI)*S, RESPECTIVELY.
ROTCOMROW REPLACES THE ROW VECTOR VR+I*VI GIVEN IN THE ARRAYS
AR,AI[I:I,L:U] AND THE ROW VECTOR YR+I*YI GIVEN IN THE ARRAYS
(YR+I*YI)*(CR+I*CI)-(VR+I*VI)*S, RESPECTIVELY.
CHSH2 COMPUTES THE COMPLEX HOUSEHOLDERMATRIX THAT
MAPS THE COMPLEX VECTOR (A1,A2) INTO THE DIRECTION (1,0).
WARNING : IN ROTCOMCOL AND ROTCOMROW THE COSINE IS COMPLEX AND THE
SINE IS REAL, IN CONTRAST TO THIS, IN CHSH2 THE SINE IS COMPLEX
AND THE COSINE IS REAL.

KEYWORDS :

COMPLEX VECTOR OPERATIONS,
ROTATION,
HOUSEHOLDER MATRIX.

SUBSECTION : ROTCOMCOL .

CALLING SEQUENCE :

```
THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" ROTCOMCOL(L, U, I, J, AR, AI, CR, CI, S);
"VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
"REAL" CR, CI, S; "ARRAY" AR, AI;
"CODE" 34357;

THE MEANING OF THE FORMAL PARAMETERS IS :
L,U,I,J:    <ARITHMETIC EXPRESSION>;
            THE ROTATION IS PERFORMED ON THE COLUMN VECTORS
            AR,AI[L:U,I:I] AND AR,AI[L:U,J:J];
AR,AI:      <ARRAY IDENTIFIER>;
            "ARRAY" AR,AI[L:U,I:J];
            ENTRY:
            AR:THE REAL PARTS OF THE COLUMN VECTORS
            AI:THE IMAGINARY PARTS OF THE COLUMN VECTORS
            EXIT:
            THE RESULTING VECTORS (SEE ALSO BRIEF DESCRIPTION);
CR,CI,S:    <ARITHMETIC EXPRESSION>;
            ENTRY:
            ROTATION FACTORS; SEE ALSO BRIEF DESCRIPTION.
```

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

SUBSECTION : ROTCOMROW .

CALLING SEQUENCE :

```
THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" ROTCOMROW(L, U, I, J, AR, AI, CR, CI, S);
"VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
"REAL" CR, CI, S; "ARRAY" AR, AI;
"CODE" 34358;

THE MEANING OF THE FORMAL PARAMETERS IS :
L,U,I,J:    <ARITHMETIC EXPRESSION>;
            THE ROTATION IS PERFORMED ON THE ROW VECTORS
            AR,AI[I:I,L:U] AND AR,AI[J:J,L:U];
AR,AI:      <ARRAY IDENTIFIER>;
            "ARRAY" AR,AI[I:J,L:U];
            ENTRY:
            AR:THE REAL PARTS OF THE ROW VECTORS
            AI:THE IMAGINARY PARTS OF THE ROW VECTORS
            EXIT:
            THE RESULTING VECTORS (SEE ALSO BRIEF DESCRIPTION);
CR,CI,S:    <ARITHMETIC EXPRESSION>;
            ENTRY:
            ROTATION FACTORS; SEE ALSO BRIEF DESCRIPTION.
```

PROCEDURES USED : NONE .

RUNNING TIME : ROUGHLY PROPORTIONAL TO (U-L) .

LANGUAGE: ALGOL 60.

EXAMPLE OF USE :

```
"BEGIN"
"COMMENT" EXAMPLE OF USE ROTCOMCOL;
"PROCEDURE" ROTCOMCOL(L,U,I,J,AR,AI,CR,CI,S);"CODE" 34357;
"REAL" "ARRAY" AP,AI[1:2,1:2];
"INTEGER" I,J;
AR[1,1]:=+4;AR[1,2]:=+5;AR[2,1]:=-5;AR[2,2]:=+4;
AI[1,1]:=+3;AI[1,2]:= 0;AI[2,1]:= 0;AI[2,2]:=-3;
OUTPUT(61,"("("("INPUT MATRIX:")",/")");
OUTPUT(61,"("-D,+D,"("*I")",4B,-D,Z/,BB-D,Z,3B,-D,+D,"("*I")",/")",
AR[1,1],AI[1,1],AR[1,2],AI[1,2],AR[2,1],AI[2,1],AR[2,2],AI[2,2]);
OUTPUT(61,"("//,"("AFTER POSTMULTIPLICATION WITH:")",/")");
OUTPUT(61,"("("("
```

```
.08-.06*I      -.1")",/,
          "("      .1      .08+.06*I")",//")");
ROTCOMCOL(1,2,1,2,AR,AI,.08,.06,-.1);
OUTPUT(61,"("("("DELIVERS:")",/")");
OUTPUT(61,"("-D,Z,2BD,Z/,BD,Z,B-D,Z")",
AR[1,1],AI[1,1],AR[1,2],AI[1,2],AR[2,1],AI[2,1],AR[2,2],AI[2,2]);
"END"
```

```
OUTPUT:
INPUT MATRIX:
 4+3*I      5
  -5      4-3*I

AFTER POSTMULTIPLICATION WITH:
.08-.06*I      -.1
    .1       .08+.06*I
DELIVERS:
 1    0
 0    1
```

SUBSECTION: CHSH2.

CALLING SEQUENCE:
THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" CHSH2(A1R,A1I,A2R,A2I,C,SR,SI);
"VALUE" A1R,A1I,A2R,A2I;"REAL" A1R,A1I,A2R,A2I,C,SR,SI;
"CODE" 34611;

THE MEANING OF THE FORMAL PARAMETERS IS:
A1R: <ARITHMETIC EXPRESSION>;
ENTRY: THE REAL PART OF THE FIRST VECTORCOMPONENT;
A1I: <ARITHMETIC EXPRESSION>;
ENTRY: THE IMAGINARY PART OF THE FIRST VECTORCOMPONENT;
A2R: <ARITHMETIC EXPRESSION>;
ENTRY: THE REAL PART OF THE SECOND VECTORCOMPONENT;
A2I: <ARITHMETIC EXPRESSION>;
ENTRY: THE IMAGINARY PART OF THE SECOND VECTORCOMPONENT;
C,SR,SI: <VARIABLE>;
EXIT: THE FACTORS THAT DETERMINE THE HOUSEHOLDER MATRIX.
THE HOUSEHOLDERMATRIX, DEFINED BY:
HA = B
A = (A1,A2)'
B = (-SIGN(A1R)*SQRT(A1*A1+A2*A2),0)',
IS DETERMINED BY:

(-C SR+I*SI)
(SR+I*SI C)

PROCEDURES USED: NONE;

LANGUAGE: ALGOL 60;

METHOD AND PERFORMANCE:

AFTER A CALL OF CHSH2 YOU ARE ABLE TO ROTATE A COMPLEX VECTOR OF DIMENSION TWO BY MEANS OF THE FACTORS C,SR AND SI.

EXAMPLE OF USE: CHSH2 IS USED IN QZI AND QZIVAL,SECTION 3.3.4

SOURCE TEXT(S) :

```
"CODE" 34357;
    "PROCEDURE" ROTCOMCOL(L, U, I, J, AR, AI, CR, CI, S);
    "VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
    "REAL" CR, CI, S; "ARRAY" AR, AI;
    "BEGIN" "REAL" ARLI, AILI, ARLJ, AILJ;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" ARLI:= AR[L,I]; AILI:= AI[L,I]; ARLJ:= AR[L,J];
            AILJ:= AI[L,J];
            AR[L,I]:= CR * ARLI + CI * AILI - S * ARLJ;
            AI[L,I]:= CR * AILI - CI * ARLI - S * AILJ;
            AR[L,J]:= CR * ARLJ - CI * AILJ + S * ARLI;
            AI[L,J]:= CR * AILJ + CI * ARLJ + S * AILI;
        "END"
    "END" ROTCOMCOL;
        "EOP"

"CODE" 34358;
    "PROCEDURE" ROTCOMROW(L, U, I, J, AR, AI, CR, CI, S);
    "VALUE" L, U, I, J, CR, CI, S; "INTEGER" L, U, I, J;
    "REAL" CR, CI, S; "ARRAY" AR, AI;
    "BEGIN" "REAL" ARIL, AIIL, ARJL, AIJL;
        "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
        "BEGIN" ARIL:= AR[I,L]; AIIL:= AI[I,L]; ARJL:= AR[J,L];
            AIJL:= AI[J,L];
            AR[I,L]:= CR * ARIL + CI * AIIL + S * ARJL;
            AI[I,L]:= CR * AIIL - CI * ARIL + S * AIJL;
            AR[J,L]:= CR * ARJL - CI * AIJL - S * ARIL;
            AI[J,L]:= CR * AIJL + CI * ARJL - S * AIIL;
        "END"
    "END" ROTCOMROW;
        "EOP"

"CODE" 34611;
    "PROCEDURE"CHSH2(A1R,A1I,A2R,A2I,C,SR,SI);
    "VALUE"A1R,A1I,A2R,A2I;"REAL"A1R,A1I,A2R,A2I,C,SR,SI;
"BEGIN" "REAL" R;
"IF" A2R=0 "OR" A2I=0 "THEN"
"BEGIN" "IF" A1R=0 "OR" A1I=0 "THEN"
    "BEGIN" R:=SQRT(A1R*A1R+A1I*A1I);C:=R;
        SR:=(A1R*A2R+A1I*A2I)/R;SI:=(A1R*A2I-A1I*A2R)/R;
        R:=SQRT(C*C+SR*SR+SI*SI);C:=C/R;SR:=SR/R;SI:=SI/R
    "END" "ELSE"
    "BEGIN" SI:=C:=0;SR:=1 "END"
"END" "ELSE" "BEGIN" C:=1;SR:=SI:=0 "END"
"END" CHSH2;
      "EOP"
```

AUTHORS :

T.J. DEKKER, W. HOFFMANN (COMSCL),
C.G. VAN DER LAAN (SCLCOM).

CONTRIBUTORS:

W. HOFFMANN, S.P.N. VAN KAMPEN (COMSCL),
H. FIOLET, C.G. VAN DER LAAN (SCLCOM).

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731030.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES :

COMSCL NORMALIZES THE REAL AND COMPLEX EIGENVECTORS GIVEN COLUMNWISE IN A TWO-DIMENSIONAL ARRAY; THE IMAGINARY PARTS OF THE CORRESPONDING EIGENVALUES MUST BE GIVEN IN A ONE-DIMENSIONAL ARRAY;
THE EIGENVECTORS ARE NORMALIZED IN SUCH A WAY THAT, IN EACH EIGENVECTOR, AN ELEMENT OF MAXIMUM MODULUS EQUALS 1;
THE NORMALIZED EIGENVECTORS ARE DELIVERED IN THE GIVEN ARRAY.

SCLCOM NORMALIZES THE (NON-NULL) COLUMNS OF A COMPLEX MATRIX IN SUCH A WAY THAT IN EACH COLUMN AN ELEMENT OF MAXIMUM ABSOLUTE VALUE BECOMES EQUAL TO ONE.

KEYWORDS:

NORMALIZATION,
SCALING OF COMPLEX EIGENVECTORS,
COMPLEX SCALING.

SUBSECTION : COMSCL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE IS:
"PROCEDURE" COMSCL(A, N, N1, N2, IM); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A, IM;
"CODE" 34193;

THE MEANING OF THE FORMAL PARAMETERS IS:

A: <ARRAY IDENTIFIER>;
"ARRAY" A[1:N,N1:N2];
ENTRY: EACH REAL EIGENVECTOR MUST BE GIVEN IN A COLUMN OF ARRAY A, WHOSE CORRESPONDING ELEMENT OF ARRAY IM EQUALS 0;
THE REAL AND IMAGINARY PART OF EACH COMPLEX EIGENVECTOR MUST BE GIVEN IN CONSECUTIVE COLUMNS OF ARRAY A, WHOSE CORRESPONDING ELEMENTS OF ARRAY IM ARE NOT EQUAL TO 0;
EXIT: THE NORMALIZED EIGENVECTORS (I.E. IN EACH EIGENVECTOR AN ELEMENT OF MAXIMUM MODULUS EQUALS 1) ARE DELIVERED IN THE CORRESPONDING COLUMNS OF A;

N: <ARITHMETIC EXPRESSION>;
THE NUMBER OF ROWS OF ARRAY A;

N1, N2: <ARITHMETIC EXPRESSION>;
THE LOWER AND UPPER BOUND OF THE COLUMN INDICES OF ARRAY A;

IM: <ARRAY IDENTIFIER>;
"ARRAY" IM[N1:N2];
THE IMAGINARY PARTS OF THE EIGENVALUES, OF WHICH THE EIGENVECTORS ARE GIVEN IN THE CORRESPONDING COLUMNS OF ARRAY A, MUST BE GIVEN IN ARRAY IM.

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N * (N2 - N1 + 1).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE REF [1].

REFERENCES:

[1].T.J. DEKKER AND W. HOFFMANN.
ALGOL 60 PROCEDURES IN NUMERICAL ALGEBRA, PART 2.
MC TRACT 23, 1968, MATH. CENTR., AMSTERDAM.

EXAMPLE OF USE:
THE PROCEDURE COMSCL IS USED IN COMEIG1, SECTION 3.3.1.2.2.

