APPENDIX

THE PROGRAMS

All the programs given on the succeeding pages have been successfully run in VMS environment using VAX FORTRAN 77 compiler and also in DOS using SOFTER FORTRAN 77 compiler. Systematic testing of the programs has been accomplished in part with the aid of programs which reconstruct polynomials from their known zeros and generate matrices with known spectra and certain known properties. Detailed description of the individual subroutines and the meanings of the parameters are described in the initial comments of the subroutines. The dimensions in the programs as given in the listings have been chosen small to facilitate execution in machines with small memory. However these can be changed simply by changing the DIMENSION statements. All the integer variables are declared to be INTEGER * 4, so that, if necessary, they may take up values which are numerically as large as possible for 32-bit machines. We have not physically joined our different programs with the programs for arithmetic operations on multi-precision integers to maintain the aesthetic beauty of coding different algorithms. But whenever occasion arises, we can do it immediately and thus we can get programs which will work satisfactorily in all possible situations. In the programs for multi-precision integers we have taken each piece to be of maximum 4 decimal digits. However larger numbers may be taken by systematically changing the PARAMETER statements and correspondingly
modifying the coding of the PRINT subroutines which have been developed using dynamic formatting. In this thesis we have not included the codes of the algorithms which can be obtained from the given listings by very simple routine modifications. For example, the subroutines for the computation of complex and surd eigenvalues of particular types of matrices with integer entries may be obtained by joining the subroutines for the computation of the characteristic polynomial of an integer matrix with the subroutines for the computation of complex and surd zeros of a particular type of integer polynomials. Also the subroutines presented in this thesis for the computation of eigenvectors corresponding to integer eigenvalues may be used for the computation of eigenvectors corresponding to rational eigenvalues by making necessary routine changes. Although we have used the integer mode in all programs, still many of the programs, such as reconstruction of polynomials from their zeros, computation of the characteristic polynomial of a matrix, etc. will work satisfactorily in real mode also, provided that we make the necessary changes in the type statements. The contents of the Appendix have been given in the end.
5.1 The program for addition of two non-negative m, n-piece integers

PARAMETER (ND = 20)
DIMENSION L (ND), M (ND), N (ND)

10 WRITE (*,'(" TYPE NUMBERS OF PIECES IN THE 2 INTEGERS")
1,1X,'(" TO BE ADDED IN FORMAT (2I2)")')
20 READ (4, '(2I2)') I, J
WRITE (6, '('" THE NUMBERS OF PIECES ARE", 214)') I, J
IF (I .EQ. 0 .AND. J .EQ. 0) STOP
IF (I .LE. 0 .OR. J .LE. 0 .OR. I .GT. ND .OR. J .GT. ND) THEN
   IF (I .LE. 0 .OR. J .LE. 0) THEN
      WRITE (6, '('" DATA ILLEGAL, PLEASE TYPE AGAIN")')
   ELSEIF (I .GT. ND .AND. J .GT. ND) THEN
      WRITE (6, '('" BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN")')
   ELSEIF (I .GT. ND) THEN
      WRITE (6, '('" FIRST INTEGER TOO LARGE, PLEASE TYPE AGAIN")')
   ELSE
      WRITE (6, '('" SECOND INTEGER TOO LARGE, PLEASE TYPE AGAIN")')
   ENDIF
GO TO 20
END IF
WRITE (*, '('" TYPE THE FIRST INTEGER IN FORMAT (2014)")')
READ (4, '(2014)') (L (K), K = I, 1, -1)
WRITE (*, '('" TYPE THE SECOND INTEGER IN FORMAT (2014)")')
READ (4, '(2014)') (M (K), K = J, 1, -1)
WRITE (6, '('" THE INTEGERS ARE AS FOLLOWS")')
CALL PRINT (L, I, 0)
CALL PRINT (M, J, 0)
IND = 0
CALL BIGADD (L, I, M, J, N, KK, IND)
IF (IND .EQ. 1) THEN
   WRITE (6, '('" THE SUM HAS MORE THAN 80 DIGITS")')
ELSE
   WRITE (6, '('" THE SUM IS")')
   CALL PRINT (N, KK, 0)
ENDIF
GO TO 10
END

C PURPOSE - TO FIND THE SUM OF TWO NON-NEGATIVE LARGE INTEGERS
C INPUT PARAMETERS
C L : LINEAR ARRAY, THE FIRST INTEGER
C LL : NUMBER OF PIECES IN THE FIRST INTEGER
C M : LINEAR ARRAY, THE SECOND INTEGER
C MM : NUMBER OF PIECES IN THE SECOND INTEGER
C OUTPUT PARAMETERS
C N : LINEAR ARRAY, THE SUM OF THE TWO INTEGERS
C \textbf{NN} : NUMBER OF PIECES IN THE SUM
C \textbf{IND} = 0 INITIALLY
C \textbf{= 1 WHEN THE SUM CONTAINS MORE THAN 80 DIGITS}

SUBROUTINE BIGADD (L, LL, M, MM, N, NN, IND)
PARAMETER (ND = 20, NNINE=9999)
DIMENSION L (LL), M (MM), N (ND)
ICARY = 0

C Compare the numbers of pieces in the 2 numbers
IF (LL .GT. MM) THEN
   LIMIT = MM
   C Store the leftmost pieces of L or M in N corresponding to
   C which there are no pieces in the other addend
   DO 11 II = MM + 1, LL
      N (II) = L (II)
   11 CONTINUE
   NN = LL
ELSEIF (LL .EQ. MM) THEN
   LIMIT = MM
   NN = LIMIT
ELSE
   LIMIT = LL
   DO 22 II = LL + 1, MM
      N (II) = M (II)
   22 CONTINUE
   NN = MM
ENDIF

C Add the numbers piece by piece, and if any piece exceeds NNINE
C take a CARRY to the next pair of pieces
DO 33 I = 1, LIMIT
   KTEST = NNINE - M (I) - ICARY
   IF (L (I) .GT. KTEST) THEN
      N (I) = L (I) - KTEST - 1
      ICARY = 1
   ELSE
      N (I) = NNINE - (KTEST - L (I))
      ICARY = 0
   ENDIF
33 CONTINUE
IF (ICARY .EQ. 1) THEN
   C Further calculation for ICARY=1
   IF (NN .NE. LIMIT) THEN
      DO 44 I = LIMIT + 1, NN
         IF (N (I) .EQ. NNINE) THEN
            N (I) = 0
         ELSE
            N (I) = N (I) + 1
         ENDIF
      44 CONTINUE
   ENDIF
RETURN
ENDIF
CONTINUE
ENDIF
NN = NN + 1
ENDIF

C IND=1 if the number of the pieces exceeds the specified limit
IF (NN .GT. ND) THEN
   IND = 1
ELSE
   N (NN) = 1
ENDIF
ENDIF
RETURN
END

C PURPOSE - GIVEN AN N-PIECE INTEGER THE SUBROUTINE
C PRINTS THE INTEGER, IN ORDINARY POSITIONAL
C NOTATION IN WHICH INTEGERS ARE EXPRESSED
C INPUT PARAMETERS
C NUMB : LINEAR ARRAY, THE INTEGER
C J : NUMBER OF PIECES IN THE INTEGER

SUBROUTINE PRINT (NUMB, J, INDEX)
INTEGER PI
PARAMETER (ND = 20, P1 = 185)
DIMENSION NUMB (ND)
CHARACTER*1 FORM (P1)
DATA (FORM(I), I = 1, 4) /'V', '1', 'X', ',','/
IF (INDEX .EQ. 1) THEN
   FORM (5) = '1'
   FORM (6) = 'H'
   FORM (7) = '-'
   FORM (8) = ','
   FORM (9) = 'I'
   NP = 10
ELSE
   FORM (5) = 'I'
   NP = 6
ENDIF
IF (NUMB (J) .LT. 10) THEN
   FORM (NP) = '1'
ELSEIF (NUMB (J) .LT. 100) THEN
   FORM (NP) = '2'
ELSEIF (NUMB (J) .LT. 1000) THEN
   FORM (NP) = '3'
ELSE
   FORM (NP) = '4'

