MC SYLLABUS 47.2

NUMAL

NUMERICAL PROCEDURES IN ALGOL 60

VOLUME 1, ELEMENTARY PROCEDURES

VOLUME 2, ALGEBRAIC EVALUATIONS

P.W. HEMKER (ed.)

MATHEMATISCH CENTRUM AMSTERDAM 1981

1980 Mathematics subject classification: 65XX04, 68B99

ISBN 90 6196 217 X

INDEX	PROCEDURE		CODE	MNT/YR	RECORD NUMBER
1. 4.LONG INTEGER ARITHMETIC					
	LNGINTADD		31200	OCT/74	201
	LNGINTSUBTRACT		31201	OCT/74	201
	LNGINTMULT		31202	OCT/74	201
	LNGINTDIVIDE		31203	OCT/74	201
	LNGINTPOWER		31204	OCT/74	201
5.LONG REAL ARITHMETIC					
1.ELEM. ARITHMETIC OPERATIONS					
	DP ADD	*	31101	MRC/77	271
	DP SUB	*	31102	MRC/77	271
	DP MUL	*	31103	MRC/77	271
	DP DIV	*	31104	MRC/77	271
	DP POW		31109	MRC/77	271
	LNG ADD	*	31105	MRC/77	271
	LNG SUB	*	31106	MRC/77	271
	LNG MUL	*	31107	MRC/77	271
	LNG DIV	*	31108	MRC/77	271
	LNG POW		31110	MRC/77	271
2.SCALAR PRODUCTS					
	LNGVECVEC	*	34410	JAN/76	285
	LNGMATVEC	*	34411	JAN/76	285
	LNGTAMVEC	*	34412	JAN/76	285
	LNGMATMAT	*	34413	JAN/76	285
	LNGTAMMAT	*	34414	JAN/76	285
	LNGMATTAM	*	34415	JAN/76	285
	LNGSEQVEC	*	34416	JAN/76	285
	LNGSCAPRD1	*	34417	JAN/76	285
	LNGSYMMATVEC		34418	JAN/76	285
	LNGFULMATVEC	*	31505	JAN/76	285
	LNGFULTAMVEC	*	31506	JAN/76	285
	LNGFULSYMMATVEC		31507	JAN/76	285
	LNGRESVEC	*	31508	JAN/76	285
	LNGSYMRESVEC		31509	JAN/76	285
3.CONVERSION					
	LNGREATODECI		31100	MRC/77	289
2.ALGEBRAIC EVALUATIONS					
1.EVAL. OF A FINITE SERIES					
2.EVAL. OF POLYNOMIALS					
1.EVAL. OF GENERAL POLYNOMIALS					
1.POLYNOMIALS IN GRUNERT FORM					
	POL		31040	OCT/75	245
	TAYPOL		31241	OCT/75	245
	NORDERPOL		31242	OCT/75	245
	DERPOL		31243	OCT/75	245
2.POLYNOMIALS IN NEWTON FORM					
2.EVAL. OF ORTHOGON. POLYNOMIALS					
1.GENERAL ORTHOGON. POLYNOMIALS					
	ORTPOL		31044	NOV/78	293
	ORTPOLSYM		31048	NOV/78	293
	ALLORTPOL		31045	NOV/78	293
	ALLORTPOLSYM		31049	NOV/78	293
	SUMORTPOL		31047	NOV/78	293
	SUMORTPOLSYM		31058	NOV/78	293
2.CHEBYCHEV POLYNOMIALS					
2. 2. 2. 2.	CHEPOLSUM		31046	DEC/78	229

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;
"CODE" 31010;
```

```
FORMAL PARAMETERS:
L,U:    <ARITHMETIC EXPRESSION>;
        LOWER AND UPPER INDEX OF THE VECTOR A, RESPECTIVELY;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[L : U], THE ARRAY TO BE INITIALIZED;
X:      <ARITHMETIC EXPRESSION>;
        INITIALIZATION CONSTANT.
```

LANGUAGE: COMPASS.

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;
"CODE" 31011;

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>;
"ARRAY" A[LR : UR, LC : UC], THE ARRAY TO BE INITIALIZED;
X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: COMPASS.

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;
"CODE" 31012;

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>;
"ARRAY" A[LR : UR, LR + SHIFT : UR + SHIFT], THE ARRAY TO BE INITIALIZED;
X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: COMPASS.

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) * J//2 + 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 INDEX OF 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) * J//2 + I]; J = 1, ... ,N; I = 1, ... ,J.
FOR FIXED ORDER N, THE PARAMETERS L, U AND I SHOULD SATISFY THE CONDITIONS : 1 <=L<= N, 1 <=U<= N, 1 <=I<= N ;

X: <ARITHMETIC EXPRESSION>;
INITIALIZATION CONSTANT.

LANGUAGE: ALGOL 60.

SOURCE TEXT(S):

THE PROCEDURES INIVEC, INIMAT AND INIMATD ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"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 <= I "THEN"
        "BEGIN" K:= (I - 1) * I//2; L := K + L;
            K := ("IF" U < I "THEN" U "ELSE" I) + K;
            "FOR" L:= L "STEP" 1 "UNTIL" K "DO" A[L]:= X;
            L := I + 1
        "END";
        "IF" U>I "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;
"CODE" 31030;

FORMAL PARAMETERS:
L,U: <ARITHMETIC EXPRESSION>;
LOWER AND UPPER VECTOR-INDEX, RESPECTIVELY;
SHIFT: <ARITHMETIC EXPRESSION>;
INDEX-SHIFTING PARAMETER;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[L : U], B[L + SHIFT : U + SHIFT], B IS COPIED INTO A.

LANGUAGE: COMPASS.

SUBSECTION: DUPVECROW.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" DUPVECROW(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "CODE" 31031;

    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[L : U], B[I : I, L : U], B IS COPIED INTO A.
```

LANGUAGE: COMPASS.

SUBSECTION: DUPROWVEC.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" DUPROWVEC(L, U, I, A, B); "VALUE" L,U,I;
    "INTEGER" L,U,I; "ARRAY" A,B;
    "CODE" 31032;

    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[I : I, L : U], B[L : U], B IS COPIED INTO A.
```

LANGUAGE: COMPASS.

SUBSECTION: DUPVECCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" DUPVECCOL(L, U, J, A, B); "VALUE" L,U,J;
"INTEGER" L,U,J; "ARRAY" A,B;
"CODE" 31033;

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[L : U], B[L : U, I : I], B IS COPIED INTO A.
```

LANGUAGE: COMPASS.

SUBSECTION: DUPCOLVEC.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" DUPCOLVEC(L, U, J, A, B); "VALUE" L,U,J;
"INTEGER" L,U,J; "ARRAY" A,B;
"CODE" 31034;

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[L : U, I : I], B[L : U], B IS COPIED INTO A.
```

LANGUAGE: COMPASS.

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;
    "CODE" 31035;

    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[L : U, I : J], B[L : U, I : J],  B IS COPIED INTO
             A.
```

LANGUAGE: COMPASS.

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"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,
MULTIPLICATIONS.

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;
"CODE" 31020;

FORMAL PARAMETERS:
L,U: <ARITMETIC EXPRESSION>;
LOWER AND UPPER VECTOR-INDEX, RESPECTIVELY;
SHIFT: <ARITHMETIC EXPRESSION>;
SUBSCRIPT-SHIFTING PARAMETER;
A,B: <ARRAY IDENTIFIER>;
"ARRAY" A[L : U], B[L + SHIFT : U + SHIFT], THE PRODUCT OF
THE CONTENTS OF B ARE STORED IN A.
X: <ARITHMETIC EXPRESSION>;
MULTIPLICATION FACTOR.

LANGUAGE: COMPASS.

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;
    "CODE" 31021;

    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[I : I, L : U], B[J : J, L : U], THE CONTENTS OF B
             MULTIPLIED BY X ARE STORED INTO A.
    X:       <ARITHMETIC EXPRESSION>;
             MULTIPLICATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 31022;

    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[L : U, I : I], B[L : U, J : J], THE CONTENTS OF B
             MULTIPLIED BY X ARE STORED INTO A;
    X:       <ARITHMETIC EXPRESSION>;
             MULTIPLICATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 31131;

    FORMAL PARAMETERS:
    L,U:    <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER ROW-INDEX, RESPECTIVELY;
    J:      <ARITHMETIC EXPRESSION>;
            COLUMN-INDEX;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[L : U, J : J];
    X:      <ARITHMETIC EXPRESSION>;
            MULTIPLICATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 31132;

    FORMAL PARAMETERS:
    L,U:    <ARITHMETIC EXPRESSION>;
            LOWER AND UPPER COLUMN-INDEX, RESPECTIVELY;
    I:      <ARITHMETIC EXPRESSION>;
            ROW-INDEX;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY" A[I : I, L : U];
    X:      <ARITHMETIC EXPRESSION>;
            MULTIPLICATION FACTOR.
```

LANGUAGE: COMPASS.

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"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, J.WOLLESWINKEL.

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

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 SCALARPRODUCT 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:

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

    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[L : U], B[L + SHIFT : U + SHIFT].
```

LANGUAGE: COMPASS.

SUBSECTION: MATVEC.

CALLING SEQUENCE:

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

    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[I : I, L : U], B[L : U].
```

LANGUAGE: COMPASS.

SUBSECTION: TAMVEC.

CALLING SEQUENCE:

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

    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[L : U, I : I], B[L : U].
```

LANGUAGE: COMPASS.

SUBSECTION: MATMAT.

CALLING SEQUENCE:

```
    HEADING:
    "REAL" "PROCEDURE" MATMAT(L, U, I, J, A, B); "VALUE" L,U,I,J;
    "CODE" 34013;

    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[I : I, L : U], B[L : U, J : J].
```

LANGUAGE: COMPASS.

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;
    "CODE" 34014;

    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[L : U, I : I], B[L : U, J : J].
```

LANGUAGE: COMPASS.

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;
    "CODE" 34015;

    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[I : I, L : U], B[J : J, L : U].
```

LANGUAGE: COMPASS.

SUBSECTION: SEQVEC.

CALLING SEQUENCE:

```
    HEADING:
    "REAL" "PROCEDURE" SEQVEC(L, U, IL, SHIFT, A, B);
    "CODE" 34016;

    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[P : Q], B[L + SHIFT, U + SHIFT];
             THE VALUES OF P AND  Q  SHOULD  SATISFY  P <= IL  AND  Q >=
             IL+(U+L-1)*(U-L)//2).
```

LANGUAGE: COMPASS.

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;
    "CODE" 34017;

    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[P : Q], B[R : S];
             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.

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 ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 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"

"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"
```

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 A 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 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 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 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 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, UC;
   "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"
```

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: ALGOL 60

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,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[LR:UR,I:I];
THE HOUSEHOLDER VECTOR;
A: <ARRAY IDENTIFIER>; "ARRAY" A[LR:UR,LC:UC];
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,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[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:
    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;
"CODE" 34020;

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[L : U], B[L + SHIFT : U + SHIFT];

X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS.

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;
"CODE" 34023;

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[L : U, I : I], B[L : U, J : J];

X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS.

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;
    "CODE" 34024;

    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[I : I, L : U], B[J : J, L : U];
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 34021;

    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[L : U], B[L : U, I : I];
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 34022;

    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[L : U, I : I], B[L : U];
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
    "CODE" 34026;

    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[L : U], B[I : I, L : U];
    X:       <ARITHMETIC EXPRESSION>;
             ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
"CODE" 34027;
```

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[I : I, L : U], B[L : U];
X:      <ARITHMETIC EXPRESSION>;
        ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
"CODE" 34029;
```

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[L : U, I : I], B[J : J, L : U],  WHEN  A = B THEN
        CORRECT  ELIMINATION  IS GUARANTEED  ONLY WHEN  THE ROW AND
        COLUMN ARE DISJUNCT;
X:      <ARITHMETIC EXPRESSION>;
        ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

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;
"CODE" 34028;
```

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[I : I, L : U], B[L : U, J : J], WHEN A = B THEN CORRECT ELIMINATION IS GUARANTEED ONLY WHEN THE ROW AND COLUMN ARE DISJUNCT;

X: <ARITHMETIC EXPRESSION>;
ELIMINATION FACTOR.

LANGUAGE: COMPASS.

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;
    "CODE" 34025;

    MAXELMROW DELIVERS THE INDEX OF THE MAXIMAL ELEMENT AFTER ELIMINA-
            TION STEP IN 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 A;
    J:      <ARITHMETIC EXPRESSION>;
            ROW-INDEX OF THE ROW VECTOR B;
    A,B:    <ARRAY IDENTIFIER>;
            "ARRAY" A[I : I, L : U], B[I : I, L : U];
    X:      <ARITHMETIC EXPRESSION>;
            ELIMINATION FACTOR.
```

LANGUAGE: COMPASS.

REFERENCES:

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

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
    ELMVEC
    ELMROW
    ELMVECCOL
    ELMCOLVEC
    MAXELMROW

"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;
"CODE" 34030;

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[P : Q]; P AND Q SHOULD SATISFY:  P <= L,  Q >= U,
         P <= L + SHIFT AND Q >= U + SHIFT.
```

LANGUAGE: COMPASS.

SUBSECTION: ICHCOL.

CALLING SEQUENCE:

```
HEADING:
"PROCEDURE" ICHCOL(L, U, I, J, A); "VALUE" L,U,I,J;
"INTEGER" L,U,I,J; "ARRAY" A;
"CODE" 34031;

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[L : U, P : Q]; P AND Q SHOULD SATISFY:  P  <=  I,
         P <= J, Q >= I AND Q >= J.
```

LANGUAGE: COMPASS.

SUBSECTION: ICHROW.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ICHROW(L, U, I, J, A); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A;
    "CODE" 34032;

    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[P : Q, L : U]; P AND Q SHOULD SATISFY:  P  <=  I,
            P <= J, Q >= I AND Q >= J.
```

LANGUAGE: COMPASS.

SUBSECTION: ICHROWCOL.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ICHROWCOL(L, U, I, J, A); "VALUE" L,U,I,J;
    "INTEGER" L,U,I,J; "ARRAY" A;
    "CODE" 34033;

    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[P : Q, R : S]; P,  Q,  R  AND  S  SHOULD SATISFY:
            P <= I, P <= L, Q >= I, Q >= U, R <= J, R <= L, S >= J  AND
            S >= U.  FURTHERMORE THE ROW AND COLUMN  TO BE INTERCHANGED
            SHOULD BE DISJUNCT.
```

LANGUAGE: COMPASS.

SUBSECTION: ICHSEQVEC.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ICHSEQVEC(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
    "INTEGER" L,U,IL,SHIFT; "ARRAY" A;
    "CODE" 34034;

    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[P : Q];
             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.

SUBSECTION: ICHSEQ.

CALLING SEQUENCE:

```
    HEADING:
    "PROCEDURE" ICHSEQ(L, U, IL, SHIFT, A); "VALUE" L,U,IL,SHIFT;
    "INTEGER" L,U,IL,SHIFT; "ARRAY" A;
    "CODE" 34035;

    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[P : Q];
            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: COMPASS.

REFERENCES:

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

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"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: 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;
"CODE" 34040;

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[L : U, P : Q]; P AND Q SHOULD SATISFY: P <= I, P <= J, Q >= I AND Q >= J;

C,S: <ARITHMETIC EXPRESSION>;
ROTATION FACTORS.

LANGUAGE: COMPASS.

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;
"CODE" 34041;
```

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[P : Q, L : U]; P AND Q SHOULD SATISFY:  P  <=  I,
        P <= J, Q >= I AND Q >= J;
C,S:    <ARITHMETIC EXPRESSION>;
        ROTATION FACTORS.
```

LANGUAGE: COMPASS.