SUBSECTION : SCLCOM.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE"SCLCOM(AR,AI,N,N1,N2);
"VALUE"N,N1,N2;"INTEGER"N,N1,N2;"ARRAY"AR,AI;
"CODE" 34360;

THE MEANING OF THE FORMAL PARAMETERS IS:
AR,AI:   <ARRAY IDENTIFIER>;
         "ARRAY" AR,AI[1:N,N1:N2];
         ENTRY:
         THE REAL PART AND THE IMAGINARY PART OF THE MATRIX OF
         WHICH THE COLUMNS ARE TO BE SCALED MUST BE GIVEN IN THE
         ARRAYS AR AND AI,RESPECTIVELY;
         EXIT:
         THE REAL PART AND THE IMAGINARY PART OF THE MATRIX WITH
         SCALED COLUMNS  ARE DELIVERED IN THE ARRAYS AR AND AI,
         RESPECTIVELY;
N,N1,N2:<ARITHMETIC EXPRESSION>;
         N    : ORDER OF THE MATRIX;
         N1,N2: THE N1-TH TO     N2-TH COLUMN VECTORS ARE TO BE
         SCALED.
```

PROCEDURES USED: COMCOLCST = CP34352.

RUNNING TIME: PROPORTIONAL TO N*(N2-N1).

LANGUAGE: ALGOL 60.

EXAMPLE OF USE: SEE EIGCOM (SECTION 3.3.2.2.2).

SOURCE TEXT(S) :

```
"CODE" 34193;
    "COMMENT" MCA 2423;
    "PROCEDURE" COMSCL(A, N, N1, N2, IM); "VALUE" N, N1, N2;
"INTEGER" N, N1, N2; "ARRAY" A, IM;
"BEGIN" "INTEGER" I, J, K;
    "REAL" S, U, V, W;

    "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO"
    "BEGIN" S:= 0; "IF" IM[J] ^= 0 "THEN"
        "BEGIN" "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
            "BEGIN" U:= A[I,J] ** 2 + A[I,J + 1] ** 2;
                "IF" U > S "THEN" "BEGIN" S:= U; K:= I "END"
            "END";
            "IF" S ^= 0 "THEN"
            "BEGIN" V:= A[K,J] / S; W:= - A[K,J + 1] / S;
                "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
                "BEGIN" U:= A[I,J]; S:= A[I,J + 1];
                    A[I,J]:= U * V - S * W;
                    A[I,J + 1]:= U * W + S * V
                "END"
                "END";
                J:= J + 1
            "END"
            "ELSE"
            "BEGIN" "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
                "IF" ABS(A[I,J]) > ABS(S) "THEN" S:= A[I,J];
                "IF" S ^= 0 "THEN"
                "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO" A[I,J]:= A[I,J] / S
            "END"
        "END"
    "END" COMSCL;
        "EOP"

"CODE" 34360;
    "PROCEDURE" SCLCOM(AR, AI, N, N1, N2); "VALUE" N, N1, N2;
    "INTEGER" N, N1, N2; "ARRAY" AR, AI;
    "BEGIN" "INTEGER" I, J, K;
        "REAL" S, R;
    "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);"CODE" 34352;
    "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO"
    "BEGIN" S:= 0;
        "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
        "BEGIN" R:= AR[I,J] ** 2 + AI[I,J] ** 2; "IF" R > S "THEN"
            "BEGIN" S:= R; K:= I "END"
        "END";
        "IF" S ^= 0 "THEN" COMCOLCST(1, N, J, AR, AI, AR[K,J] /
        S, - AI[K,J] / S)
    "END"
"END" SCLCOM;
    "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731016.

BRIEF DESCRIPTION:

COMEUCNRM CALCULATES THE EUCLIDEAN NORM OF A COMPLEX MATRIX WITH LW LOWER CODIAGONALS.

KEYWORDS:

EUCLIDEAN NORM,
COMPLEX MATRIX.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL" "PROCEDURE" COMEUCNRM(AR, AI, LW, N); "VALUE" N, LW;
"INTEGER" N, LW; "ARRAY" AR, AI;

COMEUCNRM DELIVERS THE EUCLIDEAN NORM OF A COMPLEX MATRIX WITH LW LOWER CODIAGONALS;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
THE ORDER OF THE MATRIX;
LW: <ARITHMETIC EXPRESSION>;
THE NUMBER OF LOWER CODIAGONALS;
AR,AI: <ARRAY IDENTIFIER>;
"ARRAY" AR,AI[1:N,1:N];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX MATRIX,WITH LW LOWER CODIAGONALS,MUST BE GIVEN IN THE ARRAYS AR AND AI,RESPECTIVELY.

PROCEDURES USED: MATTAM = CP34015.

RUNNING TIME: PROPORTIONAL TO N**2.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:SEE EIGVALCOM OR EIGCOM (SECTION 3.3.2.2.2).

SOURCE TEXT(S) :

```
"CODE" 34359;
     "REAL" "PROCEDURE" COMEUCNRM(AR, AI, LW, N); "VALUE" N, LW;
     "INTEGER" N, LW; "ARRAY" AR, AI;
     "BEGIN" "INTEGER" I, L;
          "REAL" "PROCEDURE" MATTAM(L,U,I,J,A,B);"CODE" 34015;
          "REAL" R;
          R:= 0;
     "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
     "BEGIN" L:= "IF" I > LW "THEN" I - LW "ELSE" 1;
          R:= MATTAM(L, N, I, I, AR, AR) + MATTAM(L, N, I,
          I, AI, AI) + R;
     "END";
     COMEUCNRM:= SQRT(R)
"END" COMEUCNRM;
     "EOP"
```

AUTHOR : C.G. VAN DER LAAN,

CONTRIBUTORS : H.FIOLET, C.G. VAN DER LAAN,

INSTITUTE: MATHEMATICAL CENTRE,

RECEIVED: 731016,

BRIEF DESCRIPTION:

SCLCOM NORMALIZES THE (NON-NULL) COLUMNS OF A COMPLEX MATRIX IN SUCH A WAY THAT IN EACH COLUMN AN ELEMENT OF MAXIMUM ABSOLUTE VALUE BECOMES EQUAL TO ONE.

KEYWORDS:

COMPLEX SCALING,

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE"SCLCOM(AR,AI,N,N1,N2);
"VALUE"N,N1,N2;"INTEGER"N,N1,N2;"ARRAY"AR,AI;

THE MEANING OF THE FORMAL PARAMETERS IS:

AR,AI: <ARRAY IDENTIFIER>,
"ARRAY" AR,AI[1:N,N1:N2];
ENTRY:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX OF WHICH THE COLUMNS ARE TO BE SCALED MUST BE GIVEN IN THE ARRAYS AR AND AI,RESPECTIVELY;
EXIT:
THE REAL PART AND THE IMAGINARY PART OF THE MATRIX WITH SCALED COLUMNS ARE DELIVERED IN THE ARRAYS AR AND AI, RESPECTIVELY;

N,N1,N2: <ARITHMETIC EXPRESSION>;
N : ORDER OF THE MATRIX;
N1,N2: THE N1-TH TO N2-TH COLUMN VECTORS ARE TO BE SCALED.

PROCEDURES USED: COMCOLCST = CP34352.

RUNNING TIME: PROPORTIONAL TO N*(N2-N1).

LANGUAGE: ALGOL 60.

EXAMPLE OF USE: SEE EIGCOM (SECTION 3.3.2.2.2).

SOURCE TEXT(S) :

```
"CODE" 34360;
     "PROCEDURE" SCLCOM(AR, AI, N, N1, N2); "VALUE" N, N1, N2;
     "INTEGER" N, N1, N2; "ARRAY" AR, AI;
     "BEGIN" "INTEGER" I, J, K;
          "REAL" S, R;
          "PROCEDURE" COMCOLCST(L,U,J,AR,AI,XR,XI);"CODE" 34352;
          "FOR" J:= N1 "STEP" 1 "UNTIL" N2 "DO"
          "BEGIN" S:= 0;
          "FOR" I:= 1 "STEP" 1 "UNTIL" N "DO"
          "BEGIN" R:= AR[I,J] ** 2 + AI[I,J] ** 2; "IF" R > S "THEN"
               "BEGIN" S:= R; K:= I "END"
          "END";
          "IF" S =/= 0 "THEN" COMCOLCST(1, N, J, AR, AI, AR[K,J] /
          S, - AI[K,J] / S)
     "END"
"END" SCLCOM;
     "EOP"
```

AUTHOR: C.G. VAN DER LAAN,

INSTITUTE: MATHEMATICAL CENTRE,

RECEIVED: 730815,

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THREE PROECEDURES:
COMABS CALCULATES THE MODULUS OF A COMPLEX NUMBER,
COMSQRT CALCULATES THE SQUARE ROOT OF A COMPLEX NUMBER
CARPOL TRANSFORMS A COMPLEX NUMBER GIVEN IN CARTESIAN COORDINATES
INTO POLAR COORDINATES

KEYWORDS:
COMPLEX NUMBER,
MODULUS,
SQUARE ROOT,
TRANSFORMATION,
CARTESIAN COORDINATES,
POLAR COORDINATES,

SUBSECTION: COMABS,

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"REAL""PROCEDURE"COMABS(XR,XI);
"VALUE"XR,XI;"REAL"XR,XI;

COMABS DELIVERS THE MODULUS OF THE COMPLEX NUMBER XR + I * XI;
THE MEANING OF THE FORMAL PARAMETERS IS:
XR,XI:<ARITHMETIC EXPRESSION>;
ENTRY:XR,XI ARE THE REAL PART AND THE IMAGINARY PART
OF THE COMPLEX NUMBER,RESPECTIVELY,

PROCEDURES USED: NONE,

LANGUAGE: ALGOL 60,

```
EXAMPLE OF USE:

"BEGIN"
"REAL""PROCEDURE"COMABS(XR,XI);
"CODE"34340;
OUTPUT(61,"(""("THE MODULUS OF ,3+,4*I EQUALS")",-D,DD")",
      COMABS(,3,,4))
"END"

THE MODULUS OF ,3+,4*I EQUALS 0,50
```

SUBSECTION : COMSQRT,

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE"COMSQRT(AR,AI,PR,PI);
"VALUE"AR,AI;"REAL"AR,AI,PR,PI;

THE MEANING OF THE FORMAL PARAMETERS IS:
AR,AI:<ARITHMETIC EXPRESSION>;
ENTRY:AR,AI ARE THE REAL PART AND THE IMAGINARY PART
OF THE COMPLEX NUMBER,RESPECTIVELY;
PR,PI;<VARIABLE>;
EXIT:THE REAL PART AND THE IMAGINARY PART OF THE SQUARE ROOT
ARE DELIVERED IN PR AND PI,RESPECTIVELY.

PROCEDURES USED: NONE.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE REPRESENTATION OF THE RESULTING COMPLEX NUMBER IS CHOSEN SUCH THAT ITS REAL PART IS NONNEGATIVE;THE PROCEDURE IS PROTECTED AGAINST INTERMEDIATE OVERFLOW.

EXAMPLE OF USE:

```
"BEGIN""REAL"R,I;
"PROCEDURE"COMSQRT(AR,AI,PR,PI);
"CODE"34343;
COMSQRT(-3,4,R,I);
OUTPUT(61,"(""("THE SQUARE ROOT OF -3+4*I IS")",-D,DD,+D,DD,"("*I")"
      ")",R,I);
"END"

THE SQUARE ROOT OF -3+4*I IS 1,00+2,00*I
```

SUBSECTION : CARPOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE"CARPOL(AR,AI,R,C,S);
"VALUE"AR,AI;"REAL"AR,AI,R,C,S;

THE MEANING OF THE FORMAL PARAMETERS IS:
AR,AI:<ARITHMETIC EXPRESSION>;
ENTRY:AR,AI ARE THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX NUMBER ,RESPECTIVELY;
R,C,S:<VARIABLE>;
EXIT:THE MODULUS OF THE COMPLEX NUMBER IS DELIVERED IN R AND THE COSINE AND THE SINE OF THE ARGUMENT ARE DELIVERED IN C AND S,RESPECTIVELY;
WHEN AR=AI=0 THEN C:=1 AND R:=S:=0.

PROCEDURES USED: NONE.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:

```
"BEGIN""REAL"R,C,S;
"PROCEDURE"CARPOL(AR,AI,R,C,S);

"CODE"34344;
CARPOL(.3,.4,R,C,S);
OUTPUT(61,"("""("THE POLAR COORDINATES OF .3+.4*I ARE:")",/,
                "("MODULUS:")",-D.DD,/,
                "("COSINE OF ARGUMENT:")",-D.DD,/,
                "("SINE OF ARGUMENT:")",-D.DD")",R,C,S)
"END"
```

```
THE POLAR COORDINATES OF .3+.4*I ARE:
MODULUS: 0.50
COSINE OF ARGUMENT: 0.60
SINE OF ARGUMENT: 0.80
```

```
SOURCE TEXT(S);

"CODE"34340;
"REAL" "PROCEDURE" COMABS(XR,XI);"VALUE" XR,XI;"REAL" XR,XI;
"BEGIN" XR:= ABS(XR); XI:= ABS(XI);
COMABS:= "IF" XI > XR "THEN" SQRT((XR/XI)**2+1)*XI
"ELSE" "IF" XI= 0 "THEN" XR "ELSE" SQRT((XI/XR)**2+1)*XR
"END" COMABS;
          "EOP"
SOURCE TEXT(S);

"CODE"34343;
"PROCEDURE" COMSQRT(AR,AI,PR,PI);
"VALUE" AR,AI; "REAL" AR,AI,PR,PI;
"IF" AR=0 & AI= 0 "THEN" PR:= PI:=0 "ELSE"
"BEGIN" "REAL" BR,BI,H;
BR:= ABS(AR); BI:= ABS(AI);
H:= "IF" BI < BR "THEN"
("IF" BR<1 "THEN" SQRT((SQRT((BI/BR)**2+1)*.5+.5)*BR)
"ELSE" SQRT((SQRT((BI/BR)**2+1)*.125+.125)*BR)*2)
"ELSE" "IF" BI<1 "THEN" SQRT((SQRT((BR/BI)**2+1)*BI+BR)*2)*.5
"ELSE" "IF" BR+1= 1 "THEN" SQRT(BI*.5)
"ELSE" SQRT(SQRT((BR/BI)**2+1)*BI*.125+BR*.125)*2;
"IF" AR >= 0 "THEN"
"BEGIN" PR:= H; PI:= AI/H*.5 "END"
"ELSE" "BEGIN" PI:= "IF" AI >= 0 "THEN" H "ELSE" -H;
PR:= BI/H*.5
"END"
"END" COMSQRT;
          "EOP"

"CODE"34344;
"PROCEDURE"CARPOL(AR,AI,R,C,S);
"VALUE" AR,AI; "REAL" AR,AI,R,C,S;
"IF" AR=0&AI=0 "THEN"
   "BEGIN" C:=1;R:=S:=0 "END"
"ELSE" "BEGIN"
    R:="IF" ABS(AR)>ABS(AI) "THEN"
     ABS(AR)*SQRT(1+(AI/AR)**2)
    "ELSE" ABS(AI)* SQRT(1+(AR/AI)**2);
    C:=AR/R;S:=AI/R
"END"CARPOL;
          "EOP"
```

AUTHOR: C.G. VAN DER LAAN.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 730815.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS TWO PROCEDURES :
COMMUL CALCULATES THE PRODUCT OF TWO COMPLEX NUMBERS.
COMDIV CALCULATES THE QUOTIENT OF TWO COMPLEX NUMBERS.

KEYWORDS:

COMPLEX MULTIPLICATION.
COMPLEX DIVISION.

SUBSECTION COMMUL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE"COMMUL(AR,AI,BR,BI,RR,RI);
"VALUE"AR,AI,BR,BI;"REAL"AR,AI,BR,BI,RR,RI;

THE MEANING OF THE FORMAL PARAMETERS IS:
AR,AI,BR,BI:<ARITHMETIC EXPRESSION>;
ENTRY:AR,BR ARE THE REAL PARTS OF THE COMPLEX
NUMBERS AND AI,BI ARE THE IMAGINARY PARTS OF
THE COMPLEX NUMBERS;
RR,RI: <VARIABLE>;
EXIT:THE REAL PART AND THE IMAGINARY PART OF THE
RESULTING COMPLEX NUMBER ARE DELIVERED IN RR AND
RI,RESPECTIVELY.

PROCEDURES USED: NONE.

LANGUAGE: ALGOL 60.