ENDIF
ENDIF
IF (J .GT. 1) THEN
  DO 44 K = 2, J
    NP = NP + 1
    FORM (NP) = ',',
    IF (NUMB (J + 1 - K) .GE. 1000) THEN
      FORM (NP + 2) = '4'
    ELSE
      NP = NP + 2
      FORM (NP) = 'H'
      IF (NUMB (J + 1 - K) .LT. 10) THEN
        FORM (NP - 1) = '3'
        DO 11 I = NP + 1, NP + 3
          FORM (I) = '0'
      ELSEIF (NUMB (J + 1 - K) .LT. 100) THEN
        FORM (NP - 1) = '2'
        DO 22 I = NP + 1, NP + 2
          FORM (I) = '0'
      ELSE
        FORM (NP - 1) = '1'
        DO 33 I = NP + 1, NP + 1
          FORM (I) = '0'
    IF (NUMB (J + 1 - K) .GE. 1000) THEN
      FORM (NP + 2) = '4'
      ELSEIF (NUMB (J + 1 - K) .LT. 100) THEN
        FORM (NP - 1) = '2'
        DO 22 I = NP + 1, NP + 2
          FORM (I) = '0'
    ELSE
      FORM (NP - 1) = '1'
      DO 33 I = NP + 1, NP + 1
        FORM (I) = '0'
  11 CONTINUE
  NP = NP + 4
  FORM (NP + 2) = '1'
ELSEIF (NUMB (J + 1 - K) .LT. 100) THEN
  FORM (NP - 1) = '2'
  DO 22 I = NP + 1, NP + 2
    FORM (I) = '0'
ELSE
  FORM (NP - 1) = '1'
  DO 33 I = NP + 1, NP + 1
    FORM (I) = '0'
ENDIF
ENDIF
FORM (NP) = ',',
ENDIF
FORM (NP + 1) = 'I'
NP = NP + 2
44 CONTINUE
ENDIF
FORM (NP + 1) = ',',
DO 55 I = NP + 2, P1 - 1
  FORM (I) = ','
55 CONTINUE
FORM (P1) = ')'
WRITE (6, FORM) (NUMB (J + 1 - K), K = 1, J)
RETURN
END
THE NUMBERS OF PIECES ARE 1 1
THE INTEGERS ARE AS FOLLOWS
654
208
THE SUM IS
862
THE NUMBERS OF PIECES ARE 2 3
THE INTEGERS ARE AS FOLLOWS
4264827
27694278406
THE SUM IS
27698543233
THE NUMBERS OF PIECES ARE 10 8
THE INTEGERS ARE AS FOLLOWS
896578434526754368850000876380430498290
8965784345188994612215464925064
THE SUM IS
896578443492538714038995488595895423354
THE NUMBERS OF PIECES ARE 10 10
THE INTEGERS ARE AS FOLLOWS
8010000040000050000020006000007000060
40000200002000000400010000400000080000
THE SUM IS
12010200042000050400030006400007080060
THE NUMBERS OF PIECES ARE 5 15
THE INTEGERS ARE AS FOLLOWS
68489645327896543289658380078808707696283610977901504950632
THE SUM IS
68489645327896543289658380078808707696286268885957444221157
THE NUMBERS OF PIECES ARE 15 15
THE INTEGERS ARE AS FOLLOWS
896578434526754368852657890657939270525000000000000000000
87076962836109779015049563268489645327896543289658380070088
THE SUM IS
1767348062887852159003153523342383572380396623289658380070088
THE NUMBERS OF PIECES ARE 21 20
FIRST INTEGER TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 20 21
SECOND INTEGER TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 21 21
BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 2 0
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 0 4
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE -1 2
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 5 -6
5.2 The program for subtracting a non-negative m-piece integer from a non-negative n-piece integer

PARAMETER (ND = 20)
DIMENSION L (ND), M (ND), N (ND)

10 WRITE (*, '('

1, 'X', 'AND THE SUBTRAHEND IN FORMAT (2I2)''

20 READ (4, '(2I2)') I, J

WRITE (6, '(" THE NUMBERS OF PIECES ARE",214)') I, J
IF (I .EQ. 0 .AND. J .EQ. 0) STOP
IF (I .LE. 0 .OR. J .LE. 0 .OR. I .GT. ND .OR. J .GT. ND) THEN
  IF (I .LE. 0 .OR. J .LE. 0) THEN
    WRITE (6, '(" DATA ILLEGAL, PLEASE TYPE AGAIN")')
  ELSEIF (I .GT. ND .AND. J .GT. ND) THEN
    WRITE (6, '(" BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN")')
  ELSEIF (I .GT. ND) THEN
    WRITE (6, '(" MINUEND TOO LARGE, PLEASE TYPE AGAIN")')
  ELSE
    WRITE (6, '(" SUBTRAHEND TOO LARGE, PLEASE TYPE AGAIN")')
  ENDIF
ENDIF

WRITE (*, '(

1, 'X', 'TYPE THE MINUEND IN FORMAT (2014)"

READ (4, '(2014)') (L (K), K = I, 1, -1)
WRITE (*, '(

1, 'X', 'TYPE THE SUBTRAHEND IN FORMAT (2014)"

READ (4, '(2014)') (M (K), K = J, 1, -1)

WRITE (6, '(" THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS")')
CALL PRINT (L, I, 0)
CALL PRINT (M, J, 0)
CALL BIGSUB (L, I, M, J, N, NN, INDEX)
WRITE (6, '(" THE DIFFERENCE IS")')
CALL PRINT (N, NN, INDEX)
GO TO 10
END

C PURPOSE - THIS SUBROUTINE SUBTRACTS ONE LARGE NON-NEGATIVE
C INTEGER FROM ANOTHER
C INPUT PARAMETERS
C L : LINEAR ARRAY, THE MINUEND
C LL : NUMBER OF PIECES IN THE MINUEND
C M : LINEAR ARRAY, THE SUBTRAHEND
C MM : NUMBER OF PIECES IN THE SUBTRAHEND
C OUTPUT PARAMETERS
C N : LINEAR ARRAY, THE REMAINDER
SUBROUTINE BIGSUB (L, LL, M, MM, N, NN, INDEX)
PARAMETER (ND = 20, NNINE=9999, NTHOUS=1000)
DIMENSION L (LL), N1 (ND), M (MM), N (ND)
IBOROW = 0

C Compare the numbers of pieces in the 2 numbers
IF (LL .GT. MM) THEN
  INDEX = 0
ELSEIF (LL .LT. MM) THEN
  INDEX = 1
ELSE
  C For equal number of pieces compare pieces starting from left
  DO 11 I = MM, 1, -1
    IF (L (I) .NE. M (I)) THEN
      IF (L (I) .GT. M (I)) THEN
        INDEX = 0
      ELSE
        INDEX = 1
      END IF
      GO TO 20
    ENDIF
  11 CONTINUE
  INDEX = 0
  NN = 1
  N (1) = 0
  RETURN
ENDIF

C For SUBTRAHEND > MINUEND, put INDEX=1, otherwise INDEX=0
20 IF (INDEX .EQ. 0) THEN
  DO 33 I = 1, MM
    N1 (I) = M (I)
  33 CONTINUE
  DO 44 I = 1, LL
    N (I) = L (I)
  44 CONTINUE
  NN = LL
  KK = MM
ELSE
  DO 55 I = 1, MM
    N (I) = M (I)
  55 CONTINUE
  DO 66 I = 1, LL
    N1 (I) = L (I)
  66 CONTINUE
KK = LL
NN = MM
ENDIF
C Subtract the corresponding pieces, taking care of borrowed figures from the previous piece
DO 77 I = 1, KK
   IDIF = N (I) - N1 (I)
   IF (IDIF .GT. 0) THEN
      N (I) = IDIF - IBOROW
      IBOROW = 0
   ELSEIF (IDIF .EQ. 0 .AND. IBOROW .EQ. 0) THEN
      N (I) = 0
   ELSE
      N (I) = IDIF + 1 - IBOROW + NNINE
      IBOROW = 1
   END IF
77 CONTINUE
IF (KK .NE. NN) THEN
   IF (IBOROW .EQ. 0) RETURN
   IF (KK + 1 .LT. NN) THEN
      DO 88 I = KK + 1, NN - 1
         IF (N (I) .NE. 0) THEN
            N (I) = N (I) - 1
            RETURN
         END IF
      N(I) = NNINE
   88 CONTINUE
   N (NN) = N (NN) - 1
ENDIF
IF (NN .GT. 1) THEN
   MN1 = NN - 1
ENDIF
C Test whether leftmost pieces are zero
DO 99 I = 1, MN1
   IF (N (NN) .NE. 0) RETURN
   NN = NN - 1
99 CONTINUE
ENDIF
RETURN
END

THE NUMBERS OF PIECES ARE  1  1
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
6549
8630
THE DIFFERENCE IS
-2081
THE NUMBERS OF PIECES ARE 2 3
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
42648278
276985432345
THE DIFFERENCE IS
-276942784067
THE NUMBERS OF PIECES ARE 10 8
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
896578434526754368850000000763804304982901
78549076284543838739340230320
THE DIFFERENCE IS
8965784345188994612215464925064964752581
THE NUMBERS OF PIECES ARE 10 10
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
800965784345263456000008763804304987669829
3470085490762896389454383898308277393402
THE DIFFERENCE IS
453957235268973817054649248212221376427
THE NUMBERS OF PIECES ARE 5 15
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
26578906579392705268
6848964532789654328965838007880870769654940004369543200890
THE DIFFERENCE IS
-6848964532789654328965838007880870769628361097790150495632
THE NUMBERS OF PIECES ARE 15 15
THE MINUEND AND SUBTRAHEND ARE AS FOLLOWS
9000000000002000000000000000005000000000002000000000000000
9000000000002000000000000000400000000000000010000000000000
THE DIFFERENCE IS
10000000000000001000000000000000
THE NUMBERS OF PIECES ARE 21 20
MINUEND TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 20 21
SUBTRAHEND TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 21 21
BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 2 0
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 0 4
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE -1 2
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 5 -6
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 0 0
5.3 The program for multiplication of two non-negative m, n-piece integers

PARAMETER (ND = 20)
DIMENSION L (ND), M (ND), N (ND)

WRITE (*, '('' TYPE NUMBERS OF PIECES OF THE MULTIPLICAND'''
1,1X, 'AND THE MULTIPLIER IN FORMAT (2I2)'')')