REFERENCES:

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

SOURCE TEXT(S):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE 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 INDEX 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 SEQUENCE:

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

PROCEDURES USED: 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 INDEX 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 COLUMN INDEX
                  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 ROW INDEX 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, ..., 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;
    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):

THE FOLLOWING PROCEDURES ARE WRITTEN IN COMPASS, AN EQUIVALENT ALGOL 60 TEXT OF THESE COMPASS ROUTINES IS GIVEN.

```
"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 : 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 SEQUENCE :

```
    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 ARGUMENT 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);
"VALUE" 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,U: <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 HSHCOMCOL);

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 USED : 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.
    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")",4B,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"
```

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: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.
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" 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/,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 3.4

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"
```

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 :

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 EIGEN-VECTOR, 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 EIGEN-
               VECTOR 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 EIGEN-
               VECTOR 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 EIGEN-
        VECTORS 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.

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"

"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[O:U[O]], V[O:V[O]],
                              SUM[O:MAX(U[O],V[O])+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[O],V[O]) + .06*MIN(U[O],V[O]) + .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[O:U[O]], V[O:V[O]],
              QUOTIENT[O:U[O]-V[O]+1], REMAINDER[O:V[O]];
              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[O] + (IF V[O]=1 OR U[O]<V[O] THEN 0 ELSE V[O]+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[O]=1 THEN (.34*U[O] + .67) ELSE IF V[1] >= 5 000 000 THEN
                        (.26*DIFF*V[O] + .57*DIFF + .10*V[O] + 1.8)
               ELSE  (.27*DIFF*V[O] + .66*DIFF + .66*V[O] + 2.0)
    (HERE DIFF=U[O]-V[O]+1, I.E. THE NUMBER OF EXECUTIONS
    OF THE STATEMENT, IN WHICH DIVISION OF A (V[O]+1)-PLACE
    NUMBER BY A V[O]-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 48 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 "DP 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 LNGREATODECI (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):

ALL PROCEDURES, EXCEPT POW AND LNG POW, ARE WRITTEN IN COMPASS. IT IS NOT POSSIBLE TO SIMULATE THESE PROCEDURES IN ALGOL 60, SO ONLY THE TEXT IS GIVEN FOR POW AND LNG POW.

```
"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 DOUBLE 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, U, 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" U "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"
```

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 OB- TAINED. SUBTRACTING THIS DIGIT FROM (X,XX)/10**EXPO, MULTIPLYING THE RESULT WITH 10, THE NEXT MOST SIGNI- FICANT DIGIT CAN BE OBTAINED BY TRUNCATION. THIS PRO- CESS 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[O]+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[O: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"
```

*********** M10502V //// END OF LIST ////
*********** M10502V //// END OF LIST ////

AUTHOR: C.G. VAN DER LAAN.

INSTITUTE: REKENCENTRUM RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED: 741114.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE PROCEDURES POL, TAYPOL, NORDERPOL AND DERPOL.

POL EVALUATES A POLYNOMIAL;

DERPOL EVALUATES THE FIRST K DERIVATIVES OF A POLYNOMIAL;

NORDERPOL EVALUATES THE FIRST K NORMALIZED DERIVATIVES OF A POLYNOMIAL (I.E. J-TH DERIVATIVE/(J FACTORIAL),J=0,1,...,K<=DEGREE;

TAYPOL EVALUATES X**J*(J-TH DERIVATIVE)/(J FACTORIAL), J=0,1,...,K<=DEGREE.

KEYWORDS:

POLYNOMIAL EVALUATION,
(NORMALIZED) DERIVATIVES,
DERIVATIVES OF A POLYNOMIAL.

SUBSECTION: POL.

CALLING SEQUENCE:

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

POL:=A[0]+A[1]*X+...+A[N]*X**N;

THE MEANING OF THE FORMAL PARAMETERS IS:
N:   <ARITHMETIC EXPRESSION>;
     ENTRY: THE DEGREE OF THE POLYNOMIAL;
X:   <ARITHMETIC EXPRESSION>;
     ENTRY: THE ARGUMENT OF THE POLYNOMIAL;
A:   <ARRAY IDENTIFIER>;
     "ARRAY"A[0:N];
     ENTRY: THE COEFFICIENTS OF THE POLYNOMIAL
            A[0]+A[1]*X+...+A[N]*X**N.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(N MULTIPLICATIONS AND ADDITIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:
THE METHOD USED FOR EVALUATION IS HORNER'S RULE (SYNTHETIC DIVISION). THE ERROR GROWTH IS GIVEN BY A LINEAR FUNCTION OF THE DEGREE OF THE POLYNOMIAL (SEE VAN DER LAAN, STOER(1972) P. 29 (EX. 11) OR WILKINSON(1963) P. 36,37).

SUBSECTION: DERPOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" DERPOL(N,K,X,A);
"VALUE"N,K,X;"INTEGER"N,K;"REAL"X;"ARRAY"A;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:    <ARITHMETIC EXPRESSION>;
      ENTRY: THE DEGREE OF THE POLYNOMIAL;
K:    <ARITHMETIC EXPRESSION>;
      ENTRY: THE FIRST K DERIVATIVES ARE TO BE CALCULATED;
X:    <ARITHMETIC EXPRESSION>;
      ENTRY: THE ARGUMENT OF THE POLYNOMIAL;
A:    <ARRAY IDENTIFIER>;
      "ARRAY"A[0:N];
      ENTRY: THE COEFFICIENTS OF THE POLYNOMIAL
             A[0]+A[1]*X+...+A[N]*X**N;
      EXIT: THE J-TH DERIVATIVE IS DELIVERED IN A[J], J=0,1,...,K<=
            DEGREE; THE OTHER ELEMENTS ARE GENERALLY ALTERED.
```

PROCEDURES USED :

NORDERPOL = CP31242

RUNNING TIME: THE NUMBER OF ADDITIONS IS (K+1)*(N-K/2) AND
THE NUMBER OF MULTIPLICATIONS IS N, IN FIRST ORDER.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE TAYPOL(THIS SECTION).

SUBSECTION: NORDERPOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" NORDERPOL(N,K,X,A);
"VALUE"N,K,X;"INTEGER"N,K;"REAL"X;"ARRAY"A;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:    <ARITHMETIC EXPRESSION>;
      ENTRY: THE DEGREE OF THE POLYNOMIAL;
K:    <ARITHMETIC EXPRESSION>;
      THE FIRST K NORMALIZED DERIVATIVES ARE TO BE CALCULATED
      (I.E. (J-TH DERIVATIVE)/(J FACTORIAL), J=0,1,...,K<=DEGREE).
X:    <ARITHMETIC EXPRESSION>;
      ENTRY: THE ARGUMENT OF THE POLYNOMIAL;
A:    <ARRAY IDENTIFIER>;
      "ARRAY"A[0:N];
      ENTRY: THE COEFFICIENTS OF THE POLYNOMIAL
             A[0]+A[1]*X+...+A[N]*X**N;
      EXIT: THE J-TH NORMALIZED DERIVATIVE IS DELIVERED IN A[J]
            J=0,1,...,K<=DEGREE; THE OTHER ELEMENTS ARE GENERALLY
            ALTERED.
```

PROCEDURES USED: NONE.

REQUIRED CENTRAL MEMORY: AN AUXILIARY ARRAY OF ORDER N + 1 IS DECLARED.

RUNNING TIME: THE NUMBER OF ADDITIONS IS (K+1)*(N-K/2) AND
THE NUMBER OF MULTIPLICATIONS/DIVISIONS IS 2 * N + K.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE: SEE TAYPOL(THIS SECTION).

SUBSECTION: TAYPOL.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"PROCEDURE" TAYPOL(N,K,X,A);
"VALUE"N,K,X;"INTEGER"N,K;"REAL"X;"ARRAY"A;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL;

K: <ARITHMETIC EXPRESSION>;
ENTRY: THE FIRST K TERMS X**J*(J-TH DERIVATIVE)/(J FACTORIAL), J=0,1,...,K<=DEGREE, ARE TO BE CALCULATED;

X: <ARITHMETIC EXPRESSION>;
ENTRY: THE ARGUMENT OF THE POLYNOMIAL;

A: <ARRAY IDENTIFIER>;
"ARRAY"A[0:N];
ENTRY: THE COEFFICIENTS OF THE POLYNOMIAL A[0]+A[1]*X+...+A[N]*X**N;
EXIT: THE J-TH TERM X**J*(J-TH DERIVATIVE)/(J FACTORIAL), IS DELIVERED IN A[J], J=0,1,...,K<=DEGREE; THE OTHER ELEMENTS ARE GENERALLY ALTERED.

PROCEDURES USED: NONE.

RUNNING TIME: THE NUMBER OF ADDITIONS IS (K+1)*(N-K/2) AND THE NUMBER OF MULTIPLICATIONS IS 2 * N.

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE:

THE METHOD OF EVALUATION IS GIVEN BY TRAUB AND SHAW(1972,1974). LET X**J*(J-TH DERIVATIVE OF THE POLYNOMIAL)/(J FACTORIAL)= (J OVER J)*A[J]*X**J+(J+1 OVER J)*A[J+1]*X**(J+1)+...+(N OVER J)* A[N]*X**N,THEN THE J-TH DERIVATIVE (UP TO A FACTOR) CAN BE OBTAINED FROM THE BINOMIAL COEFFICIENTS FOLLOWED BY EVALUATION OF THE ABOVE INPRODUCT. THE SHAW AND TRAUB ALGORITHM PERFORMS THE BUILDING UP OF THE BINOMIAL COEFFICIENTS IMPLICITLY.

WE HAVE NOT IMPLEMENTED THE MORE SOPHISTICATED ALGORITHM, BASED ON DIVISORS OF N+1, BECAUSE OF THE MORE COMPLEX APPEARANCE OF THE IMPLEMENTATION AND BECAUSE OF THE DIFFICULTY IN CHOSING THE MOST EFFICIENT DIVISOR. OUR (RESTRICTED) IMPLEMENTATION OF THE ONE-PARAMETER FAMILY OF ALGORITHMS PRESERVES THE LINEAR NUMBER OF MULTIPLICATIONS (2*N (NORDERPOL, TAYPOL) AND 3*N (DERPOL)).

THE ABSOLUTE ERROR IS OF ORDER MAX((N OVER N)*A[N]*X**(N-K),..., (N OVER K)*A[K]), FOR THE K-TH NORMALIZED DERIVATIVE (SEE VAN DER LAAN OR WOZNIAKOWSKI).

REFERENCES:

[1].SHAW,M. AND J. TRAUB:
ON THE NUMBER OF MULTIPLICATIONS FOR THE EVALUATION OF A POLYNOMIAL AND SOME OF ITS DERIVATIVES (21 P.).
JOURN. ACM, 1974, VOL. 21, NO. 1, P. 161-167.

[2].STOER,J. :
EINFUEHRUNG IN DIE NUMERISCHE MATHEMATIK 1.
SPRINGER, 1972.

[3].VAN DER LAAN C.G.:
ORTHOGONAL POLYNOMIALS IN NUMERICAL ANALYSIS.
1. ERROR ANALYSIS OF RECURRENCE RELATIONS IN FLOATING-POINT COMPUTATION, (TO APPEAR).

[4].WILKINSON,J.H. :
ROUNDING ERRORS IN ALGEBRAIC PROCESSES.
HSO, NOTES ON APPLIED SCIENCES NO. 32, 1963.

[5].WOZNIAKOWSKI,H.:
ROUNDING ERROR ANALYSIS FOR THE EVALUATION OF A POLYNOMIAL AND SOME OF ITS DERIVATIVES.
SIAM J. OF NUM. AN. VOL 11, NO. 4, P. 780-787.

EXAMPLE OF USE:

AS A FORMAL TEST OF THE PROCEDURE DERPOL THE DERIVATIVES OF THE POLYNOMIAL 3*X**3-2*X**2+X-1 ARE CALCULATED AT X=1.

```
"BEGIN""ARRAY"A[0:3];
     "PROCEDURE"DERPOL(N,K,X,A);"CODE"31243;
     A[3]:=3;A[2]:=-2;A[1]:=1;A[0]:=-1;
     DERPOL(3,3,1,A); OUTPUT(61,"("
     "("THE 0-TH UNTIL AND INCLUDING THE 3-TH DERIVATIVES :")",
     4(BZDB)")",A[0],A[1],A[2],A[3]);
"END" EXAMPLE OF USE;
```

THE 0-TH UNTIL AND INCLUDING THE 3-TH DERIVATIVES : 1 6 14 18

SOURCE TEXT(S):

```
"CODE" 31040;
   "REAL" "PROCEDURE" POL(N,X,A);
   "VALUE" N,X;"INTEGER" N;"REAL" X;"ARRAY" A;
   "BEGIN" "REAL" R;
      R:= 0;
      "FOR" N:= N "STEP" -1 "UNTIL" 0 "DO"
      R:=R*X  + A[N];
      POL:= R
   "END" POL;
      "EOP"
```

```
"CODE" 31241;
      "PROCEDURE" TAYPOL(N,K,X,A);
      "VALUE" N,K,X;
      "INTEGER" N,K;"REAL" X;"ARRAY" A;
      "IF" X^= 0 "THEN"
      "BEGIN" "INTEGER" I,J,NM1;
         "REAL" XJ,AA,H;
         XJ:=1;
         "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
         "BEGIN" XJ:=XJ*X;A[J]:=A[J]*XJ "END";
         AA:=A[N];NM1:=N-1;
         "FOR" J:= 0 "STEP" 1 "UNTIL" K "DO"
         "BEGIN" H:=AA;
            "FOR" I:= NM1 "STEP" -1 "UNTIL" J "DO"
            H:= A[ I]:=A[I]+H
         "END"
      "END" "ELSE"
      "FOR" K:= K "STEP" -1 "UNTIL" 1 "DO" A[K]:=0;
         "EOP"

"CODE" 31242;
      "PROCEDURE" NORDERPOL (N,K,X,A);
      "VALUE" N,K,X;
      "INTEGER" N,K;"REAL" X;"ARRAY" A;
      "IF" X^= 0 "THEN"
      "BEGIN" "INTEGER" I,J,NM1;
         "REAL" XJ,AA,H;
         "ARRAY" XX[0:N];
         XJ:=1;
         "FOR" J:= 1 "STEP" 1 "UNTIL" N "DO"
         "BEGIN" XJ:=XX[J]:=XJ*X;A[J]:=A[J]*XJ "END";
         H:=AA:=A[N];NM1:=N-1;
         "FOR" I:= NM1 "STEP" -1 "UNTIL" 0 "DO" H:= A[I]:=A[I]+H;
         "FOR" J:= 1 "STEP" 1 "UNTIL" K "DO"
         "BEGIN" H:=AA;
            "FOR" I:= NM1 "STEP" -1 "UNTIL" J "DO"
            H:= A[ I]:=A[I]+H;
            A[J]:=H/XX[J]
         "END"
      "END" NORDERPOL ;
         "EOP"