EXAMPLE OF USE:

```
"BEGIN""REAL"R,I;
"PROCEDURE"COMMUL (AR,AI,BR,BI,RR,RI);
"CODE"34341;
COMMUL(.1,.2,.3,.4,R,I);
OUTPUT(61,"(""("(.1+.2*I)*(.3+.4*I)=")",-D.DD,+D.DD,"("*I")""")",R,I
"END"
```

(.1+.2*I)*(.3+.4*I)=-0.05+0.10*I

SUBSECTION : COMDIV.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE"COMDIV(XR,XI,YR,YI,ZR,ZI);
"VALUE"XR,XI,YR,YI;"REAL"XR,XI,YR,YI,ZR,ZI;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

XR,XI,YR,YI: <ARITHMETIC EXPRESSION>;
ENTRY: XR,YR ARE THE REAL PARTS OF THE NUMERATOR AND THE DENOMINATOR, RESPECTIVELY AND XI,YI ARE THE CORRESPONDING IMAGINARY PARTS;

ZR,ZI: <VARIABLE>;
EXIT: THE REAL PART AND THE IMAGINARY PART OF THE RESULTING COMPLEX NUMBER ARE DELIVERED IN RR AND RI, RESPECTIVELY.

RUNNING TIME:

AT MOST SIX MULTIPLICATIONS AND/OR DIVISIONS ARE USED.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE PROCEDURE IS NOT PROTECTED AGAINST DIVISION BY ZERO.

EXAMPLE OF USE:

```
"BEGIN""REAL"R,I;
"PROCEDURE"COMDIV(XR,XI,YR,YI,ZR,ZI);
"CODE"34342;
COMDIV(-.05,.1,.1,.2,R,I);
OUTPUT(61,"("""("(-.05+.1*I)/(.1+.2*I)=")",-D.DD,+D.DD,"("*I")""")"
       ,R,I)
"END"

(-.05+.1*I)/(.1+.2*I)= 0.30+0.40*I


SOURCE TEXT(S);

"CODE"34341;
"PROCEDURE" COMMUL(AR,AI,BR,BI,RR,RI);
"VALUE" AR,AI,BR,BI; "REAL" AR,AI,BR,BI,RR,RI;
"BEGIN" RR:= AR * BR - AI * BI;
RI:= AR * BI + AI * BR
"END" COMMUL;
        "EOP"

"CODE"34342;
"PROCEDURE" COMDIV(XR,XI,YR,YI,ZR,ZI);
"VALUE" XR,XI,YR,YI;"REAL" XR,XI,YR,YI,ZR,ZI;
"BEGIN" "REAL" H,D;
"IF" ABS(YI) < ABS(YR) "THEN"
"BEGIN" "IF" YI= 0 "THEN"
"BEGIN" ZR:= XR/YR; ZI:= XI/YR "END" "ELSE"
"BEGIN" H:= YI/YR; D:= H*YI + YR;
ZR:= (XR + H * XI)/D; ZI:= (XI-H*XR)/D
"END"
"END" "ELSE"
"BEGIN" H:= YR/YI; D:= H*YR + YI;
ZR:= (XR*H + XI)/D; ZI:= (XI*H - XR)/D
"END"
"END" COMDIV;
        "EOP"
```

AUTHOR: H.J.J. TE RIELE.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 740125; REVISED: 740514;

BRIEF DESCRIPTION:

THIS SECTION CONTAINS A SET OF FIVE PROCEDURES FOR
THE BASIC ARITHMETIC OPERATIONS WITH LONG INTEGERS:
LNG INT ADD EXACTLY COMPUTES THE SUM OF TWO NONNEGATIVE INTEGERS.
LNG INT SUBTRACT EXACTLY COMPUTES THE DIFFERENCE OF TWO NONNEGATIVE INTEGERS.
LNG INT MULT EXACTLY COMPUTES THE PRODUCT OF TWO NONNEGATIVE INTEGERS.
LNG INT DIVIDE EXACTLY COMPUTES THE QUOTIENT WITH REMAINDER OF TWO NONNEGATIVE INTEGERS.
LNG INT POWER EXACTLY COMPUTES U**POWER,
WHERE U IS A NONNEGATIVE LONG INTEGER AND POWER IS THE POSITIVE (SINGLE-LENGTH) EXPONENT.

KEYWORDS:

LONG INTEGER ARITHMETIC,
ADDITION,
SUBTRACTION,
MULTIPLICATION,
DIVISION WITH REMAINDER,
EXPONENTIATION.

SUBSECTION : LNG INT ADD.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNG INT ADD(U,V,SUM);
"INTEGER" "ARRAY" U,V,SUM;

THE MEANING OF THE FORMAL PARAMETERS IS:
U,V,SUM: <ARRAY IDENTIFIER>;
"INTEGER" "ARRAY" U[0:U[0]], V[0:V[0]],
SUM[0:MAX(U[0],V[0])+1];
BEFORE THE CALL OF LNG INT ADD, U AND V MUST
CONTAIN THE LONG INTEGERS TO BE ADDED;
AFTER THE CALL, SUM CONTAINS THE MULTI-LENGTH
SUM OF U AND V, WHILE U AND V REMAIN UNCHANGED.

PROCEDURES USED : NONE.

REQUIRED CENTRAL MEMORY :

EXECUTION FIELD LENGTH : 7.

RUNNING TIME :

WE GIVE A FORMULA FOR THE RUNNING TIME IN MILLISECONDS ON THE
CD CYBER 73-28 COMPUTER; THE RELATIVE PRECISION OF THE
COEFFICIENTS IS AT MOST ONE OR TWO DIGITS:
.10*MAX(U[0],V[0]) + .06*MIN(U[0],V[0]) + .56.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE : SEE LNG INT POWER (THIS SECTION).

EXAMPLE OF USE : SEE LNG INT POWER (THIS SECTION).

SUBSECTION : LNG INT SUBTRACT.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :
"PROCEDURE" LNG INT SUBTRACT (U,V,DIFFERENCE);
"INTEGER" "ARRAY" U,V,DIFFERENCE;

THE MEANING OF THE FORMAL PARAMETERS IS:
U,V,DIFFERENCE: <ARRAY IDENTIFIER>;
"INTEGER" "ARRAY" U[0:U[0]],V[0:V[0]],DIFFERENCE[0:U[0]];
BEFORE THE CALL OF LNG INT SUBTRACT, U AND V MUST
CONTAIN THE LONG INTEGERS TO BE SUBTRACTED(V FROM U);
AFTER THE CALL, DIFFERENCE CONTAINS THE MULTI-LENGTH
DIFFERENCE U-V; IF U<V THEN DIFFERENCE[0]=0
IS DELIVERED; U AND V REMAIN UNCHANGED.

PROCEDURES USED : NONE.

REQUIRED CENTRAL MEMORY:

EXECUTION FIELD LENGTH : 7.

RUNNING TIME :

WE GIVE A FORMULA FOR THE RUNNING TIME IN MILLISECONDS ON THE
CD CYBER 73-28 COMPUTER; THE RELATIVE PRECISION OF THE
COEFFICIENTS IS AT MOST ONE OR TWO DIGITS:
.10*U[0] + .06*V[0] + .64.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE : SEE LNG INT POWER (THIS SECTION).

EXAMPLE OF USE : SEE LNG INT POWER (THIS SECTION).

SUBSECTION : LNG INT MULT.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNG INT MULT(U,V,PRODUCT);
"INTEGER" "ARRAY" U,V,PRODUCT;

THE MEANING OF THE FORMAL PARAMETERS IS:
U,V,PRODUCT: <ARRAY IDENTIFIER>;
"INTEGER" "ARRAY" U[0:U[0]], V[0:V[0]],
PRODUCT[0:U[0]+V[0]];
BEFORE THE CALL OF LNG INT MULT, U AND V MUST
CONTAIN THE LONG INTEGERS TO BE MULTIPLIED;
AFTER THE CALL, PRODUCT CONTAINS THE MULTI-LENGTH
PRODUCT OF U AND V, WHILE U AND V REMAIN UNCHANGED.

PROCEDURES USED : NONE.

REQUIRED CENTRAL MEMORY :

EXECUTION FIELD LENGTH : 7.

RUNNING TIME :

WE GIVE A FORMULA FOR THE RUNNING TIME IN MILLISECONDS ON THE
CD CYBER 73-28 COMPUTER; THE RELATIVE PRECISION OF THE
COEFFICIENTS IS AT MOST ONE OR TWO DIGITS:
.18*U[0]*V[0] + .15*U[0] + .06*V[0] + .46.

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE : SEE LNG INT POWER (THIS SECTION).

EXAMPLE OF USE : SEE LNG INT POWER (THIS SECTION).

SUBSECTION : LNG INT DIVIDE.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNG INT DIVIDE(U,V,QUOTIENT,REMAINDER); "VALUE" U;
"INTEGER" "ARRAY" U,V,QUOTIENT,REMAINDER;

THE MEANING OF THE FORMAL PARAMETERS IS:
U,V,QUOTIENT,REMAINDER: <ARRAY IDENTIFIER>;
"INTEGER" "ARRAY" U[0:U[0]], V[0:V[0]],
QUOTIENT[0:U[0]-V[0]+1], REMAINDER[0:V[0]];
BEFORE THE CALL OF LNG INT DIVIDE, U MUST CONTAIN THE
DIVIDEND, V THE DIVISOR(V =/= 0);
AFTER THE CALL, THE RESULTS OF THE LONG DIVISION
OF U BY V (I.E. U//V AND U=U//V) ARE STORED INTO
QUOTIENT AND REMAINDER; U AND V REMAIN UNCHANGED.

PROCEDURES USED : NONE.

REQUIRED CENTRAL MEMORY :

11 + U[0] + (IF V[0]=1 OR U[0]<V[0] THEN 0 ELSE V[0]+1),

RUNNING TIME :

WE GIVE A FORMULA FOR THE RUNNING TIME IN MILLISECONDS ON THE
CD CYBER 73-28 COMPUTER; THE RELATIVE PRECISION OF THE
COEFFICIENTS IS AT MOST ONE OR TWO DIGITS:
IF V[0]=1 THEN (.34*U[0] + .67) ELSE IF V[1] >= 5 000 000 THEN
(.26*DIFF*V[0] + .57*DIFF + .10*V[0] + 1.8)
ELSE (.27*DIFF*V[0] + .66*DIFF + .66*V[0] + 2.0)
(HERE DIFF=U[0]-V[0]+1, I.E. THE NUMBER OF EXECUTIONS
OF THE STATEMENT, IN WHICH DIVISION OF A (V[0]+1)-PLACE
NUMBER BY A V[0]-PLACE NUMBER IS PERFORMED).

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE : SEE LNG INT POWER (THIS SECTION).

EXAMPLE OF USE : SEE LNG INT POWER (THIS SECTION).

SUBSECTION : LNG INT POWER.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNG INT POWER(U,EXPONENT,RESULT);
"VALUE" EXPONENT; "INTEGER" EXPONENT; "INTEGER" "ARRAY" U,RESULT;

THE MEANING OF THE FORMAL PARAMETERS IS;
EXPONENT: <ARITHMETIC EXPRESSION>;
THE (POSITIVE) POWER TO WHICH THE LONG INTEGER U
WILL BE RAISED;
U,RESULT: <ARRAY IDENTIFIER>;
"INTEGER" "ARRAY" U[0:U[0]], RESULT[0:U[0]*EXPONENT];
BEFORE THE CALL OF LNG INT POWER, U MUST CONTAIN
THE LONG INTEGER WHICH HAS TO BE RAISED TO THE
POWER EXPONENT;
AFTER THE CALL, RESULT CONTAINS THE VALUE OF THE
LONG INTEGER U**EXPONENT; U REMAINS UNCHANGED.

PROCEDURES USED :

LNG INT MULT = CP31202.

REQUIRED CENTRAL MEMORY :

EXECUTION FIELD LENGTH : 4 + 3 * (U[0] * EXPONENT + 1).

RUNNING TIME :

FOR THIS PROCEDURE THE TIME FORMULA IS A COMPLICATED FUNCTION OF
U[0], EXPONENT AND THE NUMBER OF ONES IN THE BINARY REPRESENTATION
OF EXPONENT, BUT ROUGHLY THE TIME IS OF THE ORDER :
(U(0)*EXPONENT)**2).

TWO TESTCASES :

EXPONENT	TIME(IN SEC.) FOR: U[0]=1	U[0]=2
20	.04	.10
40	.13	.34
100	.68	1.94
300	5.48	16.6
500	16.8	51.0

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE:

DEFINITION:
A LONG INTEGER OF LENGTH N, OR AN N-PLACE INTEGER (N>0) IS ANY NONNEGATIVE INTEGER LESS THAN BASE**N, AND GREATER THAN OR EQUAL TO BASE**(N-1), WHERE BASE IS THE (POSITIVE) RADIX OF THE POSITIONAL NOTATION, IN WHICH THE INTEGERS ARE EXPRESSED.

ALL FIVE PROCEDURES USE THE BASE 10 000 000; THIS IS THE LARGEST POWER OF 10, THE SQUARE OF WHICH CAN BE REPRESENTED EXACTLY ON THE CD CYBER 73-28 COMPUTER. IF ONE WANTS TO USE THE PROCEDURES WITH ANOTHER VALUE OF THE BASE, SAY R (NOT NECESSARILY A POWER OF 10), THEN IN THE SOURCE TEXTS OF THE PROCEDURES THE NUMBER 10 000 000 HAS TO BE REPLACED BY R
(8 TIMES IN LNG INT ADD,
 2 TIMES IN LNG INT SUBTRACT,
 2 TIMES IN LNG INT MULT AND
16 TIMES IN LNG INT DIVIDE).
MOREOVER, IN LNG INT DIVIDE THE NUMBER 9 999 999 HAS TO BE REPLACED BY THE NUMBER R - 1.

IF A[1], A[2], ..., A[N] ARE THE N "DIGITS" OF THE LONG INTEGER M OF LENGTH N (A[1] *= 0), THEN

M=((...(A[1]*BASE+A[2])*BASE+...+A[N-2])*BASE+A[N-1])*BASE+A[N],

ACCORDINGLY, A LONG INTEGER M OF LENGTH N ALWAYS WILL BE STORED INTO A CORRESPONDING "INTEGER" "ARRAY" A, THE LENGTH N WILL BE STORED INTO THE ARRAY ELEMENT A[0].

FOR THE METHOD OF THE PROCEDURES LNG INT ADD, LNG INT SUBTRACT, LNG INT MULT AND LNG INT DIVIDE, SEE [1; PP.229-248]; PROCEDURE LNG INT POWER USES THE BINARY METHOD FOR EXPONENTIATION (SEE [1; PP.398-401]).

REFERENCES:

[1]. DONALD E. KNUTH.
THE ART OF COMPUTER PROGRAMMING, VOLUME 2/ SEMINUMERICAL ALGORITHMS.
ADDISON-WESLEY PUBLISHING COMPANY, 1969.