READ (4, '(2I2)') I, J
WRITE (6, ('' THE NUMBERS OF PIECES ARE'', 2I4'') I, J
IF (I .EQ. 0 .AND. J .EQ. 0) STOP
IF (I .LE. 0 .OR. J .LE. 0 .OR. I .GT. ND .OR. J .GT. ND) THEN
  IF (I .LE. 0 .OR. J .LE. 0) THEN
    WRITE (6, ('' DATA ILLEGAL, PLEASE TYPE AGAIN''))
  ELSEIF (I .GT. ND .AND. J .GT. ND) THEN
    WRITE (6, ('' BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN''))
  ELSEIF (I .LE. ND) THEN
    WRITE (6, ('' MULTIPLICAND TOO LARGE, PLEASE TYPE AGAIN''))
  ELSE
    WRITE (6, ('' MULTIPLIER TOO LARGE, PLEASE TYPE AGAIN''))
  ENDIF
ENDIF
GO TO 20
ENDIF
WRITE (*, '('' TYPE THE MULTIPLICAND IN FORMAT (20I4)'')')
READ (4, '20I4') (L (K), K = I, 1, -1)
WRITE (*, 'TYPE THE MULTIPLIER IN FORMAT (20I4)')
READ (4, '20I4') (M (K), K = J, 1, -1)
IND = 0
WRITE (6, ('' MULTIPLICAND AND MULTIPLIER ARE AS FOLLOWS''))
CALL PRINT (L, I, 0)
CALL PRINT (M, J, 0)
CALL BIGMUL (L, I, M, J, N, NN, IND)
IF (IND .EQ. 0) THEN
  WRITE (6, ('' THE PRODUCT IS''))
  CALL PRINT (N, NN, 0)
ELSE
  WRITE (6, ('' THE PRODUCT HAS MORE THAN 80 DIGITS''))
ENDIF
GO TO 10
END

C PURPOSE - MULTIPLICATION OF TWO LARGE INTEGERS
C INPUT PARAMETERS
C A : LINEAR ARRAY, THE MULTIPLICAND
C I : NUMBER OF PIECES IN THE MULTIPLICAND
C B : LINEAR ARRAY, THE MULTIPLIER
C J : NUMBER OF PIECES OF IN THE MULTIPLIER
C OUTPUT PARAMETERS
SUBROUTINE BIGMUL (A, I, B, J, C, M, IND)
INTEGER PI
PARAMETER (ND = 20, P1 = 9, NDIG=4)
INTEGER A (I), IPROD (ND), B (J), C (ND), D (80),
1INP (P1), ISP (P1, ND), CI (ND)
IS = - 1
C (1) = 0
M = 1
DO 11 K = 2, 9
   INP (K) = 0
11 CONTINUE
CALL INTEG (B, J, D, IDN)
DO 66 MM = 1, IDN
   IS = IS + 1
   C Special treatment for the digits 0 and 1
   IF (D (MM) .NE. 0) THEN
      IF (D (MM) .GT. 1) THEN
         IF (INP (D (MM)) .EQ. 0) THEN
            CALL MULT (A, I, D (MM), IPROD, N1, IND)
            IF (IND .EQ. 1) RETURN
         ENDIF
      ELSE
         C For the digit 1 the multiplicand is the product
         DO 44 K = 1, I
            IPROD (K) = A (K)
         44 CONTINUE
         N1 = I
      ENDIF
   ELSE
      C Multiply by a particular digit only once
      IF (INP (D (MM)) .EQ. 0) THEN
         CALL MULT (A, I, D (MM), IPROD, N1, IND)
         IF (IND .EQ. 1) RETURN
      ENDIF
      C Make a table of 8 multipliers of the multiplicand
      DO 22 K = 1, N1
         ISP (D (MM), K) = IPROD (K)
      22 CONTINUE
      INP (D (MM)) = N1
   ENDIF
   C If the multiplicand has already been multiplied by a
   C particular digit do not multiply it again
   ELSE
      N1 = INP (D (MM))
      DO 33 K = 1, N1
         IPROD (K) = ISP (D (MM), K)
      33 CONTINUE
   ENDIF
ELSE
   C For the digit 1 the multiplicand is the product
   DO 44 K = 1, I
      IPROD (K) = A (K)
   44 CONTINUE
   N1 = I
ENDIF
C Decide whether shifting is required
IF (N1 .NE. 1 .OR. IPROD (1) .NE. 0) THEN
  CALL SHIFT (IPROD, N1, IS, IND)
  IF (IND .EQ. 1) RETURN
  CALL BIGADD (C, M, IPROD, N1, C1, M1, IND)
  IF (IND .EQ. 1) RETURN
  M = M1
C Transfer the product into the original array
DO 55 MK = 1, M
  C (MK) = CI (MK)
55 CONTINUE
END IF
END IF
66 CONTINUE
RETURN
END

C PURPOSE - TO MULTIPLY A NUMBER BY A SINGLE DIGIT
C INPUT PARAMETERS
C J : LINEAR ARRAY, THE MULTIPLICAND
C N : NUMBER OF PIECES IN THE MULTIPLICAND
C K : THE DIGIT WITH WHICH IT IS TO BE MULTIPLIED
C OUTPUT PARAMETERS
C IPROD : LINEAR ARRAY, PRODUCT OF MULTIPLICAND WITH
C A SINGLE DIGIT
C N1 : NUMBER OF PIECES IN THE PRODUCT

SUBROUTINE MULT (J, N, K, IPROD, N1, IND)
  PARAMETER (ND = 20, NNINE=9999, NTHOUS=1000)
  DIMENSION J (ND), IPROD (ND)
  IPROD (1) = 0
C Special treatment for the digits 0 and 1
  IF (K .EQ. 0) THEN
    N1 = 1
  ELSEIF (K .EQ. 1) THEN
    DO 10 I = 1, N
      IPROD (I) = J (I)
10    CONTINUE
    N1 = N
  ELSE
    C Multiply by the digit
    DO 22 I = 1, N
      IQ = J (I) / NTHOUS
      M = (J (I) - NTHOUS * IQ) * K + IPROD (I)
      L = IQ * K
      IQ = L / 10
      ITEST = M - (NNINE - NTHOUS * (L - 10 * IQ))
      IF (ITEST .GT. 0) THEN
        IPROD (I) = ITEST - 1
      END IF
22  CONTINUE
  END IF
END
IQ = IQ + 1
ELSE
  IPRD(I) = ITEST + NNINE
ENDIF
IF (I .NE. N) THEN
  IPRD(I + 1) = IQ
ENDIF
CONTINUE
N1 = N
IF (IQ .NE. 0) THEN
  N1 = N1 + 1
  IND = 1 when number of pieces crosses specified limit
  IF (N1 .GT. ND) THEN
    IND = 1
  ELSE
    IPRD(N1) = IQ
  ENDIF
ENDIF
RETURN
END

C PURPOSE - THIS ROUTINE SHIFTS AN N-PIECE POSITIVE INTEGER BY A FIXED NUMBER OF POSITIONS TO THE LEFT
C INPUT PARAMETERS
C IPR : LINEAR ARRAY, THE INTEGER BEFORE SHIFTING
C IAS : NUMBER OF PIECES IN THE INTEGER BEFORE SHIFTING
C MP : THE NUMBER POSITIONS BY WHICH SHIFTING IS REQUIRED
C OUTPUT PARAMETERS
C IPR : LINEAR ARRAY, THE INTEGER AFTER SHIFTING
C IAS : NUMBER OF PIECES IN THE INTEGER AFTER SHIFTING

SUBROUTINE SHIFT (IPR, IAS, MP, IND)
PARAMETER (ND = 20, NDIG=4)
DIMENSION IPR (ND)
C Check whether shift is required
IF (MP .EQ. 0) RETURN
IP = MP
INTR = 0
C Test whether shift is possible within the specified limit
IF (IP .GE. NDIG) THEN
  ID = IP / NDIG
  IF (IAS + ID .GT. ND) THEN
    IND = 1
    RETURN
  ENDIF
ENDIF
C Shift each piece by the required number of pieces
  DO 11 I = IAS, 1, -1
IPR (I + ID) = IPR (I)

CONTINUE
IAS = IAS + ID
DO 22 I = 1, ID
    IPR (I) = 0
22 CONTINUE

C Determine the number of positions by which the digits are to be shifted
IP = IP - NDIG * ID
IF (IP .EQ. 0) RETURN
KK = ID + 1
ELSE
    KK = 1
ENDIF
K = 10 ** (NDIG- IP)

C Shift required number of digits
DO 44 I = KK, IAS
    IHIN = IPR (I) / K
    IPR (I) = (IPR (I) - IHIN * K) * 10 ** IP + INTR
    INTR = IHIN
44 CONTINUE

C The number of pieces are to be increased
IF (INTR .NE. 0) THEN
    IAS = IAS + 1
    IPR (IAS) = INTR
ENDIF
RETURN
END

C PURPOSE - THE SUBROUTINE INTEG DIGITISES A LARGE INTEGER INTO ITS DIGITS
C INPUT PARAMETERS
C B : LINEAR ARRAY, THE INTEGER
C J : NUMBER OF PIECES IN THE GIVEN INTEGER
C OUTPUT PARAMETERS
C D : LINEAR ARRAY, THE DIGITS
C IDN: THE NUMBER OF DIGITS

SUBROUTINE INTEG (B, J, D, IDN)
PARAMETER (NDIG=4)
INTEGER B (J), D (80)

C Set the initial values
IDN = 0
NG = NDIG
DO 44 II = 1, J
    IDIV = B (II)
    IF (II .EQ. J) THEN
        C Find the number of digits in the leftmost piece

DO 11 JJ = 1, NDIG - 1
   IF (IDIV .LT. 10 ** JJ) THEN
      NG = JJ
      GO TO 20
   END IF
11 CONTINUE

END IF
C Find the digits of each piece
20 DO 33 KK = 1, NG
   IDN = IDN + 1
   D (IDN) = MOD (IDIV, 10)
   IF (KK .NE. NG) THEN
      IDIV = IDIV / 10
   END IF
33 CONTINUE