"CODE" 31243;
      "PROCEDURE" DERPOL (N,K,X,A);
      "VALUE" N,K,X;
      "INTEGER" N,K;"REAL" X;"ARRAY" A;
      "BEGIN" "INTEGER" J; "REAL" FAC;
         "PROCEDURE"NORDERPOL(N,K,X,A);"CODE" 31242;
         FAC:=1;
         NORDERPOL (N,K,X,A);
         "FOR" J:= 2 "STEP" 1 "UNTIL" K "DO"
         "BEGIN" FAC:=FAC*J;A[J]:=A[J]*FAC "END"
      "END" DERPOL ;
         "EOP"
```

AUTHOR: C.G. VAN DER LAAN

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

INSTITUTE: REKENCENTRUM RIJKSUNIVERSITEIT GRONINGEN

RECEIVED: 780601

BRIEF DESCRIPTION:

THIS SECTION CONTAINS SIX PROCEDURES FOR THE EVALUATION OF ORTHOGONAL POLYNOMIALS:

ORTPOL,ORTPOLSYM: EVALUATE AN ORTHOGONAL POLYNOMIAL,

ALLORTPOL,ALLORTPOLSYM: EVALUATE ALL ORTHOGONAL POLYNOMIALS OF DEGREE LESS THAN A GIVEN POSITIVE INTEGER.

SUMORTPOL,SUMORTPOLSYM: EVALUATE A SERIES OF ORTHOGONAL POLYNOMIALS.

THE PROCEDURES ENDING WITH SYM ARE EFFICIENT VERSIONS FOR SYMMETRICAL POLYNOMIALS (I.E. ODD INDEXED POLYNOMIALS THAT ARE ODD FUNCTIONS AND EVEN INDEXED POLYNOMIALS THAT ARE EVEN FUNCTIONS);

KEYWORDS:

ORTHOGONAL POLYNOMIAL,
SERIES OF ORTHOGONAL POLYNOMIALS,
LINEAR THREE TERM (IN)HOMOGENEOUS RECURRENCE RELATION.

DATA AND RESULTS:

ORTHOGONAL POLYNOMIALS CAN BE CHARACTERIZED BY A RECURRENCE RELATION OF THE FOLLOWING FORM

```
A1[K] * F[K+1](X) = (A2[K] + X * A3[K]) * F[K](X)
                    - A4[K] * F[K-1](X),
```

WHERE AI[K] ARE REAL NUMBERS. SEE FOR INSTANCE TABLE 22.7 IN ABRAMOWITZ AND STEGUN (1964) FOR THE CLASSICAL ORTHOGONAL POLYNOMIALS. BY AN ELEMENTARY TRANSFORMATION, THE COEFFICIENTS IN THE RECURRENCE RELATION ABOVE CAN BE MADE SUCH THAT THE RECURRENCE RELATION IS GIVEN BY

```
P[K+1](X) = (X - B[K]) * P[K](X) - C[K] * P[K-1](X),

P[0](X) = 1, P[1](X) = X - B[0].
```

IN THIS WAY WE OBTAIN A NORMALIZATION OF THE ORTHOGONAL POLYNOMIAL SUCH THAT THE LEADING COEFFICIENT IN THE EXPLICIT REPRESENTATION OF P[K](X) EQUALS 1.
AS A CONSEQUENCE THE FOLLOWING RELATION HOLDS BETWEEN THE VALUES OBTAINED BY OUR PROCEDURES (E.G. ORTPOL) AND THE VALUES FROM THE REPRESENTATION IN ABRAMOWITZ AND STEGUN (1964) (I.C. F)

$$ORTPOL[N](X) = \prod_{K=0}^{N-1} (A1[K]/A3[K]) * F[N](X) ,$$

WHERE A1[K], A3[K], F[N] ARE DETERMINED BY 22.4 AND 22.7 IN ABRAMOWITZ AND STEGUN(1964). WE NOTICE THAT OVERFLOW/UNDERFLOW MAY OCCUR EARLIER AS A CONSEQUENCE OF OUR NORMALIZATION.
IN ORDER TO AVOID MISTAKES WHEN OBTAINING THE RECURRENCE COEFFICIENTS THE FOLLOWING TABLE GIVES THE RECURRENCE COEFFICIENTS FOR THE CLASSICAL ORTHOGONAL POLYNOMIALS (NOTE THAT THE FIRST AND SECOND POLYNOMIAL ARE DEFINED BY THE NORMALIZATION AND B[0]):

POLYNOMIAL KIND	RECURRENCE COEFFICIENTS	
	B[K]	C[K]
CHEBYSHEV (1-ST KIND)	0	1/2 , K=1 1/4 , K>1
CHEBYSHEV (2-ND KIND)	0	1/4
LEGENDRE	0	K**2/(4*K**2-1)
JACOBI	-(ALPHA**2-BETA**2)/ ((2*K+ALPHA+BETA)* (2*K+ALPHA+BETA+2))	4*(1+ALPHA)* (1+BETA)/((ALPHA+ BETA+2)**2*(ALPHA+ BETA+3)) , K=1 4*K*(K+ALPHA)* (K+BETA)*(K+ALPHA+ BETA)/((2*K+ALPHA+ BETA)**2*((2*K+ ALPHA+BETA)**2-1)) , K>1 (ALPHA,BETA > -1)
LAGUERRE	2*K+ALPHA+1	K*(K+ALPHA)
HERMITE	0	K/2

IN GENERAL THE RECURRENCE COEFFICIENTS MAY BE OBTAINED BY USE OF THE PROCEDURE RECCOF. THE ABOVE TABLE IS OBTAINED BY ADAPTION OF TABLE 22.7,ABRAMOWITZ AND STEGUN (1964) P. 782, AS FOLLOWS: (NOTE THAT N>=1; FOR N=0 CONSULT 22.4)

```
B[I]:=-A2[I]/A3[I]                      , I=0,1,...,N-1
C[I]:=(A4[I]*A1[I-1])/(A3[I]*A3[I-1])  , I=1,2,...,N-1.
```

METHOD AND PERFORMANCE:

LET THE ORTHOGONAL POLYNOMIAL BE DEFINED BY

```
   P[K+1](X)=(X-B[K])*P[K](X)-C[K]*P[K-1](X)   , K=1,2,...N-1
```

WHERE B[0:N-1], C[1:N-1] CONTAIN THE RECURRENCE COEFFICIENTS AND P[1](X)=X-B[0], P[0](X)=1 (SEE STOER 1972, P. 119). THEN

```
                                 N-1   / X-B[K]    1 \ / 1 \
   P[N](X) = (X-B[0],1) * PROD  [                   ] [   ]  , N=1,2,.....
                                 K=1   \  -C[K]    0 / \ 0 /
   AND

   A[0]+A[1]*P[1](X)+...+A[N]*P[N](X)=
                                      N    J-1   / X-B[K]    1 \ / A[J] \
            A[0] + (X-B[0],1) * SUM PROD  [                   ] [      ] .
                                     J=1  K=1   \  -C[K]    0 / \  0   /
```

THESE EXPRESSIONS ARE EVALUATED BY A GENERALISED HORNER RULE.

(SEE ALSO LUKE, 1969, P. 327).

REFERENCES:

ABRAMOWITZ, M. & I. STEGUN (1964):
HANDBOOK OF MATHEMATICAL FUNCTIONS,
NATIONAL BUREAU OF STANDARDS, WASHINGTON D.C.

LUKE, Y.L. (1969):
THE SPECIAL FUNCTIONS AND THEIR APPROXIMATIONS I,
ACADEMIC PRESS, LONDON, NEW YORK.

STOER, J. (1972):
EINFUEHRUNG IN DIE NUMERISCHE MATHEMATIK I,
SPRINGER VERLAG, BERLIN.

SUBSECTION: ORTPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"REAL" "PROCEDURE" ORTPOL(N,X,B,C); "VALUE" N,X;
"INTEGER" N; "REAL" X; "ARRAY" B,C;
"CODE" 31044;
```

ORTPOL DELIVERS THE VALUE OF THE ORTHOGONAL POLYNOMIAL OF DEGREE N FOR THE ARGUMENT X AS DETERMINED BY THE RECURRENCE COEFFICIENTS.

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
X:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE ARGUMENT OF THE ORTHOGONAL POLYNOMIAL;
B,C:    <ARRAY IDENTIFIER>;
        "ARRAY" B[0:N-1], C[1:N-1];
        ENTRY: THE RECURRENCE COEFFICIENTS (SEE DATA AND RESULTS).
```

SUBSECTION: ORTPOLSYM.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"REAL" "PROCEDURE" ORTPOLSYM(N,X,C); "VALUE" N,X;
"INTEGER" N; "REAL" X; "ARRAY" C;
"CODE" 31048;
```

ORTPOLSYM DELIVERS THE VALUE OF THE ORTHOGONAL POLYNOMIAL OF DEGREE N FOR THE ARGUMENT X AS DETERMINED BY THE RECURRENCE COEFFICIENTS.

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
X:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE ARGUMENT OF THE ORTHOGONAL POLYNOMIAL;
C:      <ARRAY IDENTIFIER>;
        "ARRAY" C[1:N-1];
        ENTRY: THE RECURRENCE COEFFICIENTS (SEE DATA AND RESULTS).
```

SUBSECTION: ALLORTPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ALLORTPOL(N,X,B,C,P);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" B,C,P;
"CODE" 31045;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N,X,B,C: SEE ORTPOL       (THIS SECTION);
P:       <ARRAY IDENTIFIER>;
         "ARRAY" P[0:N];
         EXIT: P[K] CONTAINS, FOR THE ARGUMENT, THE VALUE OF THE
               K-TH ORTHOGONAL POLYNOMIAL AS DEFINED BY THE
               RECURRENCE COEFFICIENTS.
```

SUBSECTION: ALLORTPOLSYM.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" ALLORTPOLSYM(N,X,C,P);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" C,P;
"CODE" 31049;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N,X,C:   SEE ORTPOLSYM   (THIS SECTION);
P:       <ARRAY IDENTIFIER>;
         "ARRAY" P[0:N];
         EXIT: P[K] CONTAINS, FOR THE ARGUMENT, THE VALUE OF THE
               K-TH ORTHOGONAL POLYNOMIAL AS DEFINED BY THE
               RECURRENCE COEFFICIENTS.
```

SUBSECTION: SUMORTPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"REAL" "PROCEDURE" SUMORTPOL(N,X,B,C,A);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" B,C,A;
"CODE" 31047;
```

```
    SUMORTPOL : DELIVERS THE VALUE OF THE POLYNOMIAL
    A[0]+A[1]*P[1](X)+...+A[N]*P[N](X)
    WHERE P[K](X) IS THE K-TH ORTHOGONAL POLYNOMIAL.

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N,X,B,C:    SEE ORTPOL       (THIS SECTION);
    A:          <ARRAY IDENTIFIER>;
                "ARRAY" A[0:N];
                ENTRY: THE COEFFICIENTS OF THE SERIES EXPANSION
                       A[0]+A[1]*P[1](X)+...+A[N]*P[N](X)
                       WHERE P[K](X) IS THE K-TH ORTHOGONAL POLYNOMIAL
                       AS DEFINED BY THE RECURRENCE COEFFICIENTS.
```

SUBSECTION: SUMORTPOLSYM.

CALLING SEQUENCE:

```
    THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

    "REAL" "PROCEDURE" SUMORTPOLSYM(N,X,C,A);
    "VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" C,A;
    "CODE" 31058;

    SUMORTPOLSYM : DELIVERS THE VALUE OF THE POLYNOMIAL
    A[0]+A[1]*P[1](X)+...+A[N]*P[N](X)
    WHERE P[K](X) IS THE K-TH ORTHOGONAL POLYNOMIAL.

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N,X,C:      SEE ORTPOLSYM    (THIS SECTION);
    A:          <ARRAY IDENTIFIER>;
                "ARRAY" A[0:N];
                ENTRY: THE COEFFICIENTS OF THE SERIES EXPANSION
                       A[0]+A[1]*P[1](X)+...+A[N]*P[N](X)
                       WHERE P[K](X) IS THE K-TH ORTHOGONAL POLYNOMIAL
                       AS DEFINED BY THE RECURRENCE COEFFICIENTS.
```

EXAMPLE OF USE:

THE FOLLOWING PROGRAM DELIVERS THE VALUES OF THE LAGUERRE POLYNOMIAL OF DEGREES 0,1,2,3,4,5 FOR X=0 BY MEANS OF THE PROCEDURE ALLORTPOL (B[K]=2*K+1, C[K]=K**2):

```
"BEGIN" "ARRAY" B[0:4], C[1:4], P[0:5];
  "INTEGER" I;
  "PROCEDURE" ALLORTPOL(N,X,B,C,P); "CODE" 31045;
  B[0]:=1;
  "FOR" I:=1 "STEP" 1 "UNTIL"  4 "DO"
  "BEGIN" B[I]:=2*I+1;
    C[I]:=I**2;
  "END" I;
  ALLORTPOL(5,0,B,C,P);
  OUTPUT(61,"("2/,6(2B,+2ZD.D)")",(P[I],I:=0:5));
"END" PROGRAM;
```

RESULTS (NOTE THE DIFFERENCE WITH FIGURE 22.9 IN ABRAMOWITZ AND STEGUN (1964) BECAUSE OF THE NORMALIZATION):

```
+1.0  -1.0  +2.0  -6.0  +24.0  -120.0
```

SOURCE TEXTS:

```
"CODE" 31044;
"REAL" "PROCEDURE" ORTPOL(N,X,B,C);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" B,C;
"IF" N=0 "THEN" ORTPOL:=1 "ELSE"
"BEGIN" "INTEGER" K,L; "REAL" R,S,H;
  R:=X-B[0]; S:=1; L:=N-1;
  "FOR" K:=1 "STEP" 1 "UNTIL" L "DO"
"BEGIN" H:=R;
  R:=(X-B[K])*R-C[K]*S;
  S:=H;
"END";
  ORTPOL:=R;
"END" ORTPOL;
        "EOP"


"CODE" 31048;
"REAL" "PROCEDURE" ORTPOLSYM(N,X,C);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" C;
"IF" N=0 "THEN" ORTPOLSYM:=1.0 "ELSE"
"BEGIN"
      "INTEGER" K,L; "REAL" R,S,H;
      R:=X; S:=1.0; L:=N-1;
      "FOR" K:=1 "STEP" 1 "UNTIL" L "DO"
      "BEGIN"
           H:=R; R:=X*R-C[K]*S;
           S:=H
      "END";
      ORTPOLSYM:=R
"END" ORTPOLSYM;
        "EOP"


"CODE" 31045;
"PROCEDURE"ALLORTPOL(N,X,B,C)RESULTS:(P);
"VALUE" N,X; "INTEGER" N;"REAL" X;"ARRAY" B,C,P;
"IF" N=0 "THEN" P[0]:=1"ELSE"
"BEGIN" "INTEGER" K,K1; "REAL" R,S,H;
  R:=P[1]:=X-B[0]; S:=P[0]:=1;K:=1;
  "FOR" K1:=2 "STEP" 1 "UNTIL" N "DO"
"BEGIN" H:=R;  P[K1]:=R:=(X-B[K])*R-C[K]*S;
  S:=H; K:=K1;
"END";
"END" ALLORTPOL
        "EOP"
```

```
"CODE" 31049;
"PROCEDURE" ALLORTPOLSYM(N,X,C)RESULTS:(P);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" C,P;
"IF" N=0 "THEN" P[0]:=1.0 "ELSE"
"BEGIN"
      "INTEGER" K; "REAL" R,S,H;
      R:=P[1]:=X; S:=P[0]:=1.0;
      "FOR" K:=2 "STEP" 1 "UNTIL" N "DO"
      "BEGIN"
            H:=R;
            P[K]:=R:=X*R-C[K-1]*S;
            S:=H
      "END";
"END" ALLORTPOLSYM;
         "EOP"
```

```
"CODE" 31047;
"REAL" "PROCEDURE" SUMORTPOL(N,X,B,C,A);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" B,C,A;
"IF" N=0 "THEN" SUMORTPOL:=A[0] "ELSE"
"BEGIN" "INTEGER" K; "REAL" H,R,S;
  R:=A[N]; S:=0;
  "FOR" K:=N-1 "STEP" -1 "UNTIL" 1 "DO"
  "BEGIN"
    H:=R;
    R:=A[K]+(X-B[K])*R+S;
    S:=        -C[K]*H
  "END";
  SUMORTPOL:=A[0]+(X-B[0])*R+S
"END" SUMORTPOL;
         "EOP"
```

```
"CODE" 31058;
"REAL" "PROCEDURE" SUMORTPOLSYM(N,X,C,A);
"VALUE" N,X; "INTEGER" N; "REAL" X; "ARRAY" C,A;
"IF" N=0 "THEN" SUMORTPOLSYM:=A[0] "ELSE"
"BEGIN"
      "INTEGER" K; "REAL" H,R,S;
      R:=A[N]; S:=0;
      "FOR" K:=N-1 "STEP" -1 "UNTIL" 1 "DO"
      "BEGIN"
            H:=R;
            R:=A[K]+X*R+S;
            S:= -C[K]*H
      "END";
      SUMORTPOLSYM:=A[0]+X*R+S
"END" SUMORTPOLSYM;
         "EOP"
```

AUTHOR: C.G. VAN DER LAAN.