EXAMPLE OF USE:

```
    "BEGIN"
       "PROCEDURE" LNG INT ADD(U,V,SUM); "CODE" 31200;
       "PROCEDURE" LNG INT SUBTRACT(U,V,DIFFERENCE); "CODE" 31201;
       "PROCEDURE" LNG INT MULT(U,V,PRODUCT); "CODE" 31202;
       "PROCEDURE" LNG INT DIVIDE(U,V,QUOTIENT,REMAINDER);"CODE" 31203;
       "PROCEDURE" LNG INT POWER (U,EXPONENT,RESULT); "CODE" 31204;

       "PROCEDURE" OUT(A); "INTEGER" "ARRAY" A;
       "BEGIN" "INTEGER" I,L; L:= A[0];
          OUTPUT(61,"("B6ZD,(B7D)")",(A[I],I:= 1:L));
          OUTPUT(61,"("/")")
       "END" OUT;
       "INTEGER" "ARRAY" U,V,R1,R2[0:100];

       U[0]:=5; U[1]:=333; U[2]:=U[3]:=U[4]:=U[5]:=7 000 000; OUT(U);
       V[0]:=2; V[1]:=4 444; V[2]:=4 444 444; OUT(V);

       LNG INT ADD(U,V,R1); OUT(R1);
       LNG INT SUBTRACT(U,V,R1); OUT(R1);
       LNG INT MULT(U,V,R1); OUT(R1);
       LNG INT DIVIDE(U,V,R1,R2); OUT(R1); OUT(R2);
       LNG INT POWER(V,5,R1); OUT(R1)
    "END"

DELIVERS:
        333 7000000 7000000 7000000 7000000
       4444 4444444
        333 7000000 7000000 7004445 1444444
        333 7000000 7000000 6995556 2555556
    1483111 1114073 9114221 9114221 9111110 8000000
     750825 0001650 0826575
        734 0700700
      17341 5299149 6553709 6327185 8964586 9972395 8069589 6628224
```

SOURCE TEXT(S):

```
"CODE" 31200;
"PROCEDURE" LNG INT ADD(U,V,SUM); "INTEGER""ARRAY" U,V,SUM;
"BEGIN""INTEGER" LU,LV,DIFF,CARRY,I,T,MAX;
   LU:=U[0]; LV:=V[0];
   "IF" LU >= LV "THEN"
   "BEGIN" MAX:=LU; DIFF:=LU - LV + 1; CARRY:=0;
      "FOR" I:=LU "STEP" -1 "UNTIL" DIFF "DO"
      "BEGIN" T:=U[I] + V[I-DIFF+1] + CARRY;
         CARRY:="IF" T<10 000 000 "THEN" 0 "ELSE" 1;
         SUM[I]:=T - CARRY * 10 000 000
      "END";
      "FOR" I:=DIFF - 1 "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=U[I] + CARRY;
         CARRY:="IF" T<10 000 000 "THEN" 0 "ELSE" 1;
         SUM[I]:=T - CARRY * 10 000 000
      "END"
   "END" "ELSE"
   "BEGIN" MAX:=LV; DIFF:=LV - LU + 1; CARRY:=0;
      "FOR" I:=LV "STEP" -1 "UNTIL" DIFF "DO"
      "BEGIN" T:=V[I] + U[I-DIFF+1] + CARRY;
         CARRY:="IF" T<10 000 000 "THEN" 0 "ELSE" 1;
         SUM[I]:=T - CARRY * 10 000 000
      "END";
      "FOR" I:=DIFF - 1 "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=V[I] + CARRY;
         CARRY:="IF" T<10 000 000 "THEN" 0 "ELSE" 1;
         SUM[I]:=T - CARRY * 10 000 000
      "END"
"END";
"IF" CARRY=1 "THEN"
"BEGIN" "FOR" I:=MAX "STEP" -1 "UNTIL" 1 "DO"
   SUM[I+1]:=SUM[I]; SUM[1]:=1; MAX:=MAX + 1
"END";
SUM[0]:=MAX
"END" LNG INT ADD;
      "EOP"
```

```
"CODE" 31201;
"PROCEDURE" LNG INT SUBTRACT(U,V,DIFFERENCE);
"INTEGER""ARRAY" U,V,DIFFERENCE;
"BEGIN""INTEGER" LU,LV,DIFF,I,T,J,CARRY;
   LU:=U[0]; LV:=V[0];
   "IF" LU<LV "OR" LU=LV "AND" U[1]<V[1] "THEN" DIFFERENCE[0]:=0 "ELSE"
   "BEGIN" DIFF:=LU - LV + 1; CARRY:=0;
      "FOR" I:=LU "STEP" -1 "UNTIL" DIFF "DO"
      "BEGIN" T:=U[I] - V[I-DIFF+1] + CARRY;
         CARRY:="IF" T<0 "THEN" -1 "ELSE" 0;
         DIFFERENCE[I]:=T - CARRY * 10 000 000
      "END";
      "FOR" I:=DIFF - 1 "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=U[I] + CARRY; CARRY:="IF" T<0 "THEN" -1 "ELSE" 0;
         DIFFERENCE[I]:=T - CARRY * 10 000 000
      "END";
      "IF" CARRY=-1 "THEN"
      "BEGIN" DIFFERENCE[0]:=0; "GOTO" READY "END";
      I:=0; J:=LU;
      "FOR" I:=I+1 "WHILE" DIFFERENCE[I]=0 "AND" J>1 "DO" J:=J-1;
      DIFFERENCE[0]:=J;
      "IF" J<LU "THEN"
      "FOR" I:=1 "STEP" 1 "UNTIL" J "DO"
      DIFFERENCE[I]:=DIFFERENCE[LU+I-J]
   "END";
   READY:
"END" LNG INT SUBTRACT;
        "EOP"

"CODE" 31202;
"PROCEDURE" LNG INT MULT(U,V,PRODUCT);
"INTEGER""ARRAY" U,V,PRODUCT;
"BEGIN""INTEGER" LU,LV,LUV,I,J,CARRY,T;
   LU:=U[0]; LV:=V[0]; LUV:=LU + LV;
   "FOR" I:=LU + 1 "STEP" 1 "UNTIL" LUV "DO" PRODUCT[I]:=0;
   "FOR" J:=LU "STEP" -1 "UNTIL" 1 "DO"
   "BEGIN" CARRY:=0;
      "FOR" I:=LV "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=U[J] * V[I] + PRODUCT[J+I] + CARRY;
         CARRY:=T//10 000 000; PRODUCT[J+I]:=T - CARRY * 10 000 000
      "END"; PRODUCT[J]:=CARRY
   "END";
   "IF" PRODUCT[1]=0 "THEN"
   "BEGIN" "FOR" I:=2 "STEP" 1 "UNTIL" LUV "DO"
      PRODUCT[I-1]:=PRODUCT[I]; LUV:=LUV - 1
   "END"; PRODUCT[0]:=LUV
"END" LNG INT MULT;
        "EOP"
```

```
"CODE" 31203;
"PROCEDURE" LNG INT DIVIDE(U,V,QUOTIENT,REMAINDER); "VALUE" U;
"INTEGER""ARRAY" U,V,QUOTIENT,REMAINDER;
"BEGIN""INTEGER" LU,LV,V1,DIFF,I,T,SCALE,D,Q1,J,CARRY;
   LU:=U[0]; LV:=V[0]; V1:=V[1]; DIFF:=LU - LV;

"IF" LV=1 "THEN"
"BEGIN" CARRY:=0;
   "FOR" I:=1 "STEP" 1 "UNTIL" LU "DO"
   "BEGIN" T:=CARRY * 10 000 000 + U[I]; QUOTIENT[I]:=T//V1;
      CARRY:=T - QUOTIENT[I] * V1
   "END"; REMAINDER[0]:=1; REMAINDER[1]:=CARRY;
   "IF" QUOTIENT[1]=0 "THEN"
   "BEGIN" "FOR" I:=2 "STEP" 1 "UNTIL" LU "DO"
      QUOTIENT[I-1]:=QUOTIENT[I];
      QUOTIENT[0]:=LU - ("IF" LU=1 "THEN" 0 "ELSE" 1)
   "END" "ELSE" QUOTIENT[0]:=LU
"END" LV=1
"ELSE"

"IF" LU<LV "THEN"
"BEGIN" QUOTIENT[0]:=1; QUOTIENT[1]:=0;
   "FOR" I:=0 "STEP" 1 "UNTIL" LU "DO" REMAINDER[I]:=U[I]
"END" LU<LV
"ELSE"

"BEGIN""INTEGER""ARRAY" A[0:LV];
   SCALE:=10 000 000//(V1+1);
   "IF" SCALE>1 "THEN"
   "BEGIN""COMMENT" NORMALIZE U; CARRY:=0;
      "FOR" I:=LU "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=SCALE * U[I] + CARRY; CARRY:=T//10 000 000;
         U[I]:=T - CARRY * 10 000 000
      "END"; U[0]:=CARRY;
      "COMMENT" NORMALIZE V; CARRY:=0;
      "FOR" I:=LV "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=SCALE * V[I] + CARRY; CARRY:=T//10 000 000;
         V[I]:=T - CARRY * 10 000 000
      "END"; V1:=V[1]
   "END" NORMALIZATION
   "ELSE" U[0]:=0;                                    "COMMENT"
```

```
"COMMENT" COMPUTE QUOTIENT AND REMAINDER;
"FOR" I:=0 "STEP" 1 "UNTIL" DIFF "DO"
"BEGIN" D:=U[I] * 10 000 000 + U[I+1];
   Q1:="IF" U[I]=V1 "THEN" 9 999 999 "ELSE" D//V1;
   "IF" V[2] * Q1 > (D-Q1*V1) * 10 000 000 + U[I+2] "THEN"
   "BEGIN" Q1:=Q1 - 1;
      "IF" V[2]*Q1>(D-Q1*V1)*10 000 000+U[I+2] "THEN" Q1:=Q1-1
   "END";

   "COMMENT" U[I:I+LV]:=U[I:I+LV] - Q1 * V[1:LV];
   CARRY:=0;
   "FOR" J:=LV "STEP" -1 "UNTIL" 1 "DO"
   "BEGIN" T:=Q1 * V[J] + CARRY; CARRY:=T//10 000 000;
      A[J]:=T - CARRY * 10 000 000
   "END"; A[0]:=CARRY;
   CARRY:=0;
   "FOR" J:=LV "STEP" -1 "UNTIL" 0 "DO"
   "BEGIN" T:=U[I+J] - A[J] + CARRY; CARRY:="IF" T<0 "THEN" -1
      "ELSE" 0; U[I+J]:=T - CARRY * 10 000 000
   "END";

   "COMMENT" IF CARRY=-1 THEN Q1 IS ONE TOO LARGE,
   AND V MUST BE ADDED BACK TO U[I:I+LV];
   "IF" CARRY=-1 "THEN"
   "BEGIN" Q1:=Q1 - 1; CARRY:=0;
      "FOR" J:=LV "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN" T:=U[I+J] + V[J] + CARRY; CARRY:="IF" T<10 000 000
         "THEN" 0 "ELSE" 1; U[I+J]:=T - CARRY * 10 000 000
      "END"
   "END"; QUOTIENT[I]:=Q1
"END" I;

"COMMENT" CORRECT STORAGE OF QUOTIENT;
"IF" QUOTIENT[0] ^= 0 "THEN"
"BEGIN" "FOR" I:=DIFF "STEP" -1 "UNTIL" 0 "DO"
   QUOTIENT[I+1]:=QUOTIENT[I]; QUOTIENT[0]:=DIFF + 1
"END" "ELSE"
"IF" QUOTIENT[1] ^= 0 "THEN" QUOTIENT[0]:=DIFF "ELSE"
"BEGIN" "FOR" I:=1 "STEP" 1 "UNTIL" DIFF - 1 "DO"
   QUOTIENT[I]:=QUOTIENT[I+1]; QUOTIENT[0]:=DIFF - 1
"END";                                                    "COMMENT"
```

```
        "COMMENT" REMAINDER :=U[DIFF+1:LU]//SCALE;
        "IF" SCALE>1 "THEN"
        "BEGIN" CARRY:=0;
            "FOR" I:=1 "STEP" 1 "UNTIL" LV "DO"
            "BEGIN" T:=CARRY * 10 000 000 + U[DIFF+I];
                REMAINDER[I]:=T//SCALE; CARRY:=T - REMAINDER[I] * SCALE
            "END"
        "END" "ELSE"
        "FOR" I:=1 "STEP" 1 "UNTIL" LV "DO" REMAINDER[I]:=U[DIFF+I];

        "COMMENT" CORRECT STORAGE OF REMAINDER;
        I:=0; J:=LV;
        "FOR" I:=I+1 "WHILE" REMAINDER[I]=0 "AND" J>1 "DO" J:=J-1;
        REMAINDER[0]:=J;
        "IF" J<LV "THEN"
        "FOR" I:=1 "STEP" 1 "UNTIL" J "DO"
        REMAINDER[I]:=REMAINDER[LV + I - J];

        "COMMENT" UNNORMALIZE THE DIVISOR V;
        "IF" SCALE>1 "THEN"
        "BEGIN" CARRY:=0;
            "FOR" I:=1 "STEP" 1 "UNTIL" LV "DO"
            "BEGIN" T:=CARRY * 10 000 000 + V[I];
                V[I]:=T//SCALE; CARRY:=T - V[I] * SCALE
            "END"
        "END"
    "END"
"END" LNG INT DIVIDE;
        "EOP"

"CODE" 31204;
"PROCEDURE" LNG INT POWER(U,EXPONENT,RESULT);
"VALUE" EXPONENT; "INTEGER" EXPONENT; "INTEGER""ARRAY" U,RESULT;
"BEGIN""INTEGER" MAX,I,N;
    "PROCEDURE" LNG INT MULT(U,V,PRODUCT); "CODE" 31202;
    MAX:=U[0] * EXPONENT;
    "BEGIN""INTEGER""ARRAY" Y,Z,H[0:MAX];
        "COMMENT" Y:=1, Z:=U;
        Y[0]:=Y[1]:=1;
        "FOR" I:=U[0] "STEP" -1 "UNTIL" 0 "DO" Z[I]:=U[I];
    HALVE: N:=EXPONENT//2; "IF" N+N=EXPONENT "THEN" "GOTO" SQUARE Z;
        LNG INT MULT(Y,Z,H);
        "FOR" I:=H[0] "STEP" -1 "UNTIL" 0 "DO" Y[I]:=H[I];
        "IF" N=0 "THEN" "GOTO" READY;
    SQUARE Z: LNG INT MULT(Z,Z,H);
        "FOR" I:=H[0] "STEP" -1 "UNTIL" 0 "DO" Z[I]:=H[I];
        EXPONENT:=N; "GOTO"HALVE;
    READY: "FOR" I:=Y[0] "STEP" -1 "UNTIL" 0 "DO" RESULT[I]:=Y[I]
    "END"
"END" LNG INT POWER;
        "EOP"
```

AUTHORS: D.T.WINTER(A-D,F-I), T.J.DEKKER(E,J)

CONTRIBUTOR: J.KOOPMAN(E,J)

INSTITUTES: MATHEMATICAL CENTRE,UNIVERSITY OF AMSTERDAM.