44 CONTINUE
RETURN
END

THE NUMBERS OF PIECES ARE  1  1
MULTIPICAND AND MULTIPLIER ARE AS FOLLOWS
6549
8630
THE PRODUCT IS
56517870

THE NUMBERS OF PIECES ARE  2  1
MULTIPICAND AND MULTIPLIER ARE AS FOLLOWS
67894521
2000
THE PRODUCT IS
135789042000

THE NUMBERS OF PIECES ARE  2  3
MULTIPICAND AND MULTIPLIER ARE AS FOLLOWS
42648278
276985432345
THE PRODUCT IS
11812951720599751910

THE NUMBERS OF PIECES ARE  5  5
MULTIPICAND AND MULTIPLIER ARE AS FOLLOWS
690804000012000044
78900280000560078
THE PRODUCT IS
54504629026453710954331040960643432

THE NUMBERS OF PIECES ARE 10 8
MULTIPICAND AND MULTIPLIER ARE AS FOLLOWS
8965784345267543688500008763804304982901
78549076284543838739340230320
5.4 Program for division of an m-piece non-negative integer by an n-piece positive integer

PARAMETER (ND = 20, NDIG=4, NTHOUS=1000)
INTEGER A (ND), A1 (ND), B (ND), IGIT (80), IQT (ND)

WRITE (*, '(TYPE THE NUMBERS OF PIECES OF DIVIDEND'
1,1X,' DIVISOR IN FORMAT (212)')')

READ (4, '(2I2)') I, J
WRITE (6, '(TYPE THE NUMBERS OF PIECES ARE', 2I4)') I, J
IF (I .EQ. 0 .AND. J .EQ. 0) STOP
IF (I .LE. 0 .OR. J .LE. 0 .OR. I .GT. ND .OR. J .GT. ND) THEN
   IF (I .LE. 0 .OR. J .LE. 0) THEN
      WRITE (6, '(DATA ILLEGAL, PLEASE TYPE AGAIN')
   ELSEIF (I .GT. ND .AND. J .GT. ND) THEN
      WRITE (6,'(BOTH NUMBERS TOO LARGE, PLEASE TYPE AGAIN')
   ELSEIF (I .GT. ND) THEN
      WRITE (6, '(DIVIDEND TOO LARGE, PLEASE TYPE AGAIN')
   ELSE
      WRITE (6, '(DIVISOR TOO LARGE, PLEASE TYPE AGAIN')
   END IF
END IF
GO TO 20
ENDIF
WRITE (*, '(TYPE THE DIVIDEND IN FORMAT (2014)')

THE PRODUCT IS
704254078487189224432816595176755510591840584986268239503549001758320
THE NUMBERS OF PIECES ARE 15 15
MULTIFICAND AND MULTIPLIER ARE AS FOLLOWS
9000000000002200220200000000000000000000000000000000200000000000000
9000000000002200220200000000000000000000000000000000100000000000000
THE PRODUCT HAS MORE THAN 80 DIGITS
THE NUMBERS OF PIECES ARE 21 20
MULTIPLIER TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 20 21
MULTIFICAND TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 21 21
BOTH INTEGERS TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 2 0
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 0 4
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE -1 2
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 5 -6
DATA ILLEGAL, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 0 0
READ (4, '(2014)') (A (K), K = I, 1, -1)
WRITE (*, '(" TYPE THE DIVISOR IN FORMAT (2014)")')
READ (4, '(2014)') (B (LL), LL = J, 1, -1)
IF (J .EQ. 1 .AND. B (1) .EQ. 0) THEN
   WRITE (6, '(" DIVISION BY ZERO IS NOT DEFINED")')
   GO TO 10
ENDIF
CALL BIGDIV (A, I, B, J, IQT, M, A1, MT)
WRITE (6, '(" THE DIVIDEND AND DIVISOR ARE AS FOLLOWS")')
CALL PRINT (A, I, 0)
CALL PRINT (B, J, 0)
WRITE (6, '(" THE QUOTIENT IS")')
CALL PRINT (IQT, M, 0)
WRITE (6, '(" THE REMAINDER IS")')
CALL PRINT (A1, MT, 0)
GO TO 10
END

C PURPOSE - PROGRAM FOR DIVISION OF A M-PIECE NON-NEGATIVE
C INTEGER BY AN N-PIECE NON-NEGATIVE INTEGER
C INPUT PARAMETERS
C A : LINEAR ARRAY, THE DIVIDEND
C I : THE NUMBER PIECES OF THE DIVIDEND
C B : LINEAR ARRAY, THE DIVISOR
C J : THE NUMBER OF PIECES IN THE DIVISOR
C OUTPUT PARAMETERS
C IQT : LINEAR ARRAY, THE QUOTIENT
C M : NUMBER OF PIECES IN THE QUOTIENT
C A1 : LINEAR ARRAY, THE REMAINDER
C MT : NUMBER OF PIECES IN THE REMAINDER
SUBROUTINE BIGDIV (A, I, B, J, IQT, M, A1, MT)
PARAMETER (ND = 20, NDIG=4, NTHOUS=1000)
INTEGER A (I), Al (I), B (J), IGIT (80), IQT (ND)
IND = 0
M = 1
IQT (1) = 0
IF (I .LT. J) THEN
   MT = I
   DO 33 II = I, 1, -1
      A1 (II) = A (II)
33   CONTINUE
C The remainder, when I < J
ELSE
   IF (I .GT. J) THEN
      LM = I - J
      C Digitise the rightmost LM pieces of the dividend
      CALL INTEG (A, LM, IGIT, NDI)
      IC = NDIG * LM
IF (NDI .NE. IC) THEN
  NDI = NDI + 1
  DO 44 II= NDI, IC
    IGIT (II) = 0
  CONTINUE
  NDI = IC
ENDIF
ELSE
  NDI = 0
ENDIF
MT = J
C
Form partial dividend with left most J pieces of dividend
IM = I - J
DO 55 II = J, 1, -1
  A1 (II) = A (II + IM)
55 CONTINUE
C
Find the number of digits in the leftmost pieces of
the dividend and the divisor
IDEND = A1 (J)
ISOR = B (J)
DO 66 II = 1, NDIG - 1
  IF (IDEND .LT. 10 ** II) GO TO 70
66 CONTINUE
II = NDIG
70 DO 88 J1 = 1, NDIG - 1
  IF (ISOR .LT. 10 ** J1) GO TO 90
88 CONTINUE
J1 = NDIG
90 DO 111 II = J, 1, -1
  IF (A1 (II) .GT. B (II)) GO TO 120
  IF (A1 (II) .LT. B (II)) GO TO 150
111 CONTINUE
120 IF (J1 .LT. NDIG .AND. J .NE. 1) THEN
  CALL PARDEF (A1, B, J, I1, J1, IGIT, NDI, ISOR, IDEND)
ENDIF
IF (IDEND .LT. ISOR) THEN
  NISOR = ISOR / 10
ELSE
  NISOR = ISOR
ENDIF
IW = IDEND / NISOR
C
Find the actual digits of the quotient
CALL EXACDQ (A1, MT, B, J, IW, IND)
C
Form the quotient by appending digit(s) to the right
CALL APPEND (IQT, M, IW, IND)
Form the partial dividend by appending digit(s) to the right

```fortran
IF (NDI .GT. 0) THEN
   CALL APPEND (A1, MT, IGIT (NDI), IND)
   NDI = NDI - 1
END IF

IF (MT .GT. J) THEN
   IDEND = A1 (MT) * NTHOUS + A1 (MT - 1) / 10
   NISOR = ISOR / 10
   GO TO 130
ELSE
   IF (IQT (M) .EQ. 0) THEN
      II = II + 1
      GO TO 90
   ENDIF
   IF (MT .LT. J) THEN
      IW = 0
   ELSE
      DO 166 II = J, 1, -1
      IF (A1 (II) .NE. B (II)) THEN
         IF (A1 (II) .LT. B (II)) THEN
            IW = 0
            GO TO 140
         ELSE
            IDEND = A1 (J)
            NISOR = ISOR
            IF (J1 .LT. NDIG .AND. J .NE. 1) THEN
               C Initialize IDEND with right most 4 digits of
               C the partial dividend and ISOR with right mos:
               C 3 digits of the divisor
               C
               IF (IDEND .GE. 10 ** J1) THEN
                  IDEND = A1 (J) * 10 ** (NDIG - (J1+1))
                  + (A1 (J - 1)/10 ** (J1 + 1))
                  NISOR = ISOR/10
               ELSE
                  IDEND = A1 (J) * 10 ** (NDIG - J1) +
                  (A1 (J - 1)/10 ** J1)
               ENDIF
            ENDIF
         ENDIF
      ENDIF
   ENDIF
166 CONTINUE
   MT = 1
   A1 (1) = 0
   IW = 1
ENDIF
ENDIF
GO TO 130
```
C PURPOSE - TO OBTAIN THE FIRST PARTIAL DIVIDEND WITH
C THE REQUIRED NUMBER OF DIGITS
C
C INPUT PARAMETERS
C A1 : LINEAR ARRAY, THE FIRST TRIAL DIVIDEND
C B : LINEAR ARRAY, THE DIVISOR
C J : THE NUMBER OF PIECES IN THE DIVISOR
C I1 : THE NUMBER OF DIGITS IN THE LEFTMOST PIECE OF
C THE DIVIDEND
C J1 : THE NUMBER OF DIGITS IN THE LEFTMOST PIECE OF
C THE DIVISOR
C IGIT: LINEAR ARRAY, THE REMAINING DIGITS OF THE DIVIDEND
C NDI : THE NUMBER OF THE REMAINING DIGITS
C IDEND: THE LEFTMOST NDIG DIGITS NUMBER OF THE DIVIDEND
C ISOR: THE LEFTMOST NDIG DIGITS NUMBER OF THE DIVISOR
C
C OUTPUT PARAMETERS
C A1 : LINEAR ARRAY, THE FIRST PARTIAL DIVIDEND
C IGIT: LINEAR ARRAY, REMAINING DIGITS OF THE DIVIDEND
C NDI : THE NUMBER OF THE REMAINING DIGITS
C IDEND: THE LEFTMOST NDIG DIGITS NUMBER OF THE DIVIDEND
C ISOR: THE LEFTMOST NDIG DIGITS NUMBER OF THE DIVISOR