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

INSTITUTE: REKENCENTRUM RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED: 780601.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS FOUR PROCEDURES ABOUT CHEBYSHEV POLYNOMIALS OF THE FIRST KIND:

- CHEPOLSUM: EVALUATES A (FINITE) SUM OF CHEBYSHEV POLYNOMIALS,
- ODDCHEPOLSUM: EVALUATES A (FINITE) SUM OF CHEBYSHEV POLYNOMIALS OF ODD DEGREE,
- CHEPOL: EVALUATES A CHEBYSHEV POLYNOMIAL,
- ALLCHEPOL: EVALUATES ALL CHEBYSHEV POLYNOMIALS WITH DEGREE LESS THAN A GIVEN POSITIVE INTEGER.

KEYWORDS:

(FINITE) SUM OF (SHIFTED) CHEBYSHEV POLYNOMIALS OF THE FIRST KIND,
GOERTZEL,WATT,CLENSHAW,GENERALIZED HORNER ALGORITHM,
LINEAR THREE TERM (INHOMOGENEOUS) RECURRENCE RELATION.

REFERENCES:

CLENSHAW, C.W. (1962):
CHEBYSHEV SERIES FOR MATHEMATICAL FUNCTIONS.
MATH. TAB. NAT. PHYS. LAB. 5, LONDON. HMO.

FOX, L. & I.B. PARKER (1972):
CHEBYSHEV POLYNOMIALS IN NUMERICAL ANALYSIS.
OXFORD UNIVERSITY PRESS.

RIVLIN, T.J. (1974):
THE CHEBYSHEV POLYNOMIALS.
WILEY.

STOER,J.(1972):
EINFUEHRUNG IN DIE NUMERISCHE MATHEMATIK 1.
HEIDELBERGER TASCHENBUECHER 105,SPRINGER-VERLAG.

SUBSECTION: CHEPOLSUM.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"REAL""PROCEDURE"CHEPOLSUM(N,X,A);
"VALUE"N,X;"INTEGER"N;"REAL"X;"ARRAY"A;
"CODE"31046;
```

CHEPOLSUM:=THE VALUE OF THE CHEBYSHEV SUM
A[0] + A[1]*T[1](X) + ... + A[N]*T[N](X),
WHERE T[1](X),....,T[N](X) ARE CHEBYSHEV POLYNOMIALS
OF THE FIRST KIND, OF DEGREE 1,....,N, RESPECTIVELY.

THE MEANING OF THE FORMAL PARAMETERS IS :

N : <ARITHMETIC EXPRESSION>;
    ENTRY: THE DEGREE OF THE POLYNOMIAL REPRESENTED BY THE
    CHEBYSHEV SUM (N>=0);

X : <ARITHMETIC EXPRESSION>;
    ENTRY: THE ARGUMENT OF THE CHEBYSHEV POLYNOMIALS , ABS(X)<=1;

A : <ARRAY IDENTIFIER>;
    "ARRAY" A[0:N];
    ENTRY: THE COEFFICIENTS OF THE CHEBYSHEV SUM MUST BE GIVEN IN
    ARRAY A, WHERE A[K] IS THE COEFFICIENT OF THE CHEBYSHEV
    POLYNOMIAL OF DEGREE K, 0<=K<=N.

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N.

METHOD AND PERFORMANCE:

```
     N                                   N  / 2*X   1 \ K   / A[K] \
    SUM A[K]*T[K](X) = (1,X) * SUM [             ]  * [         ]
    K=0                                 K=0 \  -1   0 /     \  0   /
WE USE THE CLENSHAW OR GENERALIZED HORNER ALGORITHM:
                            N
                           SUM A[K]*T[K](X) = (1,X) *
                           K=0

/ / A[0] \    / 2*X 1 \    / / A[1] \       / 2*X 1 \    / A[N] \ \  \
[ [       ] + [        ] * [ [       ] +..+ [        ] * [       ] ]..].
\ \  0   /    \ -1  0 /    \ \  0   /       \ -1 0  /    \  0   / /  /
```

```
THIS PROCEDURE MAY BE USED:
- TO EVALUATE THE SUM OF CHEBYSHEV POLYNOMIALS OF EVEN DEGREE
     N
    SUM A[K]*T[2*K](X) ,
    K=0
  BY THE CALL OF
    CHEPOLSUM(N,"IF" ABS(X)<ARREB "THEN" -1 "ELSE" 2*X*X-1,A)
  BECAUSE OF
    T[2*K](X) = T[K](T[2](X));
  (ARREB DENOTES THE MACHINE PRECISION)
- TO EVALUATE THE SUM OF SHIFTED CHEBYSHEV POLYNOMIALS FOR 0<=X<=1
     N
    SUM A[K]*T'[K](X) ,
    K=0
  BY THE CALL OF
    CHEPOLSUM(N,2*X-1,A)
  BECAUSE OF
    T'[K](X) = T[K](2*X-1).
```

EXAMPLE OF USE :

THE POLYNOMIAL : 1 + 1/2*T[1](X) + 1/4*T[2](X) IS EVALUATED FOR X = -1,0,1, WHERE T[1](X) AND T[2](X) ARE THE CHEBYSHEV POLYNOMIALS OF FIRST AND SECOND DEGREE, RESPECTIVELY.

```
"BEGIN""ARRAY"A[0:2];
        "REAL""PROCEDURE"CHEPOLSUM(N,X,A);
        "VALUE"N,X;"INTEGER"N;"REAL"X;"ARRAY"A;
        "CODE"31046;
        A[2]:=.25;A[1]:=.5;A[0]:=1;
        OUTPUT(61,"("3(BZ.DD)")",CHEPOLSUM(2,-1,A),CHEPOLSUM(2,0,A),
        CHEPOLSUM(2,1,A))
"END"

     .75  .75 1.75
```

SUBSECTION: ODDCHEPOLSUM.

CALLING SEQUENCE:

```
THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"REAL""PROCEDURE" ODDCHEPOLSUM(N,X,A);
"VALUE"N,X;"INTEGER"N;"REAL"X;"ARRAY"A;
"CODE"31059;
```

ODDCHEPOLSUM:= THE VALUE OF THE CHEBYSHEV SUM
A[1]*T[1](X) + ... + A[N]*T[2*N+1](X),
WHERE T[1](X),....,T[2*N+1](X) ARE CHEBYSHEV POLYNOMIALS
OF THE FIRST KIND, OF (ODD) DEGREE 1,....,2*N+1.

THE MEANING OF THE FORMAL PARAMETERS IS:
N : <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL REPRESENTED BY
THE CHEBYSHEV SUM IS 2*N+1 (N>=0);
X : <ARITHMETIC EXPRESSION>;
ENTRY: THE ARGUMENT OF THE CHEBYSHEV POLYNOMIALS, ABS(X)<=1;
A : <ARRAY IDENTIFIER>;
"ARRAY" A[0:N];
ENTRY: THE COEFFICIENTS OF THE CHEBYSHEV SUM MUST BE GIVEN IN
ARRAY A, WHERE A[K] IS THE COEFFICIENT OF THE CHEBYSHEV
POLYNOMIAL OF DEGREE 2*K+1, 0<=K<=N.

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N.

METHOD AND PERFORMANCE:

FROM THE REPRESENTATION, FOR ABS(X)<=1,

```
 N                                   N  / 2*T[2](X) -1 \ K  /A[K] \
SUM A[K]*T[2*K+1](X) = X * (1,-1) * SUM [              ]  * [     ]
K=0                                 K=0 \    1       0 /    \  0  /
```

WE USE THE CLENSHAW OR GENERALIZED HORNER ALGORITHM:

```
                    N                                  / / A[0] \
                .  SUM A[K]*T[2*K+1](X) = X * (1,-1) * [ [      ] +
                   K=0                                 \ \   0  /

/ 2*T[2](X) -1 \    / / A[1] \     / 2*T[2](X) -1 \    / A[N] \ \  \
[              ] *  [ [      ]+...+[              ] *  [      ] ]..].
\     1      0 /    \ \   0  /     \     1      0 /    \  0   / /  /
```

THIS PROCEDURE MAY BE USED TO EVALUATE THE SUM OF SHIFTED CHEBYSHEV
POLYNOMIALS OF ODD DEGREE FOR 0<=X<=1,

```
 N
SUM A[K]*T'[2*K+1](X) ,
K=0
```

BY THE CALL OF
ODDCHEPOLSUM(N,2*X-1,A)
BECAUSE OF
T'[K](X) = T[K](2*X-1).

EXAMPLE OF USE:

THE POLYNOMIAL 1/2*T[1](X) + 1/5*T[3](X) IS EVALUATED FOR X=-1,0,1, WHERE T[1](X) AND T[3](X) ARE CHEBYSHEV POLYNOMIALS OF THE FIRST AND THIRD DEGREE, RESPECTIVELY.

```
"BEGIN"
     "ARRAY"A[0:1];
     "REAL""PROCEDURE"ODDCHEPOLSUM(N,X,A);
     "VALUE"N,X;"INTEGER"N;"REAL"X;"ARRAY"A;
     "CODE"31049;
     A[1]:=.2;A[0]:=.5;
     OUTPUT(61,"("/,3(B,-7.DD)")",ODDCHEPOLSUM(1,-1,A),
                                  ODDCHEPOLSUM(1,0,A),
                                  ODDCHEPOLSUM(1,1,A));
"END"
```

-.70 .00 .70

SUBSECTION: CHEPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"REAL""PROCEDURE"CHEPOL(N,X);
"VALUE"N,X;"INTEGER"N;"REAL"X;
"CODE"31042;
```

CHEPOL:=THE VALUE OF THE CHEBYSHEV POLYNOMIAL OF THE FIRST KIND OF
DEGREE N FOR THE ARGUMENT X.

THE MEANING OF THE FORMAL PARAMETERS IS:

N : <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL (N>=0);

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

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N.

METHOD AND PERFORMANCE: SEE ALLCHEPOL (NEXT SUBSECTION).

EXAMPLE OF USE: SEE NEXT SUBSECTION.

SUBSECTION: ALLCHEPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE"ALLCHEPOL(N,X,T);
"VALUE"N,X;"INTEGER"N;"REAL"X;"REAL""ARRAY"T;
"CODE"31043;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N : <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE LAST POLYNOMIAL (N>=0);

X : <ARITHMETIC EXPRESSION>;
ENTRY: THE ARGUMENT OF THE CHEBYSHEV POLYNOMIALS, ABS(X)<=1;

T : <ARRAY IDENTIFIER>;
"ARRAY" T[0:N];
EXIT: THE VALUES OF THE CHEBYSHEV POLYNOMIALS OF THE FIRST KIND OF DEGREES 0,1,...,N , FOR THE ARGUMENT X, ARE DELIVERED IN T[0],T[1],...,T[N], RESPECTIVELY.

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N.

METHOD AND PERFORMANCE:

FOR A DESCRIPTION OF THE ALGORITHM SEE STOER,1972,P.21.
THE MAXIMUM (ABSOLUTE) VALUE OF THE CHEBYSHEV POLYNOMIAL EQUALS 1 AS A NORMALIZATION.
AN UPPER BOUND FOR THE (ABSOLUTE) ERROR IS A QUADRATIC FUNCTION OF THE DEGREE OF THE CHEBYSHEV POLYNOMIAL.THIS UPPER BOUND IS A ROUGH OVER-ESTIMATE FOR THE SPECIAL CASE ABS(X)<.5 (STOER,1972, P. 21-24).

EXAMPLE OF USE :

BY THE PROCEDURE (ALL)CHEPOL THE CHEBYSHEV POLYNOMIALS OF THE FIRST KIND OF DEGREES 0,1,2 ARE EVALUATED AT -1,0,1.

```
"BEGIN"
    "ARRAY"T[0:2];
    "REAL""PROCEDURE"CHEPOL(N,X);
    "VALUE"N,X;"INTEGER"N;"REAL"X;
    "CODE"31042;
    "PROCEDURE"ALLCHEPOL(N,X,T);
    "VALUE"N,X;"INTEGER"N;"REAL"X;"REAL""ARRAY"T;
    "CODE"31043;
    ALLCHEPOL(2,-1,T);OUTPUT(61,"("/,3(-DB)")",T[0],T[1],T[2]);
    ALLCHEPOL(2, 0,T);OUTPUT(61,"("/,3(-DB)")",T[0],T[1],T[2]);
    ALLCHEPOL(2, 1,T);OUTPUT(61,"("/,3(-DB)")",T[0],T[1],T[2]);
    OUTPUT(61,"("/,3(/,-D)")",CHEPOL(2,-1),CHEPOL(2,0),CHEPOL(2,1))
"END"
```

```
 1 -1  1
 1  0 -1
 1  1  1

 1
-1
 1
```

SOURCE TEXT(S):

```
"CODE"31046;
"REAL" "PROCEDURE" CHEPOLSUM(N,X,A);
"VALUE" N,X;"INTEGER" N;"REAL" X;"ARRAY" A;
"IF" N=0 "THEN" CHEPOLSUM:=A[0] "ELSE"
"IF" N=1 "THEN" CHEPOLSUM:=A[0]+A[1]*X "ELSE"
"BEGIN" "INTEGER" K;"REAL" H,R,S,TX;
   TX:=X+X;R:=A[N];
   H:=A[N-1]+R*TX;
   "FOR" K:=N-2 "STEP" -1 "UNTIL" 1 "DO"
   "BEGIN" S:=R;R:=H;
      H:=A[K]+R*TX-S
   "END"K;
   CHEPOLSUM:=A[0]-R+H*X
"END" CHEPOLSUM;
        "EOP"
```

```
"CODE"31059;
"REAL""PROCEDURE"ODDCHEPOLSUM(N,X,A);
"VALUE"N,X;"INTEGER"N;"REAL"X;"ARRAY"A;
"COMMENT" ODDCHEPOLSUM:=A[0]T[1](X)+A[1]T[3](X)+....+A[N]T[2N+1](X);
"IF" N=0 "THEN" ODDCHEPOLSUM:=X*A[0] "ELSE"
"IF" N=1 "THEN" ODDCHEPOLSUM:=X*(A[0]+A[1]*(4*X*X-3)) "ELSE"
"BEGIN"
   "INTEGER" K;
   "REAL" H,R,S,Y;
   Y:=4*X*X-2;
   R:=A[N];
   H:=A[N-1]+R*Y;
   "FOR" K:=N-2 "STEP" -1 "UNTIL" 0 "DO"
   "BEGIN"
     S:=R;
     R:=H;
     H:=A[K]+R*Y-S;
   "END" K;
   ODDCHEPOLSUM:=X*(H-R);
"END" ODDCHEPOLSUM;
         "EOP"


"CODE"31042;
"REAL""PROCEDURE"CHEPOL(N,X);
"VALUE"N,X;"INTEGER"N;"REAL"X;
"IF" N = 0 "THEN" CHEPOL :=1 "ELSE"
"IF" N = 1 "THEN" CHEPOL :=X "ELSE"
"BEGIN""INTEGER"I;"REAL"T1,T2,H,X2;
   T2:=X;T1:=1;X2:=X+X;
   "FOR"I:=2"STEP"1"UNTIL"N"DO"
   "BEGIN"H:=X2*T2-T1;T1:=T2;T2:=H"END";
   CHEPOL:=H
"END"CHEPOL;
         "EOP"


"CODE"31043;
"PROCEDURE"ALLCHEPOL(N,X,T);
"VALUE"N,X;"INTEGER"N;"REAL"X;"REAL""ARRAY"T;
"IF" N = 0 "THEN" T[0] :=1 "ELSE"
"IF" N = 1 "THEN" "BEGIN" T[0] := 1; T[1] := X "END" "ELSE"
"BEGIN""INTEGER"I;"REAL"T1,T2,H,X2;
   T[0]:=T1:=1;T[1]:=T2:=X;X2:=X+X;
   "FOR"I:=2"STEP"1"UNTIL"N"DO"
   "BEGIN"T[I]:=H:=X2*T2-T1;T1:=T2;T2:=H"END"
"END"ALLCHEPOL;
         "EOP"
```

AUTHOR: C.G. VAN DER LAAN.