RECEIVED: 770328

BRIEF DESCRIPTION:

THIS SECTION CONTAINS PROCEDURES FOR THE ELEMENTARY OPERATIONS IN DOUBLE PRECISION ARITHMETIC.

A. DPADD ADDS TWO SINGLE PRECISION NUMBERS TO A DOUBLE PRECISION SUM.
B. DPSUB SUBTRACTS TWO SINGLE PRECISION NUMBERS TO A DOUBLE PRECISION DIFFERENCE.
C. DPMUL MULTIPLIES TWO SINGLE PRECISION NUMBERS TO A DOUBLE PRECISION PRODUCT.
D. DPDIV DIVIDES TWO SINGLE PRECISION NUMBERS TO A DOUBLE PRECISION QUOTIENT.
E. DPPOW COMPUTES A**EXPON IN DOUBLE PRECISION,WHERE A IS A SINGLE PRECISION REAL NUMBER AND EXPON THE INTEGER EXPONENT.
F. LNGADD ADDS TWO DOUBLE PRECISION NUMBERS.
G. LNGSUB SUBTRACTS TWO DOUBLE PRECISION NUMBERS.
H. LNGMUL MULTIPLIES TWO DOUBLE PRECISION NUMBERS.
I. LNGDIV DIVIDES TWO DOUBLE PRECISION NUMBERS.
J. LNGPOW COMPUTES (A,AA)**EXPON IN DOUBLE PRECISION,WHERE (A,AA) IS A DOUBLE PRECISION REAL NUMBER AND EXPON THE INTEGER EXPONENT.

KEYWORDS:
DOUBLE PRECISION ARITHMETIC
EXPONENTIATION.

LANGUAGE: COMPASS(A-D,F-I),ALGOL 60(E,J)

METHOD AND PERFORMANCE:

THE PROCEDURES A=D,F=I USE THE HARDWARE FUNCTIONS FOR DOUBLE PRECISION THAT ARE AVAILIBLE ON THE CYBER.
THE PROCEDURES LNG ADD, LNG SUB, LNG MUL AND LNG DIV CHECK THE INPUT PARAMETERS (A,AA) AND (B,BB) FOR CORRECTNESS. A HEAD/TAIL PAIR IS A CORRECT DOUBLE PRECISION PARAMETER IN THE FOLLOWING CASES:
A) THE TAIL IS ZERO;
B) THE EXPONENT IN THE BINARY REPRESENTATION OF THE TAIL IS 40 LESS THAN THE EXPONENT OF THE HEAD.
AN OUTPUT PARAMETER OF THESE PROCEDURES ALWAYS IS A CORRECT DOUBLE PRECISION NUMBER. IF AN INPUT PARAMETER IS NOT CORRECT, THE ERROR MESSAGE "OP PARAMETER TAIL ERROR" WILL BE ISSUED.

BOTH PROCEDURES E AND J MAKE USE OF THE BINARY REPRESENTATION OF THE INTEGER EXPONENT. IF X DENOTES THE NUMBER THAT IS TO BE EXPONENTIATED, THE PROCEDURES E AND J RUN AS FOLLOWS:
THE SEQUENCE X,X**2,X**4,X**8,... IS FORMED WHILE SIMULTANEOUSLY THE BINARY REPRESENTATION OF THE EXPONENT IS CHECKED; WHEN THE I-TH DIGIT EQUALS ONE, THE FACTOR X**(2*(I-1)) IS TAKEN INTO ACCOUNT.

EXAMPLE OF USE:

SEE THE PROCEDURE LNGREATUDECI (SECTION 1.5.3).

SUBSECTION: DP ADD

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" DP ADD(A, B, C, CC);
"VALUE" A, B; "REAL" A, B, C, CC;
"CODE" 31101;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,B: <ARITHMETIC EXPRESSIONS>;
THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE DOUBLE PRECISION RESULT OF A+B.

SUBSECTION: DP SUB

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" DP SUB(A, B, C, CC);
"VALUE" A, B; "REAL" A, B, C, CC;
"CODE" 31102;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,B: <ARITHMETIC EXPRESSIONS>;
THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE DOUBLE PRECISION RESULT OF A-B.

SUBSECTION: DP MUL

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" DP MUL(A, B, C, CC);
"VALUE" A, B; "REAL" A, B, C, CC;
"CODE" 31103;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,B: <ARITHMETIC EXPRESSIONS>;
THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE DOUBLE PRECISION RESULT OF A*B.

SUBSECTION: DP DIV

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" DP DIV(A, B, C, CC);
"VALUE" A, B; "REAL" A, B, C, CC;
"CODE" 31104;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,B: <ARITHMETIC EXPRESSIONS>;
THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE DOUBLE PRECISION RESULT OF A/B.

SUBSECTION: DP POW.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE"DP POW(A ,EXPON ,C ,CC);
"VALUE"A,EXPON;"INTEGER"EXPON;"REAL"A,C,CC;
"CODE"31109;

THE MEANING OF THE FORMAL PARAMETERS IS:
A : <ARITHMETIC EXPRESSION>;
THE NUMBER THAT IS TO BE EXPONENTIATED;
EXPON : <ARITHMETIC EXPRESSION>;
THE (INTEGER) POWER TO WHICH A WILL BE RAISED;
C , CC : <REAL VARIABLES>;
EXIT: THE HEAD (C) AND TAIL (CC) OF THE DOUBLE
PRECISION RESULT A**EXPON.

PROCEDURES USED:

LNG POW = CP31110.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO LN(EXPON).

SUBSECTION: LNG ADD

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" LNG ADD(A, AA, B, BB, C, CC);
"VALUE" A, AA, B, BB; "REAL" A, AA, B, BB, C, CC;
"CODE" 31105;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,AA,B,BB: <ARITHMETIC EXPRESSIONS>;
THE HEADS (A AND B) AND THE TAILS (AA AND BB) OF THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE RESULT (A,AA)+(B,BB).

SUBSECTION: LNG SUB

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" LNG SUB(A, AA, B, BB, C, CC);
"VALUE" A, AA, B, BB; "REAL" A, AA, B, BB, C, CC;
"CODE" 31106;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,AA,B,BB: <ARITHMETIC EXPRESSIONS>;
THE HEADS (A AND B) AND THE TAILS (AA AND BB) OF THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE RESULT (A,AA)-(B,BB).

SUBSECTION: LNG MUL

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" LNG MUL(A, AA, B, BB, C, CC);
"VALUE" A, AA, B, BB; "REAL" A, AA, B, BB, C, CC;
"CODE" 31107;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,AA,B,BB: <ARITHMETIC EXPRESSIONS>;
THE HEADS (A AND B) AND THE TAILS (AA AND BB) OF THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE RESULT (A,AA)*(B,BB).

SUBSECTION: LNG DIV

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" LNG DIV(A, AA, B, BB, C, CC);
"VALUE" A, AA, B, BB; "REAL" A, AA, B, BB, C, CC;
"CODE" 31108;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,AA,B,BB: <ARITHMETIC EXPRESSIONS>;
THE HEADS (A AND B) AND THE TAILS (AA AND BB) OF THE OPERANDS;
C,CC: <REAL VARIABLES>;
THE HEAD AND TAIL OF THE RESULT (A,AA)/(B,BB).

SUBSECTION: LNG POW.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE"LNG POW(A ,AA ,EXPON ,C ,CC);
"VALUE"A,AA,EXPON;"INTEGER"EXPON;"REAL"A,AA,C,CC;
"CODE"31110;

THE MEANING OF THE FORMAL PARAMETERS IS:
A,AA : <ARITHMETIC EXPRESSIONS>;
THE HEAD (A) AND TAIL (AA) OF THE NUMBER THAT
IS TO BE EXPONENTIATED;
EXPON : <ARITHMETIC EXPRESSION>;
THE (INTEGER) POWER TO WHICH (A,AA) WILL BE RAISED;
C,CC : <REAL VARIABLES>;
EXIT: THE HEAD (C) AND TAIL (CC) OF THE DOUBLE
PRECISION RESULT (A,AA)**EXPON.

PROCEDURES USED:

LNG MUL = CP31107.
LNG DIV = CP31108.

RUNNING TIME:

ROUGHLY PROPORTIONAL TO LN(EXPON).

```
SOURCE TEXT(S);

     AS IT IS NOT POSIBLE TO SIMULATE THE PROCEDURES A=D,F=I IN ALGOL 60
     THEIR SOURCE TEXTS ARE NOT GIVEN. FOR THE COMPASS SOURCE TEXTS,
     ONE IS REFERRED TO APPENDIX C OF THE MANUALDOCUMENTATION.

"CODE"31109;
"PROCEDURE"DP POW(A,EXPON,C,CC);
"VALUE"A,EXPON;"INTEGER"EXPON;"REAL"A,C,CC;
"BEGIN""PROCEDURE"LNG POW(A,AA,EXPON,C,CC);"CODE"31110;
        LNG POW(A,0,EXPON,C,CC)
"END" DP POW;
   "EOP"


"CODE"31110;
"PROCEDURE"LNG POW(A,AA,EXPON,C,CC);
"VALUE"A,AA,EXPON;"INTEGER"EXPON;"REAL"A,AA,C,CC;
"BEGIN""INTEGER"OLDEX,NEWEX;"REAL"D,DD;
        "PROCEDURE"LNG MUL(A,AA,B,BB,C,CC);"CODE"31107;
        "PROCEDURE"LNG DIV(A,AA,B,BB,C,CC);"CODE"31108;
        D:=A;DD:=AA;C:=1;CC:=0;NEWEX:=ABS(EXPON);
        "FOR"OLDEX:=NEWEX"WHILE"NEWEX"=0"DO"
        "BEGIN"NEWEX:=OLDEX//2;
                "IF"NEWEX+NEWEX"="OLDEX
                "THEN"LNG MUL(C,CC,D,DD,C,CC);
                "IF"NEWEX"=0
                "THEN"LNG MUL(D,DD,D,DD,D,DD)
        "END";
        "IF"EXPON<0"THEN"LNG DIV(1,0,C,CC,C,CC)
"END" LNG POW;
   "EOP"
```

AUTHORS: J. WOLLESWINKEL AND D. WINTER.

CONTRIBUTOR: J. C. P. BUS.

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 750501.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FOURTEEN PROCEDURES FOR CALCULATING, WITH DUOBLE LENGTH ARITHMETIC, THE (SCALAR) PRODUCTS OF SINGLE LENGTH VECTORS AND MATRICES.

LNGVECVEC: CALCULATES THE SCALAR PRODUCT OF TWO VECTORS GIVEN IN ONE-DIMENSIONAL ARRAYS;

LNGMATVEC: CALCULATES THE SCALAR PRODUCT OF A VECTOR GIVEN IN A ONE-DIMENSIONAL ARRAY AND A ROW OF A MATRIX GIVEN IN A TWO DIMENSIONAL ARRAY;

LNGTAMVEC: CALCULATES THE SCALAR PRODUCT OF A VECTOR GIVEN IN A ONE-DIMENSIONAL ARRAY AND A COLUMN OF A MATRIX GIVEN IN A TWO-DIMENSIONAL ARRAY;

LNGMATMAT: CALCULATES THE SCALAR PRODUCT OF A ROW OF A MATRIX AND A COLUMN OF ANOTHER MATRIX, WHICH ARE BOTH GIVEN IN TWO-DIMENSIONAL ARRAYS;

LNGTAMMAT: CALCULATES THE SCALAR PRODUCT OF COLUMNS OF MATRICES, WHICH ARE BOTH GIVEN IN TWO-DIMENSIONAL ARRAYS;

LNGMATTAM: CALCULATES THE SCALAR PRODUCT OF ROWS OF MATRICES, WHICH ARE BOTH GIVEN IN TWO-DIMENSIONAL ARRAYS;

LNGSEQVEC: CALCULATES THE SCALAR PRODUCT OF TWO VECTORS GIVEN IN ONE-DIMENSIONAL ARRAYS, WHERE THE MUTUAL SPACINGS BETWEEN THE INDICES OF THE FIRST VECTOR CHANGE LINEARLY;

LNGSCAPRD1: CALCULATES THE SCALAR PRODUCT OF TWO VECTORS GIVEN IN ONE-DIMENSIONAL ARRAYS, WHERE THE SPACINGS OF BOTH VECTORS ARE CONSTANT;

LNGSYMMATVEC: CALCULATES THE SCALAR PRODUCT OF A VECTOR GIVEN IN A ONE-DIMENSIONAL ARRAY AND A ROW OF A SYMMETRIC MATRIX, WHOSE UPPER TRIANGLE IS STORED COLUMN-WISE IN A ONE-DIMENSIONAL ARRAY;

THE ABOVE PROCEDURES HAVE THE POSSIBILITY OF ADDING A DOUBLE LENGTH SCALAR TO THE CALCULATED SCALAR PRODUCT;

LNGFULMATVEC: CALCULATES THE VECTOR A * B, WHERE A IS A GIVEN MATRIX AND B IS A VECTOR;
LNGFULTAMVEC: CALCULATES THE VECTOR A' * B, WHERE A' IS THE TRANSPOSED OF THE MATRIX A AND B IS A VECTOR;
LNGFULSYMMATVEC: CALCULATES THE VECTOR A * B, WHERE A IS A SYMMETRIC MATRIX, WHOSE UPPERTRIANGLE IS STORED COLUMNWISE IN A ONE-DIMENSIONAL ARRAY, AND B IS A VECTOR;
LNGRESVEC: CALCULATES THE RESIDUAL VECTOR A * B + X * C, WHERE A IS A GIVEN MATRIX, B AND C ARE VECTORS AND X IS A SCALAR;
LNGSYMRESVEC: CALCULATES THE RESIDUAL VECTOR A * B + X * C, WHERE A IS A SYMMETRIC MATRIX, WHOSE UPPERTRIANGLE IS STORED COLUMNWISE IN A ONE-DIMENSIONAL ARRAY, B AND C ARE VECTORS AND, X IS A SCALAR.

IN THIS SECTION (X, XX) DENOTES A DOUBLE-LENGTH NUMBER WITH HEAD X AND TAIL XX (SEE METHOD AND PERFORMANCE).

KEYWORDS:

ELEMENTARY OPERATIONS,
VECTOR OPERATIONS,
SCALAR PRODUCTS,
DOUBLE-LENGTH ARITHMETIC.