SUBROUTINE PARDEE (A1, B, J, I1, J1, IGIT, NDI, ISOR, IDEND)
PARAMETER (ND = 20, NDIG=4)
INTEGER A1 (ND), IGIT (80), B (ND)
C Set IDEND and ISOR equal to the leftmost NDIG digits of the
C dividend and divisor
IF (I1 .NE. NDIG) THEN
   IDEND = A1 (J) * 10 ** (NDIG - I1) + (A1 (J - 1)/10
1 ** I1)
ENDIF
ISOR = B (J) * 10 ** (NDIG - J1) + (B (J - 1)/10
1 ** J1)
IF (IDEND .LT. ISOR) THEN
   ICUT = I1 - (J1 + 1)
ELSE
   ICUT = I1 - J1
ENDIF
IF (ICUT .EQ. 0) RETURN
C Digitise the excess digits of rightmost piece of the
C first partial dividend
MICUT = NDIG - ICUT
ITEM = MOD (A1 (1), 10 ** ICUT)
DO 11 I = 1, ICUT
   IGIT (NDI + I) = MOD (ITEM, 10)
11 CONTINUE
ITEM = ITEM / 10
CONTINUE
NDI = NDI + ICUT
C Form partial dividend with left most required number of digits
DO 22 II = 1, J - 1
   ITEMP = MOD (A1 (II + 1), 10 ** ICUT)
   A1 (II) = ITEMP * 10 ** (NDIG - ICUT) + A1 (II) / 10 ** ICUT
22 CONTINUE
A1 (J) = A1 (J) / 10 ** ICUT
RETURN
END

C PURPOSE - THIS SUBROUTINE FORMRS NEW NUMBER BY
C APPENDING DIGIT(S) TO THE RIGHT
C INPUT PARAMETERS
C A1 : LINEAR ARRAY, EXISTING INTEGER
C MT : NUMBER OF PIECES IN THE EXISTING INTEGER
C OUTPUT PARAMETERS
C A1 : LINEAR ARRAY, THE INTEGER AFTER APPENDING
C MT : NUMBER OF PIECES IN THE INTEGER AFTER APPENDING
SUBROUTINE APPEND (A1, MT, IW, IND)
PARAMETER (ND = 20)
INTEGER A1 (ND)
IF (A1 (MT) .NE. 0) THEN
   CALL SHIFT (A1, MT, 1, IND)
   A1 (1) = A1 (1) + IW
ELSE
   IF (IW .NE. 0) THEN
      MT = 1
      A1 (1) = IW
   ENDIF
ENDIF
RETURN
END

C PURPOSE - TO FIND THE EXACT DIGIT(S) OF THE QUOTIENT AND
C REMAINING PARTIAL DIVIDEND
C INPUT PARAMETERS
C B : LINEAR ARRAY, THE DIVISOR
C J : NUMBER OF PIECES IN THE DIVISOR
C A1 : LINEAR ARRAY, THE PARTIAL DIVIDEND
C MT : NUMBER OF PIECES IN PARITAL DIVIDEND
C IW : THE TRIAL DIGIT(S) OF QUOTIENT
C OUTPUT PARAMETERS
C A1 : LINEAR ARRAY, REMAINING PARTIAL DIVIDEND
C MT : NUMBER OF PIECES IN REMAINING PARITAL DIVIDEND
C IW : THE REQUIRED DIGIT(S) OF THE QUOTIENT
SUBROUTINE EXACDQ (A1, MT, B, J, IW, IND)
PARAMETER (ND = 20)
INTEGER B (J), A1 (MT), D1 (ND), C (1), C1 (ND)
C (1) = IW

C Multiply the divisor by the trial digit(s) of the quotient
CALL BIGMUL (B, J, C, 1, D1, NN, IND)
IF (IND .EQ. 1) THEN
  IND = 0
ELSE
C Compare the product of trial digit(s) and divisor with
C partial dividend
  IF (NN .LE. MT) THEN
    IF (NN .EQ. MT) THEN
      DO 22 KK = NN, 1, -1
        IF (D1 (KK) .LT. A1 (KK)) GO TO 30
        IF (D1 (KK) .GT. A1 (KK)) GO TO 50
      22 CONTINUE
      A1 (1) = 0
      MT = 1
      RETURN
    ENDIF
  30 CALL BIGSUB (A1, MT, D1, NN, C1, MM, INDEX)
    DO 44 II = 1, MM
      A1 (II) = C1 (II)
    44 CONTINUE
    MT = MM
    RETURN
ENDIF

C Reduce the trial digit(s) of quotient by unity, if required
50 IW = IW - 1
GO TO 10
END

THE NUMBERS OF PIECES ARE  2  1
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
67894521
2000
THE QUOTIENT IS
33947
THE REMAINDER IS
521
THE NUMBERS OF PIECES ARE  2  2
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
65490000
86300000
THE QUOTIENT IS
0
THE REMAINDER IS 65490000
THE NUMBERS OF PIECES ARE 3 1
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
199900000000
19
THE QUOTIENT IS 10521052631
THE REMAINDER IS 11
THE NUMBERS OF PIECES ARE 5 5
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
690804000012000044
78900280000560078
THE QUOTIENT IS 8
THE REMAINDER IS 59601760007519420
THE NUMBERS OF PIECES ARE 10 2
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
8965784345267543688500008763804304982901
78549
THE QUOTIENT IS 114142565090167203764529258982346114
THE REMAINDER IS 74315
THE NUMBERS OF PIECES ARE 15 1
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
6848964532789654328965838007880870769654940004369543200890
2657
THE QUOTIENT IS 257770586362425830973497850199800816580707566567385472
THE REMAINDER IS 1786
THE NUMBERS OF PIECES ARE 15 15
THE DIVIDEND AND DIVISOR ARE AS FOLLOWS
9000000000002000000000000000000000000000000000000000000000000000000
9000000000002000000000000000000000000000000000000000000000000000000
THE QUOTIENT IS 1
THE REMAINDER IS 1000000000000000100000000000000
THE NUMBERS OF PIECES ARE 21 20
DIVIDEND TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 20 21
DIVISOR TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 21 21
BOTH NUMBERS TOO LARGE, PLEASE TYPE AGAIN
THE NUMBERS OF PIECES ARE 2 0
5.5 Given a positive integer this program computes the exact integer part of its square root

```
IMPLICIT INTEGER*4 (A - Z)
10 READ (4, '(112)0 NUMBER
IF (NUMBER .EQ. 0) STOP
CALL SQRUT (NUMBER, ISQRT, IGUESV, IORIGU, NOP)
NS = ISQRT * ISQRT
NSPl = (ISQRT + 1) * (ISQRT + 1)
WRITE (6,'(" THE NUMBER",10," ROOT",16," GUES",17,
1" ORIGUE",17)') NUMBER, ISQRT, IGUESV, IORIGU
WRITE (6,'(" ITERATIONS",12," SQ OF ROOT",110,
1" SQ OF R00T+1",I11)') NOP, NS, NSPl
WRITE (6, *)
GO TO 10
END
```

When the subroutine SQRUT is called from the main program (the preceding one), it has the following arguments: NUMBER, ISQRT, IGUESV, IORIGU, NOP i.e. it is of the form

```
SUBROUTINE SQRUT (NUMBER, ISQRT, IGUESV, IORIGU, NOP)
```

instead of

```
SUBROUTINE SQRUT (NUMBER, ISQRT)
```

C PURPOSE - GIVEN A POSITIVE INTEGER THE PROGRAM FIRST COMPUTE
C THE CLOSE INTEGER INITIAL ESTIMATE OF THE SQUARE
C ROOT AND THEN THE EXACT INTEGER PART OF THE
C SQUARE ROOT USING VARIANT OF NEWTON-RAPHSON METHOD

```
SUBROUTINE SQRUT (NUMBER, ISQRT)
IMPLICIT INTEGER*4 (A - Z)
CALL SIMPGV (NUMBER, IGUESV, NB)
IORIGU = IGUESV
C Improve the guess value for odd bits number
IF (NB / 2 * 2 .NE. NB .AND. NB .GT. 2) THEN
   IGUESV = (IGUESV + 1) * 71 / 100
ENDIF
CALL TONRAP (IGUESV, NUMBER, ISQRT, NOP)
```
SUBROUTINE SIMPGV (NUMBER, IGUESV, NB)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 30)
DIMENSION LTAB (ND)
KK = NUMBER
IDETER = 2
NB = 1
LTAB (1) = 2
C Determine the number of bits
10 IF (NUMBER .GE. IDETER) THEN
   IDETER = IDETER + IDETER
   NB = NB + 1
   LTAB (NB) = IDETER
   GO TO 10
ENDIF
IF (NB .GT. 2) THEN
   IHALF = NB / 2
C For odd number of bits
   IF (NB .NE. 2 * IHALF) THEN
      IHALF = IHALF + 1
      KK = 2 * NUMBER
   ENDIF
   IGUESV = KK / LTAB (IHALF + 1)
   IGUESV = IGUESV + LTAB (IHALF - 1)
ELSE
C The number of bits is less than equal to 2
   IGUESV = 1
ENDIF
RETURN
END