INSTITUTE: RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED: 740701.

BRIEF DESCRIPTION:

THIS SECTION CONTAINS THE PROCEDURES:
SINSER FOR EVALUATING A SINE SERIES;
COSSER FOR EVALUTING A COSINE SERIES;
FOUSER,FOUSER1,FOUSER2 FOR EVALUATING A FOURIER SERIES
(IN FOUSER THE SERIES IS RESTRICTED TO A SERIES WITH SINE
COEFFICIENTS EQUAL TO COSINE COEFFICIENTS);
COMFOUSER,COMFOUSER1,COMFOUSER2 FOR EVALUATING A COMPLEX FOURIER
SERIES
(IN COMFOUSER THE SERIES IS RESTRICTED TO A SERIES WITH REAL
COEFFICIENTS).

KEYWORDS:

FINITE FOURIER SERIES EVALUATION,
TRIGONOMETRIC POLYNOMIAL EVALUATION,
GOERTZEL,WATT,CLENSHAW,REINSCH ALGORITHM,
LINEAR THREE-TERM INHOMOGENEOUS RECURRENCE RELATION.

SUBSECTION : SINSER.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL""PROCEDURE"SINSER(N,THETA,B);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"B;

    SINSER:= THE VALUE OF THE SINE SERIES
              B[1]*SIN(THETA)+...+B[N]*SIN(N*THETA).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:        <ARITHMETIC EXPRESSION>;
              ENTRY: THE NUMBER OF TERMS IN THE SINE SERIES;
    THETA:    <ARITHMETIC EXPRESSION>;
              ENTRY: THE ARGUMENT OF THE SINE SERIES;
    B:        <ARRAY IDENTIFIER>;
              "ARRAY"B[1:N];
              ENTRY: THE COEFFICIENTS OF THE SINE SERIES.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: N MULTIPLICATIONS; 3N ADDITIONS:
3 SINE/COSINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : COSSER.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL""PROCEDURE"COSSER(N,THETA,A);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A;

    COSSER:= THE VALUE OF THE COSINE SERIES
              A[0]+A[1]*COS(THETA)+...+A[N]*COS(N*THETA).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:       <ARITHMETIC EXPRESSION>;
             ENTRY: THE DEGREE OF THE TRIGONOMETRIC POLYNOMIAL.
    THETA:   <ARITHMETIC EXPRESSION>;
             ENTRY: THE ARGUMENT OF THE COSINE SERIES.
    A:       <ARRAY IDENTIFIER>;
             "ARRAY"A[0:N];
             ENTRY: THE COEFFICIENTS OF THE COSINE SERIES.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: N MULTIPLICATIONS; 3N ADDITIONS;
2 COSINE/SINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : FOUSER.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL""PROCEDURE"FOUSER (N,THETA,A);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A;

    FOUSER := THE VALUE OF THE FOURIER SERIES
              A[0]+A[1]*(COS(THETA)+SIN(THETA))+...+A[N]*(COS(N*THETA)
              +SIN(N*THETA)).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE DEGREE OF THE TRIGONOMETRIC POLYNOMIAL;
    THETA:  <ARITHMETIC EXPRESSION>;
            ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    A:      <ARRAY IDENTIFIER>;
            "ARRAY"A[0:N];
            ENTRY: THE COEFFICIENTS OF THE (FINITE) FOURIER SERIES.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: N MULTIPLICATIONS; 3N ADDITIONS;
3 COSINE/SINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : FOUSER1.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL""PROCEDURE"FOUSER1(N,THETA,A,B);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A,B;

    FOUSER1:= THE VALUE OF THE FOURIER SERIES
              A[0]+A[1]*COS(THETA)+B[1]*SIN(THETA)+...
              +A[N]*COS(N*THETA)+B[N]*SIN(N*THETA).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:       <ARITHMETIC EXPRESSION>;
             ENTRY: THE DEGREE OF THE TRIGONOMETRIC POLYNOMIAL;
    THETA:   <ARITHMETIC EXPRESSION>;
             ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    A,B:     <ARRAY IDENTIFIER>;
             "ARRAY"A[0:N],B[1:N];
             ENTRY: THE COEFFICIENTS OF THE (FINITE) FOURIER SERIES,
                    WITH A[K] COEFFICIENT OF COS(K*THETA), (K=0,...,N)
                    AND  B[K] COEFFICIENT OF SIN(K*THETA), (K=1,...,N).
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: 4N MULTIPLICATIONS; 4N ADDITIONS;
2 COSINE/SINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : FOUSER2.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "REAL""PROCEDURE"FOUSER2(N,THETA,A,B);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A,B;

    FOUSER2:= THE VALUE OF THE FOURIER SERIES
               A[0]+A[1]*COS(THETA)+B[1]*SIN(THETA)+...
               +A[N]*COS(N*THETA)+B[N]*SIN(N*THETA).

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:       <ARITHMETIC EXPRESSION>;
             ENTRY: THE DEGREE OF THE TRIGONOMETRIC POLYNOMIAL;
    THETA:   <ARITHMETIC EXPRESSION>;
             ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    A,B:     <ARRAY IDENTIFIER>;
             "ARRAY"A[0:N],B[1:N];
             ENTRY: THE COEFFICIENTS OF THE (FINITE) FOURIER SERIES,
                    WITH A[K] COEFFICIENT OF COS(K*THETA), (K=0,...,N)
                    AND  B[K] COEFFICIENT OF SIN(K*THETA), (K=1,...,N).
```

PROCEDURES USED: SINSER = CP31090,
                 COSSER = CP31091.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: 2N MULTIPLICATIONS; 6N ADDITIONS;
6 COSINE/SINE EVALUATIONS).

LANGUAGE : ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : COMFOUSER.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE"COMFOUSER (N,THETA,A,RR,RI);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"A;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:       <ARITHMETIC EXPRESSION>;
             ENTRY: THE DEGREE OF THE POLYNOMIAL IN EXP(I*THETA);
    THETA:   <ARITHMETIC EXPRESSION>;
             ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    A:       <ARRAY IDENTIFIER>;
             "ARRAY"A[0:N];
             ENTRY: THE REAL COEFFICIENTS A[K] (K=0,...,N) IN THE SERIES
                    FN(THETA)=A[0]+A[1]*EXP(I*THETA)+...+A[N]*EXP(I*
                    THETA)**N, MUST BE GIVEN IN ARRAY A;
    RR,RI:   <VARIABLE>;
             EXIT:  THE REAL PART AND THE IMAGINARY PART OF FN(THETA)
                    ARE DELIVERED IN RR AND RI, RESPECTIVELY.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: N MULTIPLICATIONS; 3N ADDITIONS;
3 COSINE/SINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : COMFOUSER1.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE"COMFOUSER1(N,THETA,AR,AI,RR,RI);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"AR,AI;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE DEGREE OF THE POLYNOMIAL IN EXP(I*THETA);
    THETA:  <ARITHMETIC EXPRESSION>;
            ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    AR,AI:  <ARRAY IDENTIFIER>;
            "ARRAY"AR,AI[0:N];
            ENTRY: THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX
                   COEFFICIENTS C[K] (K=0,...,N) IN THE SERIES
                   FN(THETA)=C[0]+C[1]*EXP(I*THETA)+...+C[N]*EXP(I*
                            THETA)**N
                   MUST BE GIVEN IN ARRAY AR AND AI, RESPECTIVELY;
    RR,RI:  <VARIABLE>;
            EXIT:  THE REAL PART AND THE IMAGINARY PART OF FN(THETA)
                   ARE DELIVERED IN RR AND RI, RESPECTIVELY.
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N
(IN FIRST ORDER: 4N MULTIPLICATIONS; 4N ADDITIONS;
2 COSINE/SINE EVALUATIONS).

LANGUAGE: ALGOL 60.

METHOD AND PERFORMANCE : SEE COMFOUSER2 (THIS SECTION).

SUBSECTION : COMFOUSER2.

CALLING SEQUENCE:

```
    THE HEADING OF THE PROCEDURE READS:
    "PROCEDURE"COMFOUSER2(N,THETA,AR,AI,RR,RI);
    "VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"AR,AI;

    THE MEANING OF THE FORMAL PARAMETERS IS:
    N:      <ARITHMETIC EXPRESSION>;
            ENTRY: THE DEGREE OF THE POLYNOMIAL IN EXP(I*THETA);
    THETA:  <ARITHMETIC EXPRESSION>;
            ENTRY: THE ARGUMENT OF THE FOURIER SERIES;
    AR,AI:  <ARRAY IDENTIFIER>;
            "ARRAY"AR,AI[0:N];
            ENTRY: THE REAL PART AND THE IMAGINARY PART OF THE COMPLEX
                   COEFFICIENTS C[K] (K=0,...,N) IN THE SERIES
                   FN(THETA)=C[0]+C[1]*EXP(I*THETA)+...+C[N]*EXP(I*
                             THETA)**N
                   MUST BE GIVEN IN ARRAY AR AND AI, RESPECTIVELY;
    RR,RI:  <VARIABLE>;
            EXIT:  THE REAL PART AND THE IMAGINARY PART OF FN(THETA)
                   ARE DELIVERED IN RR AND RI, RESPECTIVELY.
```

PROCEDURES USED: COMFOUSER= CP31095.

```
RUNNING TIME: PROPORTIONAL TO N
              (IN FIRST ORDER: 2N MULTIPLICATIONS; 6N ADDITIONS;
               6 COSINE/SINE EVALUATIONS).
```

LANGUAGE: ALGOL 60.

METHOD AND PERFORANCE:

```
    FOR THE EVALUATION OF A FINITE FOURIER SERIES
    (=TRIGONOMETRIC POLYNOMIAL OF DEGREE N SEE POLYA AND SZEGOE, 1971,
    P. 76)
    FN(THETA)=A[0]+A[1]*COS(THETA)+B[1]*SIN(THETA)+...+
              A[N]*COS(N*THETA)+B[N]*SIN(N*THETA),
    TWO ALGORITHMS ARE USED:
```

```
1.  HORNER SCHEME
    LET C[K]=A[K]+I*B[K], K=0,...,N
    AND Z=EXP(-I*THETA)
    THEN
        FN(THETA)=RE(C[0]+C[1]*Z+...+C[N]*Z**N).
    THE ALGORITHM IS GIVEN BY:
        P:=C[N]
        P:=P*Z+C[K], K=N-1,...,0
        FN(THETA):=RE(P).
    (FOUSER1)
2.  A COMBINATION OF THE CLENSHAW ALGORITHM (SEE GENTLEMAN(1969.II)
    , VAN DER LAAN, LUKE(1969, P.327-329) OR STOER(1972, P.62,63))
    AND THE MODIFICATION OF REINSCH (SEE REINSCH(1967), VAN DER
    LAAN, STOER(1972, P.64,65)).
    (SINSER,COSSER,FOUSER,FOUSER2)
A MODIFICATION OF THE IDEA OF NEWBERY IS NOT IMPLEMENTED BECAUSE
OF THE INTRODUCTION OF SINE (COSINE) TERMS IN A COSINE (SINE)
SERIES AND THE THE INEFFICIENCY OF THE ALGORITHM (SEE VAN DER
LAAN OR NEWBERY(1973)).

FOR THE EVALUATION OF A FINITE COMPLEX FOURIER SERIES
FN(THETA)=AR[0]+I*AI[0]+(AR[1]+I*AI[1])*EXP(I*THETA)+...
          +(AR[N]+I*AI[N])*EXP(I*THETA)**N,
TWO ALGORITHMS, IN REAL ARITHMETIC, ARE USED:
1. HORNER SCHEME
   LET C[K]=AR[K]+I*AI[K], K=0,...,N
   AND Z=EXP(I*THETA)
   THEN
       FN(THETA)=C[0]+C[1]*Z+...+C[N]*Z**N.
   THE ALGORITHM IS GIVEN BY
       P:=C[N]
       P:=P*Z+C[K], K=N-1,N-2,...,0
       FN(THETA):=P.
   (COMFOUSER1)
2. A COMBINATION OF THE CLENSHAW ALGORITHM AND THE MODIFICATION OF
   REINSCH.
   LET CAR=AR[0]+AR[1]*COS(THETA)+...+AR[N]*COS(N*THETA),
       SAI=       AI[1]*SIN(THETA)+...+AI[N]*SIN(N*THETA),
       SAR=       AR[1]*SIN(THETA)+...+AR[N]*SIN(N*THETA),
       CAI=AI[0]+AI[1]*COS(THETA)+...+AI[N]*COS(N*THETA)
   THEN FN(THETA)=CAR-SAI+I*(SAR+CAI).
   (COMFOUSER,COMFOUSER2)
THE HORNER SCHEME IS IMPLEMENTED BECAUSE OF THE SIMPLICITY OF
THE ALGORITHM (ALTHOUGH THIS ALGORITHM IS LESS EFFICIENT THAN THE
GOERTZEL/WATT/CLENSHAW/REINSCH ALGORITHM) AND THE STABLE NATURE
OF ORTHOGONAL TRANSFORMATIONS.
A COMBINATION OF THE ALGORITHM OF GOERTZEL/WATT/CLENSHAW AND THE
MODIFICATION OF REINSCH IS IMPLEMENTED BECAUSE OF THE EFFICIENCY
OF THE GWC ALGORITHM AND THE STABILITY OF THE MODIFICATION OF
REINSCH, ESPECIALLY FOR SMALL VALUES OF THE ARGUMENT (MOD. PI).
AN UPPER BOUND FOR THE ERROR GROWTH IS GIVEN BY A LINEAR FUNCTION
OF THE DEGREE FOR BOTH (IMPLEMENTED) ALGORITHMS (SEE VAN DER LAAN).
```

REFERENCES:

GENTLEMAN,W.M.(1969):
AN ERROR ANALYSIS OF GOERTZEL'S(WATT'S) METHOD FOR COMPUTING
FOURIER COEFFICIENTS.
COMP.J.,VOL.12,P.160-165.

LAAN,C.G.VAN DER(TO APPEAR):
ORTHOGONAL POLYNOMIALS IN NUMERICAL ANALYSIS 1.
ERROR ANALYSIS OF LINEAR TWO-TERM AND THREE-TERM RECURRENCE
RELATIONS.

LUKE,Y.L.(1969):
THE SPECIAL FUNCTIONS AND THEIR APPROXIMATIONS.VOL.1.
ACADEMIC PRESS.

NEWBERY,A.C.R.(1973):
ERROR ANALYSIS FOR FOURIER SERIES EVALUATION.
MATH.COMP.,VOL.26,P.923-924.

POLYA,G. AND G.SZEGOE(1971):
AUFGABEN UND LEHRSAETZE AUS DER ANALYSIS II.
HEIDELBERGER TASCHENBUECHER 74. SPRINGER.

REINSCH,C.(1967):
A NOTE ON TRIGONOMETRIC INTERPOLATION.
BERICHT NR. 6709.
ABTEILUNG MATHEMATIK DER TECHNISCHEN UNIVERSITAET MUENCHEN.

STOER,J.(1972):
EINFUEHRUNG IN DIE NUMERISCHE MATHEMATIK 1.
HEIDELBERGER TASCHENBUECHER 105. SPRINGER.

EXAMPLE OF USE:

THE FOURIER SERIES .5+COS(THETA)+SIN(THETA)
IS EVALUATED FOR THE ARGUMENTS 0,PI/2,PI, BY MEANS OF FOUSER