SUBSECTION: LNGVECVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGVECVEC(L, U, SHIFT, A, B, C, CC, D, DD);
"VALUE" L, U, SHIFT, C, CC; "INTEGER" L, U, SHIFT;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34410;

THE MEANING OF THE FORMAL PARAMETERS IS:
L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;
U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;
SHIFT: <ARITHMETIC EXPRESSION>;
THE INDEX-SHIFTING PARAMETER OF THE VECTOR GIVEN IN B;
A: <ARRAY IDENTIFIER>;
ONE OF THE VECTORS SHOULD BE GIVEN IN ARRAY A[L:U];
B: <ARRAY IDENTIFIER>;
THE OTHER VECTOR SHOULD BE GIVEN IN ARRAY B[L + SHIFT : U + SHIFT];
C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);
D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD):= (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[L:U] AND B[L + SHIFT : U + SHIFT].

LANGUAGE: COMPASS.

SUBSECTION: LNGMATVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGMATVEC(L, U, I, A, B, C, CC, D, DD);
"VALUE" L, U, I, C, CC; "INTEGER" L, U, I;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34411;

THE MEANING OF THE FORMAL PARAMETERS IS:

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE ROW-VECTOR GIVEN IN ARRAY A;

A: <ARRAY IDENTIFIER>;
THE ROW-VECTOR SHOULD BE GIVEN IN ARRAY A[I:I, L:U];

B: <ARRAY IDENTIFIER>;
THE VECTOR SHOULD BE GIVEN IN ARRAY B[L:U];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD):= (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[I:I, L:U] AND B[L:U].

LANGUAGE: COMPASS.

SUBSECTION: LNGTAMVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGTAMVEC(L, U, I, A, B, C, CC, D, DD);
"VALUE" L, U, I, C, CC; "INTEGER" L, U, I;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34412;

THE MEANING OF THE FORMAL PARAMETERS IS:

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE COLUMN-VECTOR GIVEN IN ARRAY A;

A: <ARRAY IDENTIFIER>;
THE COLUMN-VECTOR SHOULD BE GIVEN IN ARRAY A[L:U, I:I];

B: <ARRAY IDENTIFIER>;
THE VECTOR SHOULD BE GIVEN IN ARRAY B[L:U];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD):= (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[L:U, I:I] AND B[L:U].

LANGUAGE: COMPASS.

SUBSECTION: LNGMATMAT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGMATMAT(L, U, I, J, A, B, C, CC, D, DD);
"VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34413;

THE MEANING OF THE FORMAL PARAMETERS IS:

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE ROW-VECTOR GIVEN IN ARRAY A;

J: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE COLUMN-VECTOR GIVEN IN ARRAY B;

A: <ARRAY IDENTIFIER>;
THE ROW-VECTOR SHOULD BE GIVEN IN ARRAY A[I:I, L:U];

B: <ARRAY IDENTIFIER>;
THE COLUMN-VECTOR SHOULD BE GIVEN IN ARRAY B[L:U, J:J];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD) := (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[I:I, L:U] AND B[L:U, J:J].

LANGUAGE: COMPASS.

SUBSECTION: LNGTAMMAT.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGTAMMAT(L, U, I, J, A, B, C, CC, D, DD);
"VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34414;

THE MEANING OF THE FORMAL PARAMETERS IS;

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE COLUMN-VECTOR GIVEN IN ARRAY A;

J: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE COLUMN-VECTOR GIVEN IN ARRAY B;

A: <ARRAY IDENTIFIER>;
ONE OF THE COLUMN-VECTORS SHOULD BE GIVEN IN ARRAY A[L:U, I:I];

B: <ARRAY IDENTIFIER>;
THE OTHER COLUMN-VECTOR SHOULD BE GIVEN IN ARRAY B[L:U, J:J];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD):= (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[L:U, I:I] AND B[L:U, J:J].

LANGUAGE: COMPASS.

SUBSECTION: LNGMATTAM.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGMATTAM(L, U, I, J, A, B, C, CC, D, DD);
"VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34415;

THE MEANING OF THE FORMAL PARAMETERS IS:

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

I: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE ROW-VECTOR GIVEN IN ARRAY A;

J: <ARITHMETIC EXPRESSION>;
THE INDEX OF THE ROW-VECTOR GIVEN IN ARRAY B;

A: <ARRAY IDENTIFIER>;
ONE OF THE ROW-VECTORS SHOULD BE GIVEN IN ARRAY A[I:I,L:U];

B: <ARRAY IDENTIFIER>;
THE OTHER ROW-VECTOR SHOULD BE GIVEN IN ARRAY B[J:J, L:U];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD):= (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[I:I, L:U] AND B[J:J, L:U].

LANGUAGE: COMPASS.

SUBSECTION: LNGSEQVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGSEQVEC(L, U, IL, SHIFT, A, B, C, CC, D, DD);
"VALUE" L, J, IL, SHIFT, C, CC; "INTEGER" L, U, IL, SHIFT;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34416;

THE MEANING OF THE FORMAL PARAMETERS IS:

L: <ARITHMETIC EXPRESSION>;
THE LOWER BOUND OF THE RUNNING SUBSCRIPT;

U: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT;

IL: <ARITHMETIC EXPRESSION>;
THE FIRST INDEX OF THE VECTOR GIVEN IN ARRAY A;

SHIFT: <ARITHMETIC EXPRESSION>;
THE INDEX-SHIFTING PARAMETER OF THE VECTOR GIVEN IN B;

A: <ARRAY IDENTIFIER>;
ONE OF THE VECTORS SHOULD BE GIVEN IN ARRAY A[P:Q], WHERE
P = MIN(IL + (J + L - 1) * (J - 1) // 2; J = L, ..., U) AND
Q = MAX(IL + (J + L - 1) * (J - 1) // 2; J = L, ..., U);

B: <ARRAY IDENTIFIER>;
THE OTHER VECTOR SHOULD BE GIVEN IN ARRAY
B[L + SHIFT : U + SHIFT];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD) := (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[P:Q] (SEE THE MEANING OF PARAMETER A) AND B[L + SHIFT: U + SHIFT], WHERE THE ELEMENTS OF THE FIRST VECTOR ARE A[IL + (J + L - 1) * (J - 1) // 2], J = L, ..., U.

LANGUAGE: COMPASS.

SUBSECTION: LNGSCAPRD1.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGSCAPRD1(LA, SA, LB, SB, N, A, B, C, CC, D, DD);
"VALUE" LA, SA, LB, SB, N, C, CC; "INTEGER" LA, SA, LB, SB, N;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34417;

THE MEANING OF THE FORMAL PARAMETERS IS:

LA: <ARITHMETIC EXPRESSION>;
THE FIRST INDEX OF THE VECTOR GIVEN IN ARRAY A;

SA: <ARITHMETIC EXPRESSION>;
THE SPACING OF THE VECTOR GIVEN IN ARRAY A;

LB: <ARITHMETIC EXPRESSION>;
THE FIRST INDEX OF THE VECTOR GIVEN IN ARRAY B;

SB: <ARITHMETIC EXPRESSION>;
THE SPACING OF THE VECTOR GIVEN IN ARRAY B;

N: <ARITHMETIC EXPRESSION>;
THE UPPER BOUND OF THE RUNNING SUBSCRIPT; IF N < 1, THEN ON EXIT THE RESULT (D, DD) WILL SATISFY: (D, DD) = (C, CC);

A: <ARRAY IDENTIFIER>;
ONE OF THE VECTORS SHOULD BE GIVEN IN ARRAY A[MIN(LA, LA + (N - 1) * SA) : MAX(LA, LA + (N - 1) * SA)];

B: <ARRAY IDENTIFIER>;
THE OTHER VECTOR SHOULD BE GIVEN IN ARRAY B[MIN(LB, LB + (N - 1) * SB) : MAX(LB, LB + (N - 1) * SB)];

C, CC: <ARITHMETIC EXPRESSION>;
THE HEAD AND TAIL OF THE DOUBLE-LENGTH SCALAR THAT HAS TO BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF CC IS NOT A TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD AND PERFORMANCE);

D, DD: <REAL VARIABLE>;
EXIT: THE HEAD AND TAIL OF THE CALCULATED DOUBLE-LENGTH RESULT.

DATA AND RESULTS:

(D, DD) := (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[MIN(LA, LA + (N - 1) * SA):MAX(LA, LA + (N - 1) * SA)] AND B[MIN(LB, LB + (N - 1) * SB) : MAX(LB, LB + (N - 1) * SB)], WHERE THE ELEMENTS OF THE VECTORS ARE A[LA + (J - 1) * SA] AND B[LB + (J - 1) * SB], FOR J = 1, ..., N.

LANGUAGE: COMPASS.

SUBSECTION: LNGSYMMATVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGSYMMATVEC(L, U, I, A, B, C, CC, D, DD);
"VALUE" L, U, I, C, CC; "INTEGER" L, U, I;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"CODE" 34418;

THE MEANING OF THE FORMAL PARAMETERS IS:
L:      <ARITHMETIC EXPRESSION>;
        THE LOWER BOUND OF THE RUNNING SUBSCRIPT;
U:      <ARITHMETIC EXPRESSION>;
        THE UPPER BOUND OF THE RUNNING SUBSCRIPT;
I:      <ARITHMETIC EXPRESSION>;
        THE INDEX OF THE ROW  OF THE SYMMETRIC MATRIX,  WHOSE UPPER
        TRIANGLE IS STORED COLUMN-WISE IN THE ONE-DIMENSIONAL ARRAY
        A;
A:      <ARRAY IDENTIFIER>;
        THE ROW OF THE SYMMETRIC MATRIX  SHOULD BE GIVEN  IN  ARRAY
        A[P:Q], WHERE, IF I > L THEN P = (I - 1) * I // 2 + L  ELSE
        P  =  (L - 1) * L  //  2  +  I,     AND     IF  I > U     THEN
        Q = (I - 1) * I // 2 + U  ELSE  Q = (U - 1) * U // 2 + I;
B:      <ARRAY IDENTIFIER>;
        THE VECTOR SHOULD BE GIVEN IN ARRAY B[L:U];
C, CC:  <ARITHMETIC EXPRESSION>;
        THE HEAD AND TAIL OF  THE DOUBLE-LENGTH SCALAR  THAT HAS TO
        BE ADDED TO THE CALCULATED SCALAR PRODUCT; IF  CC  IS NOT A
        TAIL TO C THEN AN ERROR MESSAGE WILL BE PRINTED (SEE METHOD
        AND PERFORMANCE);
D, DD:  <REAL VARIABLE>;
        EXIT:   THE HEAD AND TAIL  OF  THE CALCULATED DOUBLE-LENGTH
                RESULT.
```

PROCEDURES USED:

```
DPMUL  = CP31103;
LNGADD = CP31105.
```

DATA AND RESULTS:

(D, DD) := (C, CC) + THE SCALAR PRODUCT OF THE VECTORS GIVEN IN THE ARRAYS A[P:Q] (SEE THE MEANING OF PARAMETER A) AND B[L:U], WHERE THE ELEMENTS OF THE FIRST VECTOR ARE: IF L < I THEN A[(I - 1) * I // 2 + J], J = L, ..., MIN(U, I - 1) AND A[(J - 1) * J // 2 + I], J = I, ..., U, RESPECTIVELY, OTHERWISE A[(J - 1) * J // 2 + I], J = L, ..., U; THE VALUES OF L, U, I SHOULD BE POSITIVE.

LANGUAGE: ALGOL 60.

SUBSECTION: LNGFULMATVEC.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE" LNGFULMATVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "CODE" 31505;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    LR, UR: <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER BOUND OF THE ROW-INDEX;
    LC, UC: <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER BOUND OF THE COLUMN-INDEX;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
    B:      <ARRAY IDENTIFIER>;
            "ARRAY" B[LC:UC]; THE VECTOR;
    C:      <ARRAY IDENTIFIER>;
            "ARRAY" C[LR:UR];
            THE RESULT A * B, CALCULATED WITH DOUBLE LENGTH ARITHMETIC,
            IS DELIVERED IN C.
```

LANGUAGE: COMPASS.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: LNGFULTAMVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGFULTAMVEC(LR, UR, LC, UC, A, B, C);
"VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
"ARRAY" A, B, C;
"CODE" 31506;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW-INDEX;
LC, UC: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE COLUMN-INDEX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[LR:UR]; THE VECTOR;
C: <ARRAY IDENTIFIER>;
"ARRAY" C[LC:UC];
THE RESULT A' * B, CALCULATED WITH DOUBLE LENGTH ARITHMETIC,
IS DELIVERED IN C; HERE A' DENOTES THE TRANSPOSED OF THE
MATRIX A.

LANGUAGE: COMPASS.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: LNGFULSYMMATVEC.

CALLING SEQUENCE:

```
THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGFULSYMMATVEC(LR, UR, LC, UC, A, B, C);
"VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
"ARRAY" A, B, C;
"CODE" 31507;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE ROW-INDEX; LR >= 1;
LC, UC: <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER BOUND OF THE COLUMN-INDEX; LC >= 1;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[L:U], WHERE:
        L = MIN(LR * (LR - 1) // 2 + LC, LC * (LC - 1) // 2 + LR),
        U = MAX(UR * (UR - 1) // 2 + UC, UC * (UC - 1) // 2 + UR)
        AND THE (I,J)-TH ELEMENT OF THE SYMMETRIC MATRIX SHOULD BE
        GIVEN IN A[J * (J - 1) // 2 + I];
B:      <ARRAY IDENTIFIER>;
        "ARRAY" B[LC:UC]; THE VECTOR;
C:      <ARRAY IDENTIFIER>;
        "ARRAY" C[LR:UR];
        THE RESULT A * B, CALCULATED WITH DOUBLE LENGTH ARITHMETIC,
        IS DELIVERED IN C.
```

PROCEDURES USED:

LNGSYMMATVEC = CP34418.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: LNGRESVEC.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:
"PROCEDURE" LNGRESVEC(LR, UR, LC, UC, A, B, C, X);
"VALUE" LR, UR, LC, UC, B, X; "INTEGER" LR, UR, LC, UC;
"REAL" X; "ARRAY" A, B, C;
"CODE" 31508;

THE MEANING OF THE FORMAL PARAMETERS IS:
LR, UR: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE ROW-INDEX;
LC, UC: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER BOUND OF THE COLUMN-INDEX;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[LR:UR,LC:UC]; THE MATRIX;
B: <ARRAY IDENTIFIER>;
"ARRAY" B[LC:UC]; THE VECTOR;
X: <ARITHMETIC EXPRESSION>;
THE VALUE OF THE MULTIPLYING SCALAR;
C: <ARRAY IDENTIFIER>;
"ARRAY" C[LR:UR];
THE RESULT A * B + X * C, CALCULATED WITH DOUBLE LENGTH ARITHMETIC, IS OVERWRITTEN ON C.

LANGUAGE: COMPASS.

METHOD AND PERFORMANCE:

ELEMENTARY.