C PURPOSE - TO EVALUATE THE SQUARE ROOT BY NEWTON-RAPHSON METHOD
C INPUT PARAMETERS
C IGUESV : THE INITIAL ESTIMATE OF THE SQUARE ROOT
C NUMBER : NUMBER THE SQUARE ROOT OF WHICH IS TO BE EVALUATED
C OUTPUT PARAMETERS
C IX : THE REQUIRED SQUARE ROOT
C NOP : THE NUMBER OF ITERATIONS
SUBROUTINE TONRAP (IGUESV, NUMBER, IX, NOP)
IMPLICIT INTEGER*4 (A - Z)
NOP = 0
IX = IGUESV
10
IY = (IX + NUMBER / IX) / 2
IF (IY .GE. IX) RETURN
IX = IY
NOP = NOP + 1
GO TO 10
END

THE NUMBER 1 ROOT 1 GUES 1 ORIGUE 1
ITERATIONS 0 SQ OF ROOT 1 SQ OF ROOT+1 4
THE NUMBER 4 ROOT 2 GUES 2 ORIGUE 3
ITERATIONS 0 SQ OF ROOT 4 SQ OF ROOT+1 9
THE NUMBER 6 ROOT 2 GUES 2 ORIGUE 3
ITERATIONS 0 SQ OF ROOT 4 SQ OF ROOT+1 9
THE NUMBER 11 ROOT 3 GUES 3 ORIGUE 3
ITERATIONS 0 SQ OF ROOT 9 SQ OF ROOT+1 16
THE NUMBER 16 ROOT 4 GUES 4 ORIGUE 6
ITERATIONS 0 SQ OF ROOT 16 SQ OF ROOT+1 25
THE NUMBER 25 ROOT 5 GUES 5 ORIGUE 7
ITERATIONS 0 SQ OF ROOT 25 SQ OF ROOT+1 36
THE NUMBER 256 ROOT 16 GUES 17 ORIGUE 24
ITERATIONS 1 SQ OF ROOT 256 SQ OF ROOT+1 289
THE NUMBER 1324 ROOT 36 GUES 37 ORIGUE 52
ITERATIONS 1 SQ OF ROOT 1296 SQ OF ROOT+1 1369
THE NUMBER 7068 ROOT 84 GUES 85 ORIGUE 119
ITERATIONS 1 SQ OF ROOT 7056 SQ OF ROOT+1 7225
THE NUMBER 4096 ROOT 64 GUES 68 ORIGUE 96
ITERATIONS 1 SQ OF ROOT 4096 SQ OF ROOT+1 4225
THE NUMBER 9999 ROOT 99 GUES 103 ORIGUE 103
ITERATIONS 2 SQ OF ROOT 9801 SQ OF ROOT+1 10000
THE NUMBER 16396 ROOT 128 GUES 137 ORIGUE 192
ITERATIONS 1 SQ OF ROOT 16384 SQ OF ROOT+1 16641
THE NUMBER 99999 ROOT 316 GUES 320 ORIGUE 451
ITERATIONS 1 SQ OF ROOT 99856 SQ OF ROOT+1 100489
THE NUMBER 100111 ROOT 316 GUES 320 ORIGUE 451
ITERATIONS 1 SQ OF ROOT 99856 SQ OF ROOT+1 100489
THE NUMBER 1558888 ROOT 1248 GUES 1268 ORIGUE 1785
ITERATIONS 1 SQ OF ROOT 1557504 SQ OF ROOT+1 1560001
THE NUMBER 16999935 ROOT 4123 GUES 4382 ORIGUE 6171
ITERATIONS 2 SQ OF ROOT 16999129 SQ OF ROOT+1 17007376
THE NUMBER 999111189 ROOT 31608 GUES 31629 ORIGUE 31629
ITERATIONS 1 SQ OF ROOT 999065664 SQ OF ROOT+1 999128881
THE NUMBER 121000000 ROOT 11000 GUES 11060 ORIGUE 15577
ITERATIONS 1 SQ OF ROOT 121000000 SQ OF ROOT+1 121022001
5.6 The program to generate all the prime numbers

PARAMETER (ND = 26379)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION IPRIM (ND)
COMMON/A1/IPRIM

10 WRITE (6, '((" TYPE THE LIMIT IN FORMAT I8")')
READ (4, '(I8)') LIMIT
WRITE (6, '((IX, I8)') LIMIT
IF (LIMIT .EQ. 0) STOP
IF (LIMIT .LT. 0) THEN
   WRITE (6, '((" LIMIT CANNOT BE NEGATIVE")')
   GO TO 10
ENDIF
CALL PRIME (LIMIT, NP)
WRITE (6,'((" PRIME NUMBERS ARE AS FOLLOWS")'(1X, 10I8))')
KIPRIM (II), II = 1, NP)
WRITE (6, '((" TOTAL NUMBER OF PRIMES = ", 15)') NP
GO TO 10
END

When the subroutine PRIME is called from the main program (the preceding one), it has the following arguments: LIMIT, NP, i.e. it is of the form
SUBROUTINE PRIME (LIMIT, NP)
instead of
SUBROUTINE PRIME (LIMIT)

C PURPOSE :- TO GENERATE ALL THE PRIME NUMBERS UPTO A CERTAIN UPPER LIMIT

SUBROUTINE PRIME (LIMIT)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION S (100), IPRIM (26379), IREP (8), INTV (2)
COMMON/A1/IPRIM
CALL UPTHIR (LIMIT, IPRIM, NP, INDEX)

C Set the initial values
NOFT = 0
IF (INDEX .EQ. 0) THEN
   IND = 1
   INTV (1) = 49
   INTV (2) = 77
   IREP (1) = 1
   DO 22 I = 2, 8
      IREP (I) = IPRIM (I + 2)
   CONTINUE
IND = 1
NSTOR = 8
M = 3
ENDIF
30  NOFT = NOFT + 30
C Check whether all possible numbers are tested for primality
IF (NOFT + 29 .GT. LIMIT) THEN
   IF (NOFT + 1 .GT. LIMIT) RETURN
   DO 44 II = NSTOR - 1, 2, -1
      IF (NOFT + IREP (II) .LE. LIMIT) THEN
         NSTOR = II
         GO TO 50
      ENDIF
   44 CONTINUE
   NSTOR = 1
ENDIF
C Test whether the number lies between P[j] square and P[j+1]P[j+2]
50  DO 88 JJ = 1, NSTOR
    N = NOFT + IREP (JJ)
    IF (N .EQ. INTV (IND)) THEN
       IF (IND .EQ. 1) THEN
          IND = 2
       ELSE
          C Initialize for N=P[j+1]P[j+2]
          IND = 1
          M = M + 1
          S (M) = 2 * IPRIM (M) + INTV (2)
          PMPl = IPRIM (M + 1)
          INTV (1) = PMPl * PMPl
          INTV (2) = PMPl * IPRIM (M + 2)
       ENDIF
    ELSE
       IF (M .GT. 3) THEN
          C Carry out the primality test
          DO 77 J = 4, M
             IF (N .GE. S (J)) THEN
                IF (N .EQ. S (J)) THEN
                   S (J) = S (J) + 2 * IPRIM (J)
                   GO TO 88
                ELSE
                   S (J) = S (J) + 2 * IPRIM (J)
                   GO TO 60
                ENDIF
          77 CONTINUE
       ENDIF
       NP = NP + 1
       IPRIM (NP) = N
    ENDIF
ENDIF

CONTINUE
GO TO 30
END

C PURPOSE :- THIS SUBROUTINE COMPUTES ALL THE PRIME NUMBERS FROM 2
C TO A GIVEN NATURAL NUMBER LESS THAN EQUAL TO 30
C INPUT PARAMETER
C LIMIT : THE NUMBER UPTO WHICH PRIME IS REQUIRED
C OUTPUT PARAMETER
C IPRIM : ONE-DIMENSIONAL ARRAY, CONTAINS ALL THE PRIME
C NUMBERS UPTO A CERTAIN LIMIT
C NP : THE NUMBER OF PRIMES
C INDEX : 0 IF THE NUMBER IS GREATER THAN 29
C : 1 IF THE NUMBER IS LESS THAN 29
SUBROUTINE UPTHIR (LIMIT, IPRIM, NP, INDEX)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION IPRIM (10), IREP (8)
C Set the initial values
IPRIM (1) = 2
NP = 1
IREP (1) = 1
NOFT = 0
NSTOR = 1
NTIME = 2
IADD = 2
   10 DO 33 J = 1, NTIME
      NOFT = NOFT + IADD
      DO 22 I = 1, NSTOR
         N = NOFT + IREP (I)
      C Check whether all the numbers are tested for primality
      IF (N .GT. LIMIT) THEN
         INDEX = 1
         RETURN
      ENDIF
      C The only composite number
      IF (N .NE. 25) THEN
         NP = NP + 1
         IPRIM (NP) = N
      ENDIF
      22 CONTINUE
33 CONTINUE
C Determine primes upto 30 is computed
IF (NOFT .EQ. 24) THEN
   INDEX = 0
   RETURN
ENDIF
C Initialization for numbers greater than 5
IREP (2) = IPRIM (3)
NSTOR = 2
NTIME = 4
NOFT = 0
IADD = 6
GO TO 10
END