```
"BEGIN""REAL"THETA,PI;"ARRAY"A[0:1];
   "REAL""PROCEDURE"FOUSER(N,THETA,A  );"CODE" 31092;
   PI:=ARCTAN(1)*4;A[0]:=.5;A[1]:=1;
   "FOR"THETA:=0,PI/2,PI"DO"
   OUTPUT(61,"("/,B-D.DD")",FOUSER(1,THETA,A))
"END"

 1.50
 1.50
-0.50
```

SOURCE TEXTS:

```
"CODE" 31090;
"REAL""PROCEDURE"SINSER(N,THETA,B);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"B;
"BEGIN""INTEGER"K;"REAL"C,CC,LAMBDA,H,DUN,UN,UN1;
    C:=COS(THETA);
    "IF"C<-.5"THEN"
    "BEGIN"LAMBDA:= 4*COS(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"1"DO"
            "BEGIN"DUN:=LAMBDA*UN-DUN+B[K] ;
                    UN:=DUN-UN;
            "END"
    "END""ELSE""IF"C> .5"THEN"
    "BEGIN"LAMBDA:=-4*SIN(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"1"DO"
            "BEGIN"DUN:=LAMBDA*UN+DUN+B[K] ;
                    UN:=DUN+UN;
            "END"
    "END""ELSE"
    "BEGIN"CC:=C+C;UN:=UN1:=0;
            "FOR"K:=N"STEP"-1"UNTIL"1"DO"
            "BEGIN"H:=CC*UN-UN1+B[K]; UN1 := UN; UN := H; "END"
    "END";
    SINSER:=UN*SIN(THETA)
"END"SINSER;
        "EOP"

"CODE" 31091;
"REAL""PROCEDURE"COSSER(N,THETA,A);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A;
"BEGIN""INTEGER"K;"REAL"C,CC,LAMBDA,H,DUN,UN,UN1;
    C:=COS(THETA);
    "IF"C<-.5"THEN"
    "BEGIN"LAMBDA:= 4*COS(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"0"DO"
            "BEGIN"UN:=DUN-UN;
                    DUN:=LAMBDA*UN-DUN+A[K]
            "END";COSSER:=DUN-LAMBDA/2*UN
    "END""ELSE""IF"C> .5"THEN"
    "BEGIN"LAMBDA:=-4*SIN(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"0"DO"
            "BEGIN"UN:=DUN+UN;
                    DUN:=LAMBDA*UN+DUN+A[K]
            "END";COSSER:=DUN-LAMBDA/2*UN
    "END""ELSE"
    "BEGIN"CC:=C+C;UN:=UN1:=0;
            "FOR"K:=N"STEP"-1"UNTIL"1"DO"
            "BEGIN"H:=CC*UN-UN1+A[K];
                    UN1:=UN;UN:=H
            "END";COSSER:=A[0]+UN*C-UN1
    "END"
"END"COSSER;
        "EOP"
```

```
"CODE" 31092;
"REAL""PROCEDURE"FOUSER (N,THETA,A);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A;
"BEGIN""INTEGER"K;"REAL"C,CC,LAMBDA,H,DUN,UN,UN1,C2,S2;
   C:=COS(THETA);
   "IF"C<-.5"THEN"
   "BEGIN"C2:=COS(THETA/2);LAMBDA:=4*C2**2;UN:=DUN:=0;
           "FOR"K:=N"STEP"-1"UNTIL"0"DO"
           "BEGIN"UN:=DUN-UN;
                  DUN:=LAMBDA*UN-DUN+A[K]
           "END";FOUSER :=DUN+2*C2*(SIN(THETA/2)-C2)*UN
   "END""ELSE""IF"C> .5"THEN"
   "BEGIN"S2:=SIN(THETA/2);LAMBDA:=-4*S2*S2;UN:=DUN:=0;
           "FOR"K:=N"STEP"-1"UNTIL"0"DO"
           "BEGIN"UN:=DUN+UN;
                  DUN:=LAMBDA*UN+DUN+A[K]
           "END";FOUSER :=DUN+2*S2*(S2+COS(THETA/2))*UN
   "END""ELSE"
   "BEGIN"CC:=C+C;UN:=UN1:=0;
           "FOR"K:=N"STEP"-1"UNTIL"1"DO"
           "BEGIN"H:=CC*UN-UN1+A[K];
                  UN1:=UN;UN:=H
           "END";FOUSER :=A[0]-UN1+(C+SIN(THETA))*UN
   "END"
"END"FOUSER;
        "EOP"

"CODE" 31093;
"REAL""PROCEDURE" FOUSER1(N,THETA,A,B);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A,B;
"BEGIN""INTEGER"I;"REAL"R,S,H,CO,SI;
   R:=S:=0;CO:=COS(THETA);SI:=SIN(THETA);
   "FOR"I:=N"STEP"-1"UNTIL"1"DO"
   "BEGIN" H:=CO*R+SI*S+A[I];
           S:=CO*S-SI*R+B[I];
           R:=H
   "END";FOUSER1:=CO*R+SI*S+A[0]
"END"FOUSER1;
        "EOP"

"CODE" 31094;
"REAL""PROCEDURE"FOUSER2(N,THETA,A,B);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA;"ARRAY"A,B;
"BEGIN"
   "REAL""PROCEDURE"SINSER(N,THETA,B);"CODE" 31090;
   "REAL""PROCEDURE"COSSER(N,THETA,A);"CODE" 31091;
   FOUSER2:=COSSER(N,THETA,A)+SINSER(N,THETA,B);
"END"FOUSER2;
        "EOP"
```

```
"CODE" 31095;
"PROCEDURE"COMFOUSER(N,THETA,A,RR,RI);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"A;
"BEGIN""INTEGER"K;"REAL"C,CC,LAMBDA,H,DUN,UN,UN1;
    C:=COS(THETA);
    "IF"C<-.5"THEN"
    "BEGIN"LAMBDA:= 4*COS(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"0"DO"
            "BEGIN"UN:=DUN-UN;
                   DUN:=LAMBDA*UN-DUN+A[K]
            "END";RR     :=DUN-LAMBDA/2*UN
    "END""ELSE""IF"C> .5"THEN"
    "BEGIN"LAMBDA:=-4*SIN(THETA/2)**2;UN:=DUN:=0;
            "FOR"K:=N"STEP"-1"UNTIL"0"DO"
            "BEGIN"UN:=DUN+UN;
                   DUN:=LAMBDA*UN+DUN+A[K]
            "END";RR     :=DUN-LAMBDA/2*UN
    "END""ELSE"
    "BEGIN"CC:=C+C;UN:=UN1:=0;
            "FOR"K:=N"STEP"-1"UNTIL"1"DO"
            "BEGIN"H:=CC*UN-UN1+A[K];
                   UN1:=UN;UN:=H
            "END";RR     :=A[0]+UN*C-UN1
    "END";RI:=UN*SIN(THETA)
"END"COMFOUSER;
        "EOP"

"CODE" 31096;
"PROCEDURE"COMFOUSER1(N,THETA,AR,AI,RR,RI);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"AR,AI;
"BEGIN""INTEGER"K;"REAL"H,HR,HI,CO,SI;
    HR:=HI:=0;CO:=COS(THETA);SI:=SIN(THETA);
    "FOR"K:=N"STEP"-1"UNTIL"1"DO"
    "BEGIN"H:=CO*HR-SI*HI+AR[K];
          HI:=CO*HI+SI*HR+AI[K];
          HR:=H
    "END";
    RR:=CO*HR-SI*HI+AR[0];
    RI:=CO*HI+SI*HR+AI[0]
"END"COMFOUSER1;
        "EOP"

"CODE" 31097;
"PROCEDURE"COMFOUSER2(N,THETA,AR,AI,RR,RI);
"VALUE"N,THETA;"INTEGER"N;"REAL"THETA,RR,RI;"ARRAY"AR,AI;
"BEGIN""REAL"CAR,CAI,SAR,SAI;
    "PROCEDURE"COMFOUSER(N,THETA,A,RR,RI);"CODE" 31095;
    COMFOUSER(N,THETA,AR,CAR,SAR);
    COMFOUSER(N,THETA,AI,CAI,SAI);
    RR:=CAR-SAI;
    RI:=CAI+SAR
"END"COMFOUSER2;
        "EOP"
```

AUTHOR : H.FIOLET

INSTITUTE: MATHEMATICAL CENTRE.

RECEIVED: 731105.

BRIEF DESCRIPTION:

JFRAC CALCULATES A TERMINATING CONTINUED FRACTION.

KEYWORDS:

CONTINUED FRACTION,
TERMINATING CONTINUED FRACTION.

CALLING SEQUENCE:

THE HEADING OF THE PROCEDURE READS:

```
"REAL" "PROCEDURE" JFRAC(N,A,B);
"VALUE" N;"INTEGER" N;"ARRAY" A,B;
"CODE" 35083;
```

JFRAC DELIVERS THE VALUE OF THE TERMINATING CONTINUED FRACTION:
B[0]+A[1]/(B[1]+A[2]/(B[2]+A[3]/(B[3]+ . . . + A[N]/B[N]))...))

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:          <ARITHMETIC EXPRESSION>;
            THE UPPER INDEX OF THE ARRAYS A AND B;
A,B:        <ARRAY IDENTIFIER>;
            "ARRAY" A[1:N];
            "ARRAY" B[0:N];
            THE ELEMENTS OF THE CONTINUED FRACTION:
            B[0]+A[1]/(B[1]+A[2]/(B[2]+A[3]/(B[3]+ . . . +
            + A[N]/B[N]))...)).
```

PROCEDURES USED: NONE.

RUNNING TIME: PROPORTIONAL TO N.

EXAMPLE OF USE:

```
"BEGIN"
"REAL" "PROCEDURE" JFRAC(N,A,B);"CODE" 35083;
"REAL" "ARRAY" P[1:10],Q[0:10];
"INTEGER" I;
"FOR" I:=1 "STEP" 1 "UNTIL" 10 "DO"
"BEGIN" P[I]:=1;Q[I]:=2 "END";
Q[0]:=1;
"FOR" I:=7 "STEP" 1 "UNTIL" 10 "DO"
OUTPUT(61,"("N/")",JFRAC(I,P,Q))
"END"
```

DELIVERS:

```
+1.4142156862745"+000
+1.4142131979695"+000
+1.4142136248949"+000
+1.4142135516461"+000
```

SOURCE TEXT:

```
"CODE" 35083;
    "REAL" "PROCEDURE" JFRAC(N,A,B);
    "VALUE" N;"INTEGER" N;"ARRAY" A,B;
    "BEGIN" "REAL" D;"INTEGER" I;
        D:=0;
        "FOR" I:=N "STEP" -1 "UNTIL" 1 "DO" D:=A[I]/(B[I]+D);
        JFRAC:=D+B[0]
    "END" JFRAC;
        "EOP"
```

AUTHOR: C.G. VAN DER LAAN.

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

INSTITUTE: REKENCENTRUM DER RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED: 780601.

BRIEF DESCRIPTION:

WE CONSIDER THE REPRESENTATIONS

```
                                   N
        POWER SUM               :  SUM A[K]*X**K,
                                   K=0
                                   N
        CHEBYSHEV SUM           :  SUM A[K]*T[K](X),
                                   K=0
                                   N
        SHIFTED CHEBYSHEV SUM   :  SUM A[K]*T'[K](X),
                                   K=0
                                   N          K-1
        NEWTON SUM              :  SUM (A[K] * PROD (X-X[J])).
                                   K=0        J=0
```

THE SHIFTED CHEBYSHEV POLYNOMIAL T'[N] IS
DEFINED BY T'[N](X) = T[N](2*X-1).

THIS SECTION CONTAINS THE TRANSFORMATIONS:

PROCEDURE NAME	TRANSFORMATION
POLCHS	POWER SUM INTO CHEBYSHEV SUM
CHSPOL	CHEBYSHEV SUM INTO POWER SUM
POLSHTCHS	POWER SUM INTO SHIFTED CHEBYSHEV SUM
SHTCHSPOL	SHIFTED CHEBYSHEV SUM INTO POWER SUM
GRNNEW	POWER SUM INTO NEWTON SUM
NEWGRN	NEWTON SUM INTO POWER SUM
LINTFMPOL	POWER SUM IN X INTO POWER SUM IN Y, X=P*Y+Q

KEYWORDS:

TRANSFORMATION OF POLYNOMIAL REPRESENTATION.

RUNNING TIME: PROPORTIONAL TO THE SQUARED DEGREE OF THE POLYNOMIAL.

METHOD AND PERFORMANCE:

ALTHOUGH THE TRANSFORMATION OF REPRESENTATIONS OF POLYNOMIALS COULD HAVE BEEN OBTAINED BY FAST EVALUATION AND FAST INTERPOLATION WE IMPLEMENTED THE ALGORITHM OF HAMMING (1973, 474,475), BECAUSE OF ITS SIMPLE APPEARANCE. AN EXPLANATION OF THE HAMMING ALGORITHM IS GIVEN IN VAN DER LAAN (1977,224-229).

REFERENCES:

HAMMING, R.W. (1973):
NUMERICAL METHODS FOR SCIENTISTS AND ENGINEERS.
MCGRAW-HILL.

LAAN, C.G. VAN DER (1977):
APPROXIMATIE VAN FUNCTIES EN DATA.
IN: RIELE, H.J.J. TE (ED.):
COLLOQUIUM NUMERIEKE PROGRAMMATUUR, DEEL 2,
MC SYLLABUS 29.2, MATHEMATISCH CENTRUM AMSTERDAM.

SUBSECTION: POLCHS.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" POLCHS(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"CODE" 31051;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[0:N];
        ENTRY: THE COEFFICIENTS OF THE POWER SUM;
        EXIT:  THE COEFFICIENTS OF THE CHEBYSHEV SUM;
```

PROCEDURES USED: NONE.

EXAMPLE OF USE: SEE NEXT SUBSECTION.

SUBSECTION: CHSPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" CHSPOL(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"CODE" 31052;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

```
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[0:N];
        ENTRY: THE COEFFICIENTS OF THE CHEBYSHEV SUM;
        EXIT:  THE COEFFICIENTS OF THE POWER SUM;
```

PROCEDURES USED: NONE.

EXAMPLE OF USE:

AS AN EXAMPLE WE TRANSFORMED
THE POWER SUM: 1 + 2*X + 3*X**2
INTO
ITS CHEBYSHEV SUM;
AS A CHECK WE TRANSFORMED THE LATTER REPRESENTATION
BACK INTO THE ORIGINAL POWER SUM.