SUBSECTION: LNGSYMRESVEC.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE" LNGSYMRESVEC(LR, UR, LC, UC, A, B, C, X);
    "VALUE" LR, UR, LC, UC, B, X; "INTEGER" LR, UR, LC, UC;
    "REAL" X; "ARRAY" A, B, C;
    "CODE" 31509;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    LR, UR: <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER BOUND OF THE ROW-INDEX; LR >= 1;
    LC, UC: <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER BOUND OF THE COLUMN-INDEX; LC >= 1;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[L:U], WHERE:
            L = MIN(LR * (LR - 1) // 2 + LC, LC * (LC - 1) // 2 + LR),
            U = MAX(UR * (UR - 1) // 2 + UC, UC * (UC - 1) // 2 + UR)
            AND THE (I,J)-TH ELEMENT OF THE SYMMETRIC MATRIX SHOULD BE
            GIVEN IN A[J * (J - 1) // 2 + I];
    B:      <ARRAY IDENTIFIER>;
            "ARRAY" B[LC:UC]; THE VECTOR;
    X:      <ARITHMETIC EXPRESSION>;
            THE VALUE OF THE MULTIPLYING SCALAR;
    C:      <ARRAY IDENTIFIER>;
            "ARRAY" C[LR:UR];
            THE RESULT A * B  + X * C, CALCULATED WITH DOUBLE LENGTH
            ARITHMETIC, IS OVERWRITTEN ON C.
```

PROCEDURES USED:

```
    DPMUL          = CP31103;
    LNGSYMMATVEC = CP34418.
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

ELEMENTARY.

METHOD AND PERFORMANCE:

ALL PROCEDURES GIVEN IN THIS SECTION MAKE USE OF THE DOUBLE-LENGTH ARITHMETIC OPERATIONS AVAILABLE IN HARDWARE ON THE CYBER 73. LET (X, XX) DENOTE A DOUBLE-LENGTH NUMBER, THEN WE WILL SAY THAT XX IS A TAIL TO X, WHEN THE FOLLOWING CONDITIONS HOLD:

$$X = FL1(X + XX),$$
$$(X, XX) = FL2(X + XX),$$

WHERE FL1(. + .) AND FL2(. + .) DENOTE SINGLE-LENGTH RESPECTIVELY DOUBLE-LENGTH ADDITION WITH TRUNCATING OF THE RESULT TO 48 AND 96 BITS RESPECTIVELY.
WHEN A PROCEDURE DELIVERS A DOUBLE LENGTH RESULT IN D AND DD, THEN THESE RESULTS ARE SUCH THAT DD IS A TAIL TO D; WHEN ONE SHOULD PROVIDE AN INITIALIZING DOUBLE LENGTH SCALAR IN C AND CC, THEN CC SHOULD BE A TAIL TO C, OTHERWISE THE FOLLOWING ERROR MESSAGE WILL BE PRINTED:

DP PARAMETER TAIL ERROR

AND EXECUTION OF THE PROGRAM WILL TERMINATE IN THE USUAL WAY.
NOTE THAT CC = 0 IS A TAIL TO C FOR ALL VALUES OF C. FURTHERMORE, IT SEEMS WORTHWHILE TO NOTE THAT THE ARRAY B MUST BE A VALUE PARAMETER IN ALGOL 60, HOWEVER, IN THE COMPASS ROUTINE THE DUPLICATION OF THIS ARRAY IS ONLY DONE IF NECESSARY, I.E. IF THE ACTUAL PARAMETERS B AND C ARE THE SAME.

SOURCE TEXTS:

THE PROCEDURES IN THIS SECTION ARE WRITTEN IN COMPASS, EXCEPT FOR LNGSYMMATVEC, LNGFULSYMMATVEC AND LNGSYMRESVEC.
WE GIVE EQUIVALENT ALGOL 60 TEXTS OF THE COMPASS ROUTINES. FOR THE COMPASS TEXT SEE APPENDIX C, SECTION 1.5.2.

```
"CODE" 34410;
   "PROCEDURE" LNGVECVEC(L, U, SHIFT, A, B, C, CC, D, DD);
   "VALUE" L, U, SHIFT, C, CC; "INTEGER" L,U,SHIFT;
"REAL" C, CC, D, DD; "ARRAY" A, B;
"BEGIN" "REAL" E, EE;
   "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
   "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
   "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
   "BEGIN" DPMUL(A[L], B[L + SHIFT], E, EE);
      LNGADD(C, CC, E, EE, C, CC)
   "END";
   D:= C; DD:= CC
"END" LNGVECVEC;
      "EOP"
```

```
"CODE" 34411;
   "PROCEDURE" LNGMATVEC(L, U, I, A, B, C, CC, D, DD);
   "VALUE" L, U, I, C, CC; "INTEGER" L, U, I;
   "REAL" C, CC, D, DD; "ARRAY" A, B;
   "BEGIN" "REAL" E, EE;
      "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
      "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
      "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
      "BEGIN" DPMUL(A[I,L], B[I], E, EE); LNGADD(C, CC, E, EE, C, CC)
      "END";
      D:= C; DD:= CC
   "END" LNGMATVEC;
         "EOP"


"CODE" 34412;
   "PROCEDURE" LNGTAMVEC(L, U, I, A, B, C, CC, D, DD);
   "VALUE" L, U, I, C, CC; "INTEGER" L, U, I;
   "REAL" C, CC, D, DD; "ARRAY" A, B;
   "BEGIN" "REAL" E, EE;
      "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
      "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
      "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
      "BEGIN" DPMUL(A[L,I], B[I], E, EE); LNGADD(C, CC, E, EE, C, CC)
      "END";
      D:= C; DD:= CC
   "END" LNGTAMVEC;
         "EOP"


"CODE" 34413;
   "PROCEDURE" LNGMATMAT(L, U, I, J, A, B, C, CC, D, DD);
   "VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
   "REAL" C, CC, D, DD; "ARRAY" A, B;
"BEGIN" "REAL" E, EE;
   "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
   "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
   "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
   "BEGIN" DPMUL(A[I,L], B[L,J], E, EE); LNGADD(C, CC, E, EE, C, CC)
   "END";
   D:= C; DD:= CC
"END" LNGMATMAT;
      "EOP"
```

```
"CODE" 34414;
    "PROCEDURE" LNGTAMMAT(L, U, I, J, A, B, C, CC, D, DD);
    "VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
    "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "REAL" E, EE;
       "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
       "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
       "FOR" L:= L "STEP" 1 "UNTIL" J "DO"
       "BEGIN" DPMUL(A[L,I], B[L,J], E, EE); LNGADD(C, CC, E, EE, C, CC)
       "END";
       D:= C; DD:= CC
    "END" LNGTAMMAT;
          "EOP"


"CODE" 34415;
    "PROCEDURE" LNGMATTAM(L, U, I, J, A, B, C, CC, D, DD);
    "VALUE" L, U, I, J, C, CC; "INTEGER" L, U, I, J;
    "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "REAL" E, EE;
       "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
       "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
       "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
       "BEGIN" DPMUL(A[I,L], B[J,L], E, EE); LNGADD(C, CC, E, EE, C, CC)
       "END";
       D:= C; DD:= CC
    "END" LNGMATTAM;
          "EOP"


"CODE" 34416;
    "PROCEDURE" LNGSEQVEC(L, U, IL, SHIFT, A, B, C, CC, D, DD);
    "VALUE" L, U, IL, SHIFT, C, CC; "INTEGER" L, U, IL, SHIFT;
    "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "REAL" E, EE;
       "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
       "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
       "FOR" L:= L "STEP" 1 "UNTIL" U "DO"
       "BEGIN" DPMUL(A[IL], B[L + SHIFT], E, EE); IL:= IL + L;
          LNGADD(C, CC, E, EE, C, CC)
       "END";
       D:= C; DD:= CC
"END" LNGSEQVEC;
          "EOP"
```

```
"CODE" 34417;
    "PROCEDURE" LNGSCAPRD1(LA, SA, LB, SB, N, A, B, C, CC, D, DD);
    "VALUE" LA, SA, LB, SB, N, C, CC, D, DD; "INTEGER" LA, SA, LB, SB,N;
    "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "REAL" E, EE; "INTEGER" K;
       "PROCEDURE" DPMUL(A, B, C, CC); "CODE" 31103;
       "PROCEDURE" LNGADD(A, AA, B, BB, C, CC); "CODE" 31105;
       "FOR" K:= 1 "STEP" 1 "UNTIL" N "DO"
       "BEGIN" DPMUL(A[LA], B[LB], E, EE); LA:= LA + SA; LB:= LB + SB;
          LNGADD(C, CC, E, EE, C, CC)
       "END";
       D:= C; DD:= CC
    "END";
          "EOP"

"CODE" 34418;
    "PROCEDURE" LNGSYMMATVEC(L, U, I, A, B, C, CC, D, DD);
    "VALUE" L, U, I, C, CC;
    "INTEGER" L, U, I; "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "INTEGER" K, M;
       "PROCEDURE" LNGVECVEC(L, U, S, A, B, C, T, D, R); "CODE" 34410;
       "PROCEDURE" LNGSEQVEC(L, U, IL, S, A, B, C, T, D, R);
       "CODE" 34416;
       M:= "IF" L > I "THEN" L "ELSE" I; K:= M * (M - 1) // 2;
       LNGVECVEC(L, "IF" I <= U "THEN" I - 1 "ELSE" U,
       K, B, A, C, CC, C, CC);
       LNGSEQVEC(M, U, K + I, 0, A, B, C, CC, D, DD)
    "END" LNGSYMMATVEC;
          "EOP"

"CODE" 31505;
    "PROCEDURE" LNGFULMATVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" D, DD;
       "PROCEDURE" LNGMATVEC(L, U, I, A, B, C, CC, D, DD); "CODE" 34411;
       "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
       "BEGIN" LNGMATVEC(LC, UC, LR, A, B, 0, 0, D, DD); C[LR]:= D + DD
       "END"
    "END" LNGFULMATVEC;
          "EOP"

"CODE" 31506;
    "PROCEDURE" LNGFULTAMVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" D, DD;
       "PROCEDURE" LNGTAMVEC(L, U, I, A, B, C, CC, D, DD); "CODE" 34412;
       "FOR" LC:= LC "STEP" 1 "UNTIL" UC "DO"
       "BEGIN" LNGTAMVEC(LR, UR, LC, A, B, 0, 0, D, DD); C[LC]:= D + DD
       "END"
    "END" LNGFULTAMVEC;
          "EOP"
```

```
"CODE" 31507;
    "PROCEDURE" LNGFULSYMMATVEC(LR, UR, LC, UC, A, B, C);
    "VALUE" LR, UR, LC, UC, B; "INTEGER" LR, UR, LC, UC;
    "ARRAY" A, B, C;
    "BEGIN" "REAL" D, DD;
       "PROCEDURE" LNGSYMMATVEC(L, U, I, A, B, C, CC, D, DD);
       "CODE" 34418;
       "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
       "BEGIN" LNGSYMMATVEC(LC, UC, LR, A, B, 0, 0, D, DD);
          C[LR]:= D + DD
       "END"
    "END" LNGFULSYMMATVEC;
          "EOP"


"CODE" 31508;
    "PROCEDURE" LNGRESVEC(LR, UR, LC, UC, A, B, C, X);
    "VALUE" LR, UR, LC, UC, X; "INTEGER" LR, UR, LC, UC;
    "REAL" X; "ARRAY" A, B, C;
    "BEGIN" "REAL" D, DD, E, EE;
       "PROCEDURE" DPMUL(X, Y, E, EE); "CODE" 31103;
       "PROCEDURE" LNGMATVEC(L, U, I, A, B, C, CC, D, DD); "CODE" 34411;
       "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
       "BEGIN" DPMUL(C[LR], X, E, EE);
          LNGMATVEC(LC, UC, LR, A, B, E, EE, D, DD); C[LR]:= D + DD
       "END"
    "END" LNGRESVEC;
          "EOP"


"CODE" 31509;
    "PROCEDURE" LNGSYMRESVEC(LR, UR, LC, UC, A, B, C, X);
    "VALUE" LR, UR, LC, UC, B, X; "INTEGER" LR, UR, LC, UC;
    "REAL" X; "ARRAY" A, B, C;
    "BEGIN" "REAL" D, DD, E, EE;
       "PROCEDURE" DPMUL(X, Y, E, EE); "CODE" 31103;
       "PROCEDURE" LNGSYMMATVEC(L, U, I, A, B, C, CC, D, DD);
       "CODE" 34418;
       "FOR" LR:= LR "STEP" 1 "UNTIL" UR "DO"
       "BEGIN" DPMUL(C[LR], X, E, EE);
          LNGSYMMATVEC(LC, UC, LR, A, B, E, EE, D, DD); C[LR]:= D + DD
       "END"
"END" LNGSYMRESVEC;
          "EOP"
```

```
                                  INPUT: A4 AND A5 CONTAIN ADDRESSES O
                                  HEAD AND TAIL OF OUTPUT PARS
                                  ST(B4+5) - ST(B4+9) CONTAIN RESP.
                                  1ST EL. ADDR AND STRIDE OF 1ST ARRAY
                                  1ST EL. ADDR AND STRIDE OF 2ND ARRAY
                                  NR.-1 OF ELEMENTS TO BE MULTIPLIED,
                                  ALL IN INTEGER FORMAT
        ENTRY   PC3440H
PC3440H BSS     0
COMPUT1 EQ      *+1S17
        SA1     B4+9              N
        SB5     X1
        BX6     X4                "INITIALISATION
        BX7     X5
        LT      B5,COMPUT1
        SA1     B4+5              ADDR 1ST ELEM 1ST ARRAY
        SA2     B4+7              ADDR 1ST ELEM 2ND ARRAY
        SA1     X1
        SA2     X2
        SA3     B4+6              STRIDE 1ST ARRAY
        SB6     X3
        SA3     B4+8              STRIDE 2ND ARRAY
        SB7     X3
AGAIN   DX3     X1*X2             X1*X2 -> (X1,X3)
        FX1     X1*X2
                                  (X1,X3) + (X6,X7)
        DX2     X1+X6
        FX1     X1+X6
        FX3     X3+X7
        NX1
        FX2     X3+X2
        FX3     X2+X1
        DX2     X2+X1
        NX3
        NX2
        FX6     X3+X2
        DX7     X3+X2
        NX7

        SA1     A1+B6             NEXT ELEMENTS
        SA2     A2+B7
        SB6     B6+1
        SB5     B5-1
        GE      B5,AGAIN
        SA6     A4
        SA7     A5
        EQ      COMPUT1

                                  OUTPUT: (D,DD) CONTAIN D.P. RESULT
END
IDENT   ERROR
                                  THIS ROUTINE GIVES THE APPROPRIATE
                                  ERROR MESSAGES IN CASE OF
                                  WRONG KIND, TYPE, DIMENSION, NUMBER
                                  OF PARS, (C,CC) NOT D.P. PAIR, OR
```

```
                                                EXCEEDING (AFTER RND) THE INTEGER
                                                CAPACITY
            ENTRY   PC3100A,PC3100B,PC3100C,PC3100D,PC3100E,PC3100F
OUTPAR      VFD     12/7747B,18/67B,30/0        ERROR
MODOUT      BSSZ    1                           MESSAGE
FORSTD      VFD     12/5767B,18/-FORMST,12/4000,18/4050
VAL61       CON     61.                         IN CASE
DESC61      VFD     12/2061B,30/0,18/VAL61
RETINF      VFD     12/2,18/-TYPEERR,12/4003B,18/0
FORMST      VFD     12/2000B,48/8H"("/"("*      CC NOT
            VFD     12/2000B,48/8H* CC CAN
            VFD     12/2000B,48/8HNOT BE T      TAIL TO C
            VFD     12/2000B,48/8HAIL TO C
            VFD     12/2000B,48/8H")",/")"
PC3100F     BSS     0
CCCERR      SX2     B1                          SUPERGLOBAL PTR,
            SA1     OUTPAR                      OUTPUT PROC DESCR
            IX6     X1+X2
            SA6     MODOUT
            SX1     0
            SB7     3                           ASK FOR 3 WRDS
            RJ      SATISFY                     IN THE STACK
            SA2     FORSTD                      "CREATE
            BX7     X2
            SA7     X1                          PARAMETER
            SA2     DESC61
            BX7     X2                          STACK
            SA7     X1+1
            SA2     RETINF
            SX3     B4
            IX7     X2+X3
            SA7     X1+2                        "
            SA1     MODOUT
            RJ      GOTOPRO
PC3100B     BSS     0
TYPEERR     SA0     63                          GEN. TYPE ERROR MESSAGE
            EQ      ERR
PC3100A     BSS     0
KINDERR     SA0     60
            EQ      ERR
PC3100E     BSS     0
CNTERR      SA0     57
            EQ      ERR
PC3100C     BSS     0
DIMERR      SA0     66
            EQ      ERR
PC3100D     BSS     0
OVFLERR     SA0     132                         INTEGER CAPACITY EXCEEDED
ERR         RJ      JUMPSEG
-           PS      2
            USE     /DATA/
ENTBLOC     EQU     *
XITBLOC     EQU     *+2
XITPROC     EQU     *+4
```

```
GOTOPRO    EQU     *+10B
JUMPSEG    EQU     *+14B
SATISFY    EQU     *+20B
SPEC       EQU     *+30B
EXECEXP    EQU     *+32B
ARRAYER    EQU     *+36B
LASTUSE    EQU     *+50B
STANLST    EQU     *+133B
           END
           IDENT   GETIV
                                          INPUT: X5 HOLDS PAR DESCRIPTION
           ENTER   PC3000A,GETIV
           ERRNAMS   (KIND,TYPE,OVFL)
           UX1     B7,X5
           SA5     X1
           SX7     B7
           MX0     57
           BX0     -X0*X7                 TYPE => X0
           AX7     3                      KIND => X7
           LT      B7,EXPR                EXPR IF 'SIGN OF EXPR' -=T59
           SX7     X7-6                   ALLOWED KIND: 6,10B
           ZR      X7,OKSIMP
           SX7     X7-2
           NZ      X7,KINDERR
OKSIMP     SX0     X0-1                   ALLOWED TYPE: 1,2
           ZR      X0,TRAFO
           SX0     X0-1
           NZ      X0,TYPEERR
TRAFO      BX7     X5
           AX5     59                     SIGN RESULT
           BX6     -X6*X6                 '0'
           PX6
           BX7     X7-X5
           RX7     X7+X6
           BX7     X7-X5                  SIGN(RES)*(ABS(RES)+2000 00,,,0)
           UX7     B7                     REAL => INT
           GT      B7,OVFLERR
           LX7     B7
           JP      B4+4                   RETURN INDIRECT (VIA STACK)
EXPR       BSS     0
           ECHO    2,NR=(166B,2,3)
           SX7     X7+NR
           ZR      X7,KINDOK
           EQ      KINDERR
KINDOK     SX0     X0-1
           ZR      X0,EXEC
           SX0     X0-1
           NZ      X0,TYPEERR
EXEC       RJ      EXECEXP
           EQ      TRAFO
           ALGNAMS
           END
           IDENT   GETRV
                                          INPUT: X5 HOLDS PAR DESCRIPTION
```

```
            ENTER   PC3000B,GETRV
            ERRNAMS   (KIND,TYPE)
            UX1     B7,X5
            SA5     X1
            SX7     B7
            MX0     57
            BX0     -X0*X7                  TYPE => X0
            AX7     3                       KIND => X7
            LT      B7,EXPR                 EXPR IF !SIGN OF EXPR! "=T59
            SX7     X7-6                    ALLOWED KIND: 6,10B
            ZR      X7,OKSIMP
            SX7     X7-2
            NZ      X7,KINDERR
OKSIMP      SX0     X0-1                    ALLOWED TYPE: 1,2
            ZR      X0,RET
            SX0     X0-1
            NZ      X0,TYPEERR
RET         BX7     X5
            JP      B4+4                    RETURN INDIRECT (VIA STACK)
EXPR        BSS     0
            ECHO    2,NR=(166B,2,3)
            SX7     X7+NR
            ZR      X7,KINDOK
            EQ      KINDERR
KINDOK      SX0     X0-1
            ZR      X0,EXEC
            SX0     X0-1
            NZ      X0,TYPEERR
EXEC        RJ      EXECEXP
            EQ      RET
            ALGNAMS
            END
            IDENT   GETRR
```