TYPE THE LIMIT IN FORMAT I8
  11
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11
TOTAL NUMBER OF PRIMES =  5
TYPE THE LIMIT IN FORMAT I8
  24
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
TOTAL NUMBER OF PRIMES =  9
TYPE THE LIMIT IN FORMAT I8
  25
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
TOTAL NUMBER OF PRIMES =  9
TYPE THE LIMIT IN FORMAT I8
  30
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
TOTAL NUMBER OF PRIMES = 10
TYPE THE LIMIT IN FORMAT I8
  31
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
TOTAL NUMBER OF PRIMES = 11
TYPE THE LIMIT IN FORMAT I8
  59
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
  31  37  41  43  47  53  59
TOTAL NUMBER OF PRIMES = 17
TYPE THE LIMIT IN FORMAT I8
  60
PRIME NUMBERS ARE AS FOLLOWS
  2  3  5  7  11  13  17  19  23
TOTAL NUMBER OF PRIMES = 17
TYPE THE LIMIT IN FORMAT I8
100
PRIME NUMBERS ARE AS FOLLOWS
2 3 5 7 11 13 17 19 23 29
31 37 41 43 47 53 59 61 67 71
73 79 83 89 97
TOTAL NUMBER OF PRIMES = 25
TYPE THE LIMIT IN FORMAT I8
0

5.7 This program creates a nest of DO loops and finds the distinct indices of the loops and their sum

INTEGER P
PARAMETER (P = 20)
DIMENSION LINV (P), LTEV (P), LCRE (P), LDEX (P), INTERP(P)
10 WRITE (6, '(IX, "TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)")')
20 READ (4, '(I2)') ND
WRITE (6,'(IX, I2)') ND
IF (ND .LT. 0) THEN
   WRITE (6, '(IX, "DATA ILLEGAL, KINDLY TYPE IT AGAIN")')
   GO TO 20
ELSEIF (ND .GT. P) THEN
   WRITE (6, '(IX, "THIS NUMBER WILL CROSS THE DIMENSION, CHANGE THE DIMENSION")')
ELSEIF (ND .NE. 0) THEN
   WRITE (6,'(IX, "TYPE INITIAL, TEST VALUES AND INCREMENTS OF" ')
   1 /1X,'DO LOOPS IN FORMAT (20I2), ONE SET PER LINE")')
   DO 44 J = 1, ND
      30 READ (4, '(3I2)') LINV (J), LTEV (J), LCRE (J)
      WRITE (6,'(IX, I3)') LINV (J), LTEV (J), LCRE (J)
      IF (LCRE (J) .EQ. 0) THEN
         WRITE (6, '(IX, "NO INCREMENT CAN BE ZERO, KINDLY TYPE THIS SET AGAIN")')
         GO TO 30
      ENDIF
   44 CONTINUE
   WRITE (6,'(IX, "TYPE THE NUMBER OF ELEMENTS IN THE SET OF:" ')
   1 /1X,'PROHIBITED SEQUENCE IN FORMAT (I2)")')
   45 READ (4, '(I2)') NINT
WRITE (6, '(1X, I2)') NINT
IF (NINT .EQ. 0) THEN
  IOVER = 1
ELSEIF (NINT .GT. ND) THEN
  WRITE (6, '(1X, ’NUMBER OF ELEMENTS CANNOT EXCEED’, 1X,
  ’NUMBER OF LOOPS, KINDLY TYPE AGAIN’)')
  GO TO 45
ELSEIF (NINT .LT. 0) THEN
  WRITE (6, '(1X, ’NUMBER OF ELEMENTS CANNOT NEGATIVE,’,
  1 ,1X, ’KINDLY TYPE AGAIN’)')
  GO TO 45
ELSE
  WRITE (6, '(1X, ’TYPE THE SET OF VALUES OF THE’, 1X,
  ’PROHIBITED SEQUENCE IN FORMAT (20I2)’)')
  READ (4, '(20I2)') (INTERP (K), K = 1, ND)
  WRITE (6, '(1X, 20I2)') (INTERP (K), K = 1, ND)
  IOVER = 0
ENDIF

C Starting set of values of the DO loops
CALL START (LDEX, ND, LINV, LTEV, LCRE)
LP = 1
INDEX = 0
ICHEK = 0
50 CALL PRE (LDEX, ND, INDEX, LP)
IF (INDEX .EQ. 0) GO TO 70
C Change the values of the indices of the DO loops
60 CALL NEST (LDEX, ND, INDEX, LP, LINV, LTEV, LCRE)
IF (INDEX .EQ. 0 .AND. ND .GT. 1) GO TO 50
C To perform calculation using distinct indices
70 CALL CORE (LDEX, ND, INDEX, ICHEK)
IF (INDEX .EQ. 1) GO TO 10
IF (IOVER .EQ. 1) GO TO 60
C To interrupt a sequence at a particular position
CALL POST (LDEX, LTEV, ND, INTERP, NINT, IOVER)
GO TO 60
ENDIF
STOP
END

C PURPOSE : TO START DO LOOPS FROM ANY SET OF STARTING VALUES
C INPUT PARAMETERS
C LINV : LINEAR ARRAY, CONTAINS INITIAL VALUES
C LTEV : LINEAR ARRAY, CONTAINS TEST VALUES
C LCRE : LINEAR ARRAY, CONTAINS INCREMENTS
C ND : DENOTES THE NUMBER OF DO LOOPS
C OUTPUT PARAMETERS:
C LDEX : LINEAR ARRAY, CONTAINS STARTING VALUES
SUBROUTINE START (LDEX, ND, LINV, LTEV, LCRE)
INTEGER P
PARAMETER (P = 20)
DIMENSION LINV (ND), LTEV (ND), LCRE (ND), LDEX (ND), LNET (P)
WRITE (6,'(IX,' Type the set of starting values,'MX,
1 "ONE VALUE PER LINE IN FORMAT (I2)'')
C Validate the data
DO 22 J = 1, ND
   READ (4, '(I2)') LNET (J)
   WRITE (6, '(IX, 12)') LNET (J)
   IF (LINV (J) .NE. LNET (J) ) THEN
      IF ((LCRE (J) .LT. 0 .AND. (LINV (J) .LT. LNET (J) .OR.
1 . OR. (LCRE (J) .GT. 0 .AND. (LINV (J) .GT. LNET (J) .OR.
1 . OR. (MOD (LNET (J) - LINV (J), LCRE(J)) .NE. 0)) THEN
         WRITE (6,'(IX, "STARTING VALUE OF LOOP NUMBER" ,IX,
1 "IS WRONG, PLEASE TYPE IT AGAIN")') J
         GO TO 10
   END IF
22 CONTINUE
C Put indices of all DO loops equal to the starting values
DO 33 K = 1, ND
   LDEX (K) = LNET (K)
33 CONTINUE
RETURN
END

C PURPOSE - TO FIND OUT WHETHER THE INDICES OF THE
C DO LOOPS ARE DISTINCT
C INPUT PARAMETERS
C ND : NUMBER OF DO LOOPS
C LP : THE LOOP NUMBER
C INDEX : = 1 IF THE CONDITION IS NOT SATISFIED
C = 0 OTHERWISE
SUBROUTINE PRE (LDEX, ND, INDEX, LP)
DIMENSION LDEX (ND)
IF (ND .GT. 1) THEN
   IF (LP .EQ. 1) LP = 2
   DO 22 K = LP, ND
      IPV = LDEX (K)
      C Compare the indices of the DO loops
      DO 11 J = 1, K - 1
         IF (LDEX (J) .EQ. IPV) THEN
C If all indices are not distinct
C PURPOSE - TO CHANGE VALUES OF INDICES OF DO LOOPS
C INPUT PARAMETERS
C LDEX : LINEAR ARRAY, CONTAINS THE STARTING VALUES
C LINV : LINEAR ARRAY, CONTAINS THE INITIAL VALUES
C LTEV : LINEAR ARRAY, CONTAINS THE TEST VALUES
C LCRE : LINEAR ARRAY, CONTAINS THE INCREMENTS
C ND : DENOTES NUMBER OF LOOPS
C INDEX = 0 IF INCREMENT OF A PARTICULAR LOOP IS POSSIBLE
C = 1 OTHERWISE
C LP : IS A PARTICULAR LOOP NUMBER
C OUTPUT PARAMETER
C LDEX : LINEAR ARRAY, CONTAINS AUGMENTED VALUES
C INDICES OF DO LOOPS

SUBROUTINE NEST (LDEX, ND, INDEX, LP, LINV, LTEV, LCRE)
DIMENSION LDEX (ND), LINV (ND), LTEV (ND), LCRE (ND)
IF (INDEX .EQ. 1) THEN
INDEX = 0
ELSEIF (LP .LT. ND) THEN
LP = ND
ENDIF
M = LP
C Check whether a particular index can be increased or decreased
DO 11 J = 1, M
LDEX (LP) = LDEX (LP) + LCRE (LP)
IF (((LCRE (LP) .LT. 0 .AND. LDEX (LP) .GE. LTEV (LP))
1 .OR. (LCRE (LP) .GT. 0 .AND. LDEX (LP) .LE. LTEV (LP)))) RETURN
LDEX (LP) = LINV (LP)
LP = LP - 1
11 CONTINUE
INDEX = 1
RETURN
END