```
"BEGIN"
    "PROCEDURE" POLCHS(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE" 31051;
    "PROCEDURE" CHSPOL(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE" 31052;
    "ARRAY" A[0:2];
    A[0]:=1; A[1]:=2; A[2]:=3;
    OUTPUT(61,"("/,16B,"("A[0]")",4B,"("A[1]")",4B,"("A[2]")"")");
    OUTPUT(61,"("/,"(" INPUT")",5B,"(":")",3(2B,+ZD.DD)")",A);
    POLCHS(2,A);
    OUTPUT(61,"("/,"(" POLCHS")",4B,"(":")",3(2B,+ZD.DD)")",A);
    CHSPOL(2,A);
    OUTPUT(61,"("/,"(" CHSPOL")",4B,"(":")",3(2B,+ZD.DD)")",A);
"END" PROGRAM;
```

		A[0]	A[1]	A[2]
INPUT	:	+1.00	+2.00	+3.00
POLCHS	:	+2.50	+2.00	+1.50
CHSPOL	:	+1.00	+2.00	+3.00

SUBSECTION: POLSHTCHS.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" POLSHTCHS(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"CODE" 31053;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[0:N];
ENTRY: THE COEFFICIENTS OF THE POWER SUM;
EXIT: THE COEFFICIENTS OF THE SHIFTED CHEBYSHEV SUM;

PROCEDURES USED: LINTFMPOL = CP31250,
POLCHS = CP31051.

EXAMPLE OF USE: SEE NEXT SUBSECTION.

SUBSECTION: SHTCHSPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" SHTCHSPOL(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"CODE" 31054;

THE MEANING OF THE FORMAL PARAMETERS IS:
N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL;
A: <ARRAY IDENTIFIER>;
"ARRAY" A[0:N];
ENTRY: THE COEFFICIENTS OF THE SHIFTED CHEBYSHEV SUM;
EXIT: THE COEFFICIENTS OF THE POWER SUM.

PROCEDURES USED: LINTFMPOL = CP31250,
                 CHSPOL   = CP31052.

EXAMPLE OF USE:

AS AN EXAMPLE WE TRANSFORMED
        THE POWER SUM: 1 + 2*X + 3*X**2
INTO
        ITS SHIFTED CHEBYSHEV SUM;
AS A CHECK WE TRANSFORMED THE LATTER REPRESENTATION
BACK INTO THE ORIGINAL POWER SUM.

```
"BEGIN"
    "PROCEDURE" POLSHTCHS(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE" 31053;
    "PROCEDURE" SHTCHSPOL(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE" 31054;
    "ARRAY" A[0:2];
    A[0]:=1; A[1]:=2; A[2]:=3;
    OUTPUT(61,"("/,16B,"("A[0]")",4B,"("A[1]")",4B,"("A[2]")"")");
    OUTPUT(61,"("/,"(" INPUT")",5B,"(":")",3(2B,+ZD.DD)")",A);
    POLSHTCHS(2,A);
    OUTPUT(61,"("/,"(" POLSHTCHS")",B,"(":")",3(2B,+ZD.DD)")",A);
    SHTCHSPOL(2,A);
    OUTPUT(61,"("/,"(" SHTCHSPOL")",B,"(":")",3(2B,+ZD.DD)")",A);
"END" PROGRAM;
```

		A[0]	A[1]	A[2]
INPUT	:	+1.00	+2.00	+3.00
POLSHTCHS	:	+3.13	+2.50	+0.38
SHTCHSPOL	:	+1.00	+2.00	+3.00

SUBSECTION: GRNNEW.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" GRNNEW(N,X,A);
"VALUE" N; "INTEGER" N; "ARRAY" X,A;
"CODE" 31055;
```

```
THE MEANING OF THE FORMAL PARAMETERS IS:
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
X:      <ARRAY IDENTIFIER>;
        "ARRAY" X[0:N-1];
        ENTRY: THE INTERPOLATION POINTS;
A:      <ARRAY IDENTIFIER>;
        ENTRY: THE COEFFICIENTS OF THE POWER SUM;
        EXIT:  THE COEFFICIENTS OF THE NEWTON SUM;
```

PROCEDURES USED: NONE.

EXAMPLE OF USE: SEE NEXT SUBSECTION.

SUBSECTION: NEWGRN.

CALLING SEQUENCE:

```
THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:
"PROCEDURE" NEWGRN(N,X,A);
"VALUE" N; "INTEGER" N; "ARRAY" X,A;
"CODE" 31050;

THE MEANING OF THE FORMAL PARAMETERS IS:
N:      <ARITHMETIC EXPRESSION>;
        ENTRY: THE DEGREE OF THE POLYNOMIAL;
X:      <ARRAY IDENTIFIER>;
        "ARRAY" X[0:N-1];
        ENTRY: THE INTERPOLATION POINTS;
A:      <ARRAY IDENTIFIER>;
        "ARRAY" A[0:N];
        ENTRY: THE COEFFICIENTS OF THE NEWTON SUM;
        EXIT:  THE COEFFICIENTS OF THE POWER SUM;
```

PROCEDURES USED : ELMVEC = CP34020.

EXAMPLE OF USE:

```
AS AN EXAMPLE WE TRANSFORMED
        THE POWER SUM:  1 + 2*X + 3*X**2
INTO
        ITS NEWTON SUM WITH
        INTERPOLATION POINTS: X[0]:=1.0, X[1]:=2.0;
AS A CHECK WE TRANSFORMED THE LATTER REPRESENTATION
BACK INTO THE ORIGINAL POWER SUM.
```

```
"BEGIN"
    "PROCEDURE" GRNNEW(N,X,A);
    "VALUE" N; "INTEGER" N; "ARRAY" X,A;
    "CODE" 31055;
    "PROCEDURE" NEWGRN(N,X,A);
    "VALUE" N; "INTEGER" N; "ARRAY" X,A;
    "CODE" 31050;
    "ARRAY" X[0:1], A[0:2];
    A[0]:=1; A[1]:=2; A[2]:=3;
    X[0]:=1; X[1]:=2;
    OUTPUT(61,"("/,16B,"("A[0]")",4B,"("A[1]")",4B,"("A[2]")"")");
    OUTPUT(61,"("/,"(" INPUT")",5B,"(":")",3(2B,+ZD.DD)")",A);
    GRNNEW(2,X,A);
    OUTPUT(61,"("/,"(" GRNNEW")",4B,"(":")",3(2B,+ZD.DD)")",A);
    NEWGRN(2,X,A);
    OUTPUT(61,"("/,"(" NEWGRN")",4B,"(":")",3(2B,+ZD.DD)")",A);
"END" PROGRAM;
```

		A[0]	A[1]	A[2]
INPUT	:	+1.00	+2.00	+3.00
GRNNEW	:	+6.00	+11.00	+3.00
NEWGRN	:	+1.00	+2.00	+3.00

SUBSECTION: LINTFMPOL.

CALLING SEQUENCE:

THE DECLARATION OF THE PROCEDURE IN THE CALLING PROGRAM READS:

```
"PROCEDURE" LINTFMPOL(P,Q,N,A); "VALUE" N,P,Q;
"INTEGER" N; "REAL" P,Q; "ARRAY" A;
"CODE" 31250;
```

THE MEANING OF THE FORMAL PARAMETERS IS:

N: <ARITHMETIC EXPRESSION>;
ENTRY: THE DEGREE OF THE POLYNOMIAL;

P,Q: <ARITHMETIC EXPRESSION>;
ENTRY: DEFINING THE LINEAR TRANSFORMATION OF THE INDEPENDENT VARIABLE X=P*Y+Q; (P=0 GIVES THE VALUE OF THE POLYNOMIAL WITH ARGUMENT Q.)

A: <ARRAY IDENTIFIER>;
"ARRAY" A[0:N];
ENTRY: THE COEFFICIENTS OF THE POWER SUM IN X;
EXIT: THE COEFFICIENTS OF THE POWER SUM IN Y;

PROCEDURES USED: NORDERPOL = CP31242.

EXAMPLE OF USE:

AS AN EXAMPLE WE TRANSFORMED
THE POWER SUM: 1 + 2*X + 3*X**2
INTO
ITS POWER SUM IN Y WITH
X = 2*Y + 3;
AS A CHECK WE TRANSFORMED THE LATTER REPRESENTATION
BACK INTO THE ORIGINAL POWER SUM.

```
"BEGIN"
    "PROCEDURE" LINTFMPOL(P,Q,N,A);
    "VALUE" N,P,Q; "INTEGER" N;
    "REAL" P,Q; "ARRAY" A;
    "CODE" 31250;
    "ARRAY" A[0:2];
    A[0]:=1; A[1]:=2; A[2]:=3;
    OUTPUT(61,"("/,16B,"("A[0]")",4B,"("A[1]")",4B,"("A[2]")"")");
    OUTPUT(61,"("/,"(" INPUT")",5B,"(":")",3(2B,+ZD.DD)")",A);
    LINTFMPOL(2,3,2,A);
    OUTPUT(61,"("/,"(" LINTFMPOL")",B,"(":")",3(2B,+ZD.DD)")",A);
    LINTFMPOL(1/2,-3/2,2,A);
    OUTPUT(61,"("/,"(" LINTFMPOL")",B,"(":")",3(2B,+ZD.DD)")",A);
"END" PROGRAM;
```

```
                 A[0]     A[1]     A[2]
INPUT      :    +1.00    +2.00    +3.00
LINTFMPOL  :   +34.00   +40.00   +12.00   (POWER SUM IN Y)
LINTFMPOL  :    +1.00    +2.00    +3.00   (POWER SUM IN X)
```

SOURCE TEXTS:

```
"CODE" 31051;
"PROCEDURE" POLCHS(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"IF" N>1 "THEN"
"BEGIN"
   "COMMENT"  SCALING;
   "INTEGER" K,L,TWOPOW;
   TWOPOW:=2;
   "FOR" K:=1 "STEP" 1 "UNTIL" N-2 "DO"
   "BEGIN"
      A[K]:=A[K]/TWOPOW; TWOPOW:=TWOPOW*2;
   "END";
   A[N-1]:=2*A[N-1]/TWOPOW;
   A[N]:=A[N]/TWOPOW;
   A[N-2]:=A[N-2]+A[N];
   "COMMENT" N<=2 READY;
   "FOR" K:=N-2 "STEP" -1 "UNTIL" 1 "DO"
   "BEGIN"
      A[K-1]:=A[K-1]+A[K+1]; A[K]:=A[K]*2 + A[K+2];
      "FOR" L:=K+1 "STEP" 1 "UNTIL" N-2 "DO"
      A[L]:=A[L]+A[L+2];
   "END";
"END" POLCHS;
         "EOP"
```

```
"CODE" 31052;
"PROCEDURE" CHSPOL(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"IF" N>1 "THEN"
"BEGIN"
    "INTEGER" K,L,TWOPOW;
    "FOR" K:=0 "STEP" 1 "UNTIL" N-2 "DO"
    "BEGIN"
        "FOR" L:=N-2 "STEP" -1 "UNTIL" K "DO"
        A[L]:=A[L]-A[L+2]; A[K+1]:=A[K+1]/2;
    "END";
    TWOPOW:=2;
    "FOR" K:=1 "STEP" 1 "UNTIL" N-2 "DO"
    "BEGIN"
        A[K]:=A[K]*TWOPOW; TWOPOW:=TWOPOW*2;
    "END";
    A[N-1]:=TWOPOW*A[N-1];
    A[N]:=TWOPOW*A[N];
"END" CHSPOL;
         "EOP"
```

```
"CODE" 31053;
"PROCEDURE" POLSHTCHS(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"BEGIN"
    "PROCEDURE" LINTFMPOL(P,Q,N,A);
    "VALUE" N,P,Q; "INTEGER" N; "REAL" P,Q; "ARRAY" A;
    "CODE"31250;
    "PROCEDURE" POLCHS(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE"31051;
    LINTFMPOL(.5,.5,N,A);
    POLCHS(N,A);
"END" POLSHTCHS;
        "EOP"
```

```
"CODE" 31054;
"PROCEDURE" SHTCHSPOL(N,A);
"VALUE" N; "INTEGER" N; "ARRAY" A;
"BEGIN"
    "PROCEDURE" CHSPOL(N,A);
    "VALUE" N; "INTEGER" N; "ARRAY" A;
    "CODE"31052;
    "PROCEDURE" LINTFMPOL(P,Q,N,A);
    "VALUE" N,P,Q; "INTEGER" N; "REAL" P,Q; "ARRAY" A;
    "CODE"31250;
    CHSPOL(N,A);
    LINTFMPOL(2,-1,N,A);
"END" SHTCHSPOL;
        "EOP"
```

```
"CODE" 31055;
"PROCEDURE" GRNNEW(N,X,A);
"VALUE" N; "INTEGER" N; "ARRAY" X,A;
"BEGIN"
    "PROCEDURE" ELMCEV(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;
    "INTEGER" K;
    "FOR" K:=N-1 "STEP" -1 "UNTIL" 0 "DO"
    ELMCEV(N-1,N-1-K,1,A,A,X[N-1-K]);
"END" GRNNEW;
        "EOP"
```

```
"CODE" 31050;
"PROCEDURE" NEWGRN(N,X,A);
"VALUE" N; "INTEGER" N; "ARRAY" X,A;
"BEGIN"
    "PROCEDURE" ELMVEC(L,U,SHIFT,A,B,X);
    "VALUE" L,U,SHIFT,X; "INTEGER" L,U,SHIFT;
    "REAL" X; "ARRAY" A,B;
    "CODE" 34020;
    "INTEGER" K;
    "FOR" K:=N-1 "STEP" -1 "UNTIL" 0 "DO"
    ELMVEC(K,N-1,1,A,A,-X[K]);
"END" NEWGRN;
        "EOP"


"CODE" 31250;
"PROCEDURE" LINTFMPOL(P,Q,N,A);
"VALUE" N,P,Q; "INTEGER" N; "REAL" P,Q; "ARRAY" A;
"BEGIN"
    "PROCEDURE" NORDERPOL(N,K,X,A);
    "VALUE" N,K,X; "INTEGER" N,K; "REAL" X; "ARRAY" A;
    "CODE" 31242;
    "INTEGER" K;
    "REAL" PPOWER;
    NORDERPOL(N,N,Q,A);
    PPOWER:=P;
    "FOR" K:=1 "STEP" 1 "UNTIL" N "DO"
    "BEGIN"
        A[K]:=PPOWER*A[K];
        PPOWER:=P*PPOWER;
    "END";
"END" LINTFMPOL;
        "EOP"
```

AUTHOR : C.G. VAN DER LAAN.

INSTITUTE : RIJKSUNIVERSITEIT GRONINGEN.

RECEIVED : 740131.

BRIEF DESCRIPTION :

INTCHS COMPUTES THE INDEFINITE INTEGRAL OF A GIVEN CHEBYSHEV SERIES.

KEYWORDS :

INDEFINITE INTEGRATION,
CHEBYSHEV SERIES.

CALLING SEQUENCE :

THE HEADING OF THE PROCEDURE READS :

```
"PROCEDURE"INTCHS(N,A,B);
"VALUE"N;"INTEGER"N;"ARRAY"A,B;
```

THE MEANING OF THE FORMAL PARAMETERS IS :

```
N   : <ARITHMETIC EXPRESSION>;
      ENTRY:
      THE DEGREE OF THE POLYNOMIAL REPRESENTED BY THE CHEBYSHEV
      SERIES;
A,B: <ARRAY IDENTIFIER>;
      "ARRAY" A[0:N],B[1:N+1];
      ENTRY:
      THE COEFFICIENTS OF THE CHEBYSHEV SERIES,A[0]+A[1]*T1(X)+...+
      +A[N]*TN(X),SHOULD BE GIVEN IN ARRAY A.
      EXIT:
      THE COEFFICIENTS OF THE INTEGRAL CHEBYSHEV SERIES,
      B[1]*T1(X)+...+B[N+1]*TN+1(X), ARE DELIVERED IN ARRAY B.
      (T1(X),...TN+1(X) DENOTE CHEBYSHEV POLYNOMIALS OF THE FIRST
      KIND,OF DEGREE 1,...N+1,RESPECTIVELY).
```

METHOD AND PERFORMANCE :

FOR A DESCRIPTION OF THE ALGORITHM SEE AMONG OTHERS :
CLENSHAW,1962,P.11,OR FOX AND PARKER,1968,P.59.