INPUT: X5 HOLDS PAR DESCRIPTION

```
          ENTER   PC3000C,GETRR
          ERRNAMS   (KIND,TYPE)
          UX1     B7,X5
          SA5     X1
          SB6     102B                    SIMPLE REAL VAR
          EQ      B6,B7,OK
          SB6     -1665B                  SUBSCR REAL
          NE      B6,B7,ERROR
          RJ      EXECEXP
OK        SX7     A5
          JP      B4+4
ERROR     SX7     B7
          AX7     3
          MX6     56
          BX5     -X6*X7                  KIND => X5
          SB7     X5-10B                  ALLOWED: 10B,11B
          ZR      B7,TYPEERR
          SB7     B7-1
          ZR      B7,TYPEERR
          EQ      KINDERR
          ALGNAMS
                                          OUTPUT: X7 HOLDS ADDR OF PAR
          END
```

```
"CODE" 34418;
    "PROCEDURE" LNGSYMMATVEC(L, U, I, A, B, C, CC, D, DD);
    "VALUE" L, U, I, C, CC;
    "INTEGER" L, U, I; "REAL" C, CC, D, DD; "ARRAY" A, B;
    "BEGIN" "INTEGER" K, M;
        "PROCEDURE" LNGVECVEC(L, U, S, A, B, C, T, D, R); "CODE" 34410;
        "PROCEDURE" LNGSEQVEC(L, U, S, A, B, C, T, D, R); "CODE" 34416;
        M:= "IF" L > I "THEN" L "ELSE" I; K:= M * (M - 1) // 2;
        LNGVECVEC(L, "IF" I <= U "THEN" I - 1 "ELSE" U,
        K, B, A, C, CC, C, CC);
        LNGSEQVEC(M, U, K + I, 0, A, B, C, CC, D, DD)
    "END" LNGSYMMATVEC;
        "EOP"
```

2-nd REVISION, 1977

MC

SECTION : 1.5.3 (MARCH 1977) PAGE 1

AUTHORS: T.J.DEKKER & J.KOOPMAN.

CONTRIBUTOR: J.KOOPMAN.

INSTITUTE: UNIVERSITY OF AMSTERDAM.

RECIEVED: 770328

BRIEF DESCRIPTION:

LNGREATODECI CONVERTS A DOUBLE-LENGTH NUMBER TO A
NUMBER IN DECIMAL FLOATING-POINT REPRESENTATION.
THE RESULT CONSISTS OF A MANTISSA MANT OF MAGNITUDE<1
(AND >=.1) AND A DECIMAL EXPONENT EXPO.

KEYWORDS:
DOUBLE PRECISION ARITHMETIC,
CONVERSION,
DECIMAL REPRESENTATION.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" LNGREATODECI(X, XX, S, MANT, EXPO);
"VALUE"X,XX,S;"INTEGER"S,EXPO;"REAL"X,XX;"INTEGER""ARRAY"MANT;
"CODE" 31100;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

X,XX : <ARITHMETIC EXPRESSIONS>;
ENTRY: THE HEAD (X) AND TAIL (XX) OF THE NUMBER
THAT IS TO BE CONVERTED;

S : <ARITHMETIC EXPRESSION>;
ENTRY: THE DESIRED NUMBER OF SIGNIFICANT DIGITS
OF THE CONVERTED VARIABLE.
ONE SHOULD NOT CHOOSE S LARGER THAN THE NUMBER
OF DIGITS CORRESPONDING TO THE DOUBLE LENGTH
MACHINE PRECISION (FOR CDC: S<29). OTHERWISE,
THE LAST DIGITS ARE USELESS, AS ALL OPERATIONS
IN LNGREATODECI ARE PERFORMED IN DOUBLE-LENGTH A-
RITHMETIC; IF S IS CHOSEN NONPOSITIVE,ONLY THE SIGN
AND THE DECIMAL EXPONENT OF THE CONVERTED NUMBER
ARE DELIVERED;

```
MANT    : <ARRAY IDENTIFIER>;
          "INTEGER""ARRAY"MANT[0:S];
          EXIT: MANT[0]: THE SIGN OF THE DECIMAL NUMBER;
                MANT[I] (I"=0): THE I-TH  SIGNIFICANT
                DIGIT OF THE MANTISSA OF THE CONVERTED NUMBER;
                I.E. THE VALUE OF THE MANTISSA EQUALS
                MANT[0]*(MANT[1]/10+MANT[2]/100+...MANT[S]/10**S);
EXPO    : <INTEGER VARIABLE>;
          EXIT: THE DECIMAL EXPONENT OF THE CONVERTED NUMBER,
                I.E. THE DOUBLE-LENGTH NUMBER (X,XX) APPROXIMATELY
                EQUALS MANTISSA*10**EXPO WITH THE VALUE OF
                MANTISSA GIVEN IN MANT.
```

PROCEDURES USED:

```
LNG SUB = CP31106.
LNG MUL = CP31108.
DP POW  = CP31109.
```

RUNNING TIME:

ROUGHLY PROPORTIONAL TO LN(LN(X))+S.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

LNGREATODECI DETERMINES THE DECIMAL EXPONENT EXPO. AFTER THAT, THE LONG REAL NUMBER (X,XX) IS DIVIDED BY 10**EXPO IN DOUBLE PRECISION. BY TRUNCATING THE RESULT, THE FIRST MOST SIGNIFICANT DIGIT OF THE MANTISSA IS OBTAINED. SUBTRACTING THIS DIGIT FROM (X,XX)/10**EXPO, MULTIPLYING THE RESULT WITH 10, THE NEXT MOST SIGNIFICANT DIGIT CAN BE OBTAINED BY TRUNCATION. THIS PROCESS OF SUBTRACTION, MULTIPLICATION AND TRUNCATION WILL BE REPEATED UNTIL S DIGITS ARE OBTAINED. FINALLY, THE MANTISSA THUS OBTAINED IS PROPERLY ROUNDED.

EXAMPLE OF USE:

```
"BEGIN""COMMENT"EXAMPLE OF USE OF LNGREATODECI AND DP POW;
      "INTEGER"S,EXPO;
      "REAL"OP,OPL;
      "PROCEDURE"DP POW(A,EXPON,C,CC);"CODE"31109;
      "PROCEDURE"LNGREATODECI(X,XX,S,MANT,EXPO);"CODE"31110;

      "PROCEDURE"PRINT(S,MANT,EXPONENT);
      "VALUE"S,EXPONENT;"INTEGER"S,EXPONENT;"INTEGER""ARRAY"MANT;
      "BEGIN""INTEGER"K;
            OUTCHARACTER(61,"("-+")",MANT[0]+2);
            "FOR"K:=1"STEP"1"UNTIL"S"DO"
            "BEGIN""IF"K=1"THEN"OUTPUT(61,"(""(".")""")");
                  OUTPUT(61,"("D")",MANT[K])
            "END";OUTPUT(61,"(""("")",+3D")",EXPONENT)
      "END"PRINT;
      DP POW(2,48,OP,OPL);
      "FOR"S:=0"STEP"4 UNTIL"28"DO"
      "BEGIN""INTEGER""ARRAY"MANT[0:S];
            LNGREATODECI(OP,OPL,S,MANT,EXPO);
            PRINT(S,MANT,EXPO);
            OUTPUT(61,"("/")")
      "END"
"END"
```

DELIVERS:

```
+"+015
+.2815"+015
+.28147498"+015
+.281474976711"+015
+.2814749767106560"+015
+.28147497671065600000"+015
+.281474976710656000000000"+015
+.2814749767106560000000000000"+015
```

SOURCE TEXT(S):

```
    "CODE"31100;
"PROCEDURE"LNGREATODECI(X,XX,S,MANT,EXPO);
"VALUE"X,XX,S;"INTEGER"S,EXPO;"REAL"X,XX;"INTEGER""ARRAY"MANT;
"BEGIN""INTEGER"I,K;
        "REAL"P,PP;
        "PROCEDURE"LNG SUB(A,AA,B,BB,C,CC);"CODE"31106;
        "PROCEDURE"LNG MUL(A,AA,B,BB,C,CC);"CODE"31107;
        "PROCEDURE"DP POW(A,EXPON,C,CC);"CODE"31109;
        MANT[0]:=SIGN(X);"IF"X<0"THEN""BEGIN"X:=-X;XX:=-XX"END";
        "IF"X=0"THEN"EXPO:=0
                "ELSE"EXPO:=ENTIER(LN(X)/LN(10))+1;
        DP POW(10,-EXPO,P,PP);
        LNG MUL(X,XX,P,PP,X,XX);
        "FOR"I:=0"WHILE"ENTIER(X)=0 & X"=0 "DO"
        "BEGIN"LNG MUL(X,XX,10,0,X,XX);EXPO:=EXPO-1"END";
        "FOR"I:=1"STEP"1"UNTIL"S"DO"
        "BEGIN"K:=ENTIER(X);"IF"K>9"THEN"K:=9;MANT[I]:=K;
              LNG SUB(X,XX,K,0,P,PP);LNG MUL(P,PP,10,0,X,XX)
        "END";
        "IF"ENTIER(X)>=5
        "THEN""BEGIN""FOR"I:=S"STEP"-1"UNTIL"1"DO"
              "BEGIN"K:=MANT[I]+1;
                    "IF"K<10"THEN""BEGIN"MANT[I]:=K;"GOTO"READY
                                        "END";
                    MANT[I]:=0
              "END";
              EXPO:=EXPO+1;
              "IF"S>0"THEN"MANT[1]:=1;
               READY:
            "END";
        EXPO:=EXPO+1
"END" LNGREATODECI;
    "EOP"
```