REFERENCES :

BROUCKE,R.(1973):
TEN SUBROUTINES FOR THE MANIPULATION OF CHEBYSHEV SERIES.
ALGORITHM 446.(FORTRAN).
COMM.ACM,VOL.16,1,P.254-256.

CLENSHAW,C.W.(1962):
CHEBYSHEV SERIES FOR MATHEMATICAL FUNCTIONS.
MATH.TAB.NAT.PHYS.LAB. 5,LONDON.
H.M. STATIONARY OFFICE.

FOX,L.&I.B.PARKER(1968):
CHEBYSHEV POLYNOMIALS IN NUMERICAL ANALYSIS.
OXFORD UNIVERSITY PRESS.

EXAMPLE OF USE :

AS A FORMAL TEST OF THE PROCEDURE INTCHS THE CHEBYSHEV SERIES :
1+1/2*T1(X)+1/5*T2(X)+1/10*T3(X) IS TRANSFORMED INTO ITS INTEGRAL.

```
"BEGIN""ARRAY"A[0:3],B[1:4];
   "PROCEDURE"INTCHS(N,A,B);"CODE"31248;
   A[0]:=1;A[1]:=.5;A[2]:=.2;A[3]:="-1;
   INTCHS(3,A,B);
   OUTPUT(61,"("/,4(BZ.4D)")",B[1],B[2],B[3],B[4]);
"END"
```

```
 .9000  .1000  .0333  .0125
```

SOURCE TEXT(S):

```
"CODE"31248;
"PROCEDURE"INTCHS(N,A,B);
"VALUE"N;"INTEGER"N;"ARRAY"A,B;
"COMMENT"
    INTCHS DELIVERS THE COEFFICIENTS B[I],I=1,...N+1, OF THE INTEGRAL
    CHEBYSHEV SERIES B[1]*T1(X)+...+B[N]*TN(X)+B[N+1]*TN+1(X).
    THESE COEFFICIENTS ARE OBTAINED BY MEANS OF INDEFINITE INTEGRATION
    OF THE CHEBYSHEV SERIES A[0]+A[1]*T1(X)+...+A[N]*TN(X).
    T1(X),...TN+1(X) DENOTE CHEBYSHEV POLYNOMIALS OF THE FIRST KIND,
    OF DEGREE 1,...N+1,RESPECTIVELY;
"IF"N=0"THEN"B[1]:=A[0]
"ELSE""IF"N=1"THEN""BEGIN"B[2]:=A[1]/4;B[1]:=A[0]"END"
      "ELSE""BEGIN""INTEGER"I;"REAL"H,L,DUM;
         H:=A[N];DUM:=A[N-1];B[N+1]:=H/((N+1)*2);B[N]:=DUM/(N*2);
         "FOR"I:=N-1"STEP"-1"UNTIL"2"DO"
         "BEGIN"L:=A[I-1];B[I]:=(L-H)/(2*I);H:=DUM;DUM:=L
         "END";B[1]:=A[0]-H/2
      "END"INTCHS;
        "EOP"
```

*********** M10512W //// END OF LIST ////
*********** M10512W //// END OF LIST ////

UITGAVEN IN DE SERIE MC SYLLABUS

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

MCS 1.1	F. GÖBEL & J. VAN DE LUNE, *Leergang Besliskunde, deel 1: Wiskundige basiskennis*, 1965. ISBN 90 6196 014 2.
MCS 1.2	J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 2: Kansberekening*, 1965. ISBN 90 6196 015 0.
MCS 1.3	J. HEMELRIJK & J. KRIENS, *Leergang Besliskunde, deel 3: Statistiek*, 1966. ISBN 90 6196 016 9.
MCS 1.4	G. DE LEVE & W. MOLENAAR, *Leergang Besliskunde, deel 4: Markovketens en wachttijden*, 1966. ISBN 90 6196 017 7.
MCS 1.5	J. KRIENS & G. DE LEVE, *Leergang Besliskunde, deel 5: Inleiding tot de mathematische besliskunde*, 1966. ISBN 90 6196 018 5.
MCS 1.6a	B. DORHOUT & J. KRIENS, *Leergang Besliskunde, deel 6a: Wiskundige programmering 1*, 1968. ISBN 90 6196 032 0.
MCS 1.6b	B. DORHOUT, J. KRIENS & J.TH. VAN LIESHOUT, *Leergang Besliskunde, deel 6b: Wiskundige programmering 2*, 1977. ISBN 90 6196 150 5.
MCS 1.7a	G. DE LEVE, *Leergang Besliskunde, deel 7a: Dynamische programmering 1*, 1968. ISBN 90 6196 033 9.
MCS 1.7b	G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7b: Dynamische programmering 2*, 1970. ISBN 90 6196 055 X.
MCS 1.7c	G. DE LEVE & H.C. TIJMS, *Leergang Besliskunde, deel 7c: Dynamische programmering 3*, 1971. ISBN 90 6196 066 5.
MCS 1.8	J. KRIENS, F. GÖBEL & W. MOLENAAR, *Leergang Besliskunde, deel 8: Minimaxmethode, netwerkplanning, simulatie*, 1968. ISBN 90 6196 034 7.
MCS 2.1	G.J.R. FÖRCH, P.J. VAN DER HOUWEN & R.P. VAN DE RIET, *Colloquium Stabiliteit van differentieschema's, deel 1*, 1967. ISBN 90 6196 023 1.
MCS 2.2	L. DEKKER, T.J. DEKKER, P.J. VAN DER HOUWEN & M.N. SPIJKER, *Colloquium Stabiliteit van differentieschema's, deel 2*, 1968. ISBN 90 6196 035 5.
MCS 3.1	H.A. LAUWERIER, *Randwaardeproblemen, deel 1*, 1967. ISBN 90 6196 024 X.
MCS 3.2	H.A. LAUWERIER, *Randwaardeproblemen, deel 2*, 1968. ISBN 90 6196 036 3.
MCS 3.3	H.A. LAUWERIER, *Randwaardeproblemen, deel 3*, 1968. ISBN 90 6196 043 6.
MCS 4	H.A. LAUWERIER, *Representaties van groepen*, 1968. ISBN 90 6196 037 1.

MCS 5 J.H. VAN LINT, J.J. SEIDEL & P.C. BAAYEN, *Colloquium Discrete wiskunde*, 1968.
ISBN 90 6196 044 4.

MCS 6 K.K. KOKSMA, *Cursus ALGOL 60*, 1969. ISBN 90 6196 045 2.

MCS 7.1 *Colloquium Moderne rekenmachines, deel 1*, 1969. ISBN 90 6196 046 0.

MCS 7.2 *Colloquium Moderne rekenmachines, deel 2*, 1969. ISBN 90 6196 047 9.

MCS 8 H. BAVINCK & J. GRASMAN, *Relaxatietrillingen*, 1969.
ISBN 90 6196 056 8.

MCS 9.1 T.M.T. COOLEN, G.J.R. FÖRCH, E.M. DE JAGER & H.G.J. PIJLS, *Elliptische differentiaalvergelijkingen, deel 1*, 1970.
ISBN 90 6196 048 7.

MCS 9.2 W.P. VAN DEN BRINK, T.M.T. COOLEN, B. DIJKHUIS, P.P.N. DE GROEN, P.J. VAN DER HOUWEN, E.M. DE JAGER, N.M. TEMME & R.J. DE VOGELAERE, *Colloquium Elliptische differentiaalvergelijkingen, deel 2*, 1970.
ISBN 90 6196 049 5.

MCS 10 J. FABIUS & W.R. VAN ZWET, *Grondbegrippen van de waarschijnlijkheidsrekening*, 1970. ISBN 90 6196 057 6.

MCS 11 H. BART, M.A. KAASHOEK, H.G.J. PIJLS, W.J. DE SCHIPPER & J. DE VRIES, *Colloquium Halfalgebra's en positieve operatoren*, 1971.
ISBN 90 6196 067 3.

MCS 12 T.J. DEKKER, *Numerieke algebra*, 1971. ISBN 90 6196 068 1.

MCS 13 F.E.J. KRUSEMAN ARETZ, *Programmeren voor rekenautomaten; De MC ALGOL 60 vertaler voor de EL X8*, 1971. ISBN 90 6196 069 X.

MCS 14 H. BAVINCK, W. GAUTSCHI & G.M. WILLEMS, *Colloquium Approximatietheorie*, 1971. ISBN 90 6196 070 3.

MCS 15.1 T.J. DEKKER, P.W. HEMKER & P.J. VAN DER HOUWEN, *Colloquium Stijve differentiaalvergelijkingen, deel 1*, 1972. ISBN 90 6196 078 9.

MCS 15.2 P.A. BEENTJES, K. DEKKER, H.C. HEMKER, S.P.N. VAN KAMPEN & G.M. WILLEMS, *Colloquium Stijve differentiaalvergelijkingen, deel 2*, 1973. ISBN 90 6196 079 7.

MCS 15.3 P.A. BEENTJES, K. DEKKER, P.W. HEMKER & M. VAN VELDHUIZEN, *Colloquium Stijve differentiaalvergelijkingen, deel 3*, 1975.
ISBN 90 6196 118 1.

MCS 16.1 L. GEURTS, *Cursus Programmeren, deel 1: De elementen van het programmeren*, 1973. ISBN 90 6196 080 0.

MCS 16.2 L. GEURTS, *Cursus Programmeren, deel 2: De programmeertaal ALGOL 60*, 1973. ISBN 90 6196 087 8.

MCS 17.1 P.S. STOBBE, *Lineaire algebra, deel 1*, 1974. ISBN 90 6196 090 8.

MCS 17.2 P.S. STOBBE, *Lineaire algebra, deel 2*, 1974. ISBN 90 6196 091 6.

MCS 17.3 N.M. TEMME, *Lineaire algebra, deel 3*, 1976. ISBN 90 6196 123 8.

MCS 18 F. VAN DER BLIJ, H. FREUDENTHAL, J.J. DE IONGH, J.J. SEIDEL & A. VAN WIJNGAARDEN, *Een kwart eeuw wiskunde 1946-1971, Syllabus van de Vakantiecursus 1971*, 1974. ISBN 90 6196 092 4.

MCS 19 A. HORDIJK, R. POTHARST & J.Th. RUNNENBURG, *Optimaal stoppen van Markovketens*, 1974. ISBN 90 6196 093 2.

MCS 20 T.M.T. COOLEN, P.W. HEMKER, P.J. VAN DER HOUWEN & E. SLAGT, *ALGOL 60 procedures voor begin- en randwaardeproblemen*, 1976. ISBN 90 6196 094 0.

MCS 21 J.W. DE BAKKER (red.), *Colloquium Programmacorrectheid*, 1975. ISBN 90 6196 103 3.

MCS 22 R. HELMERS, F.H. RUYMGAART, M.C.A. VAN ZUYLEN & J. OOSTERHOFF, *Asymptotische methoden in de toetsingstheorie; Toepassingen van naburigheid*, 1976. ISBN 90 6196 104 1.

MCS 23.1 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 1*, 1976. ISBN 90 6196 105 X.

MCS 23.2 J.W. DE ROEVER (red.), *Colloquium Onderwerpen uit de biomathematica, deel 2*, 1976. ISBN 90 6196 115 7.

MCS 24.1 P.J. VAN DER HOUWEN, *Numerieke integratie van differentiaalvergelijkingen, deel 1: Eenstapsmethoden*, 1974. ISBN 90 6196 106 8.

MCS 25 *Colloquium Structuur van programmeertalen*, 1976. ISBN 90 6196 116 5.

MCS 26.1 N.M. TEMME (ed.), *Nonlinear analysis, volume 1*, 1976. ISBN 90 6196 117 3.

MCS 26.2 N.M. TEMME (ed.), *Nonlinear analysis, volume 2*, 1976. ISBN 90 6196 121 1.

MCS 27 M. BAKKER, P.W. HEMKER, P.J. VAN DER HOUWEN, S.J. POLAK & M. VAN VELDHUIZEN, *Colloquium Discretiseringsmethoden*, 1976. ISBN 90 6196 124 6.

MCS 28 O. DIEKMANN, N.M. TEMME (EDS), *Nonlinear Diffusion Problems*, 1976. ISBN 90 6196 126 2.

MCS 29.1 J.C.P. BUS (red.), *Colloquium Numerieke programmatuur, deel 1A, deel 1B*, 1976. ISBN 90 6196 128 9.

MCS 29.2 H.J.J. TE RIELE (red.), *Colloquium Numerieke programmatuur, deel 2*, 1976. ISBN 90 6196 144 0

* MCS 30 P. GROENEBOOM, R. HELMERS, J. OOSTERHOFF & R. POTHARST, *Efficiency begrippen in de statistiek*, . ISBN 90 6196 149 1.

MCS 31 J.H. VAN LINT (red.), *Inleiding in de coderingstheorie*, 1976. ISBN 90 6196 136 X.

MCS 32 L. GEURTS (red.), *Colloquium Bedrijfssystemen*, 1976. ISBN 90 6196 137 8.

MCS 33 P.J. VAN DER HOUWEN, *Differentieschema's voor de berekening van waterstanden in zeeën en rivieren*, 1977. ISBN 90 6196 138 6.

MCS 34 J. HEMELRIJK, *Oriënterende cursus mathematische statistiek*, ISBN 90 6196 139 4.

MCS 35 P.J.W. TEN HAGEN (red.), *Colloquium Computer Graphics*, 1977. ISBN 90 6196 142 4.

MCS 36 J.M. AARTS, J. DE VRIES, *Colloquium Topologische Dynamische Systemen*, 1977. ISBN 90 6196 143 2.

MCS 37 J.C. van Vliet (red.), *Colloquium Capita Datastructuren*, 1978. ISBN 90 6196 159 9.

MCS 38.1 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 161 0.

MCS 38.2 T.H. KOORNWINDER (ED.), *Representations of locally compact groups with applications*, 1979. ISBN 90 6196 181 5.

MCS 39 O.J. VRIEZE & G.L. Wanrooij, *Colloquium Stochastische Spelen*, 1978. ISBN 90 6196 167 X.

MCS 40 J. VAN TIEL, *Convexe Analyse*, 1979. ISBN 90 6196 187 4.

MCS 41 H.J.J. TE RIELE (ED.), *Colloquium Numerical Treatment of Integral Equations*, 1979. ISBN 90 6196 189 0.

MCS 42 J.C. VAN VLIET (RED.), *Colloquium Capita Implementatie van Programmeertalen*, 1980. ISBN 90 6196 191 2.

MCS 43 A.M. COHEN & H.A. WILBRINK, *Eindige groepen (Een inleidende cursus)*, 1980. ISBN 90 6196 203 X

MCS 44 J.G. VERWER (ED.), *Numerical solution of partial differential equations*, 1980. ISBN 90 6196 205 6.

MCS 45 P. KLINT (red.), *Colloquium hogere programmeertalen en computerarchitectuur*, 1980. ISBN 90 6196 206 4.

MCS 46.1 P.M.G. APERS (RED.), *Colloquium Databankorganisatie*, 1981. ISBN 90 6196 212 9.

MCS 47.1 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part I: *General information and indices*, 1981. ISBN 90 6196 217 X.

MCS 47.2 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part II: *Elementary procedures, algebraic evaluation*, 1981, ISBN 90 6196 217 X.

MCS 47.3 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part III: *Linear algebra, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.4 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part IV: *Linear algebra, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.5 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part V: *Analytical evaluations, analytical problems, part I*, 1981. ISBN 90 6196 217 X.

MCS 47.6 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VI: *Analytical problems, part II*, 1981. ISBN 90 6196 217 X.

MCS 47.7 P.W. HEMKER (ED.), *NUMAL numerical procedures in ALGOL 60*, Part VII: *Special functions and constants, interpolation and approximation*, 1981. ISBN 90 6196 217 X.

De met een * gemerkte uitgaven moeten nog verschijnen.