C PURPOSE : TO PERFORM THE CALCULATION, USING A DISTINCT
C SET OF INDICES
C INPUT PARAMETERS
C LDEX : LINEAR ARRAY, IS THE SET OF DISTINCT VALUES OF THE INDICES
SUBROUTINE CORE (LDEX, ND, INDEX, ICHEK)
DIMENSION LDEX (ND)
IF (INDEX .EQ. 1) THEN
  IF (ICHEK .EQ. 1) RETURN
  WRITE (6,-'(lX,^^N0 SET OF DISTINCT VALUES EXISTS')0'
  RETURN
ELSE
  IF (ICHEK .EQ. 0) THEN
    ICHEK = 1
    WRITE (6,'(IX, "SET OF DISTINCT VALUES/PERMUTATIONS"
        , 1X,"AND THEIR SUM")')
  ENDIF
  NSUM = 0
  DO 11 I = 1, ND
    NSUM = NSUM + LDEX (I)
  11 CONTINUE
  WRITE (6, '(IX, 1515)') (LDEX (J), J = 1, ND), NSUM
ENDIF
RETURN
END

C PURPOSE : TO INTERRUPT A SEQUENCE AT A POSITION K (1 ≤ K < ND)
C INPUT PARAMETERS
C LDEX : LINEAR ARRAY, CONTAINS VALUES OF INDICES
C OF THE DO LOOPS
C LTEV : LINEAR ARRAY, CONTAINS TEST VALUES
C INTERP: LINEAR ARRAY, CONTAINS SET OF VALUES,
C AFTER WHICH INTERRUPTION IS NECESSARY
C ND : NUMBER OF DO LOOPS
C K : THE POSITION NUMBER OF THE INTERRUPTION
C IOVER: 0 IF INTERRUPTION IS NOT CARRIED OUT
C : 1 IF INTERRUPTION IS CARRIED OUT
C OUTPUT PARAMETERS
C LDEX : LAST (ND - (K+1)) CONTAINS TEST VALUES OF RESPECTIVE LOOPS,
C AND THE REST REMAIN UNCHANGED
SUBROUTINE POST (LDEX, LTEV, ND, INTERP, K, IOVER)
DIMENSION LDEX (ND), LTEV (ND), INTERP (ND)
IF (ND .GT. K) THEN
  DO 11 I = 1, ND
    IF (LDEX (I) .NE. INTERP (I)) RETURN
  11 CONTINUE
  IOVER = 1
DO 22 J = K + 1, ND, 1
    LDEX (J) = LTEV (J)
22    CONTINUE
END

TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)
  2
TYPE INITIAL, TEST VALUES AND INCREMENTS OF
DO LOOPS IN FORMAT (2012), ONE SET PER LINE
  5 6 1
  5 9 1
TYPE THE NUMBER OF ELEMENTS IN THE SET OF PROHIBITED SEQUENCE IN FORMAT (I2)
  1
TYPE THE SET OF VALUES OF THE PROHIBITED SEQUENCE IN FORMAT (2012)
  5 7
TYPE THE SET OF STARTING VALUES, ONE VALUE PER LINE IN FORMAT (I2)
  5
  5

SET OF DISTINCT VALUES/PERMUTATIONS AND THEIR SUM
  5  6  11
  5  7  12
  6  5  11
  6  7  13
  6  8  14
  6  9  15

TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)
  3
TYPE INITIAL, TEST VALUES AND INCREMENTS OF
DO LOOPS IN FORMAT (2012), ONE SET PER LINE
  6 1-2
  4 5 1
  3 3-1
TYPE THE NUMBER OF ELEMENTS IN THE SET OF PROHIBITED SEQUENCE IN FORMAT (I2)
  3
TYPE THE SET OF VALUES OF THE PROHIBITED SEQUENCE IN FORMAT (2012)
  2 3 4
TYPE THE SET OF STARTING VALUES, ONE VALUE PER LINE IN FORMAT (I2)
  4
  5
  4
STARTING VALUE OF LOOP NUMBER 3 IS WRONG, PLEASE TYPE IT AGAIN
  3
SET OF DISTINCT VALUES/PERMUTATIONS AND THEIR SUM
  4  5  3  12
  2  4  3  9
  2  5  3 10
TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)
  3
TYPE INITIAL, TEST VALUES AND INCREMENTS OF
DO LOOPS IN FORMAT (20I2), ONE SET PER LINE
  8 8 7
  8 9 6
1215 5
TYPE THE NUMBER OF ELEMENTS IN THE SET OF PROHIBITED SEQUENCE IN FORMAT (I2)
  0
TYPE THE SET OF STARTING VALUES, ONE VALUE PER LINE IN FORMAT (I2)
  8
  8
12
NO SET OF DISTINCT VALUES EXISTS
TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)
  2
TYPE INITIAL, TEST VALUES AND INCREMENTS OF
DO LOOPS IN FORMAT (20I2), ONE SET PER LINE
  2 3 1
  10 8-2
TYPE THE NUMBER OF ELEMENTS IN THE SET OF PROHIBITED SEQUENCE IN FORMAT (I2)
  1
TYPE THE SET OF VALUES OF THE PROHIBITED SEQUENCE IN FORMAT (20I2)
  310
TYPE THE SET OF STARTING VALUES, ONE VALUE PER LINE IN FORMAT (I2)
  2
  4
STARTING VALUE OF LOOP NUMBER 2 IS WRONG, PLEASE TYPE IT AGAIN
  8
SET OF DISTINCT VALUES/PERMUTATIONS AND THEIR SUM
  2 8 10
  3 10 13
TYPE THE NUMBER OF DO LOOPS IN FORMAT (I2)
  2
TYPE INITIAL, TEST VALUES AND INCREMENTS OF
DO LOOPS IN FORMAT (20I2), ONE SET PER LINE
  2 7 0
NO INCREMENT CAN BE ZERO, KINDLY TYPE THIS SET AGAIN
  5 5 4
  1 3 2
TYPE THE NUMBER OF ELEMENTS IN THE SET OF PROHIBITED SEQUENCE IN FORMAT (I2)
  3
NUMBER OF ELEMENTS CANNOT EXCEED NUMBER OF LOOPS, KINDLY TYPE AGAIN
  2
TYPE THE SET OF VALUES OF THE PROHIBITED SEQUENCE IN FORMAT (20I2)
  1 2
TYPE THE SET OF STARTING VALUES, ONE VALUE PER LINE IN FORMAT (I2)
  5
  3
5.8 Program to reconstruct a polynomial from its integer zeros

```
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION ITZ (ND), ICOF (ND)
10 WRITE (6, '(" TYPE THE NUMBER OF ZEROS IN FORMAT (12)"
   1,1X,
   1,1X,'FOR STOPPING PLEASE TYPE : 0")')
20 READ (4, '(I2)') NZ
WRITE (6,'(1X, I2)') NZ
IF (NZ .EQ. 0) STOP
IF (NZ .LT. 0) THEN
   WRITE (6, '(" DATA ILLEGAL, KINDLY TYPE AGAIN")')
   GO TO 20
ELSEIF (NZ .GT. ND) THEN
   WRITE (6, '(" THE GIVEN NUMBER WILL CROSS THE DIMENSION,"
   1,1X,'CHANGE DIMENSION")')
   STOP
ENDIF
WRITE (6, '(" TYPE ALL ZEROS IN FORMAT (10I5)")')
READ (4, '(1SI5)') (ITZ (K), K = 1, NZ)
WRITE (6, '(1X, 10I5)') (ITZ (K), K = 1, NZ)
CALL RECONZ (ITZ, NZ, ICOF)
WRITE (6, '(" COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
   1/1X, "STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS")')
WRITE (6, '(1X, 10(I7, 1X))') (ICOF (K), K = 1, NZ)
GO TO 10
```
SUBROUTINE RECONZ (ITZ, NZ, ICOF)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ITZ (NZ), ICOF (NZ)
C Form the linear polynomial with the first zero
ICOF (1) = - ITZ (1)
M = 1
IF (NZ .GT. 1) THEN
C Multiply the existing polynomial by linear polynomial
DO 33 I = 2, NZ
    CALL LINZ (ITZ (I), ICOF, M)
33 CONTINUE
ENDIF
RETURN
END

SUBROUTINE LINZ (MLZ, ICOF, M)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ICOF (M)
M = M + 1
ICOF (M) = - MLZ * ICOF (M - 1)
IF (M .NE. 2) THEN
    DO 11 K = 1, M - 2
        ICOF (M - K) = ICOF (M - K) - MLZ * ICOF (M - K - 1)
11 CONTINUE
ENDIF
ICOF (1) = ICOF (1) - MLZ
RETURN
END
TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
1

TYPE ALL ZEROS IN FORMAT (1015)
42745
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-42745

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
2

TYPE ALL ZEROS IN FORMAT (1015)
256 421
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-677 107776

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
3

TYPE ALL ZEROS IN FORMAT (1015)
11 22 33
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-66 1331 -7986

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
4

TYPE ALL ZEROS IN FORMAT (1015)
10 20 30 40
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-100 3500 -50000 240000

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
5

TYPE ALL ZEROS IN FORMAT (1015)
1 2 3 4 5
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-15 85 -225 274 -120

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
8

TYPE ALL ZEROS IN FORMAT (1015)
1 2 3 4 5 6 7 8
COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,
STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS
-36 546 -4536 22449 -67284 118124 -109584 40320

TYPE THE NUMBER OF ZEROS IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
-2
DATA ILLEGAL, KINDLY TYPE AGAIN
30
THE GIVEN NUMBER WILL CROSS THE DIMENSION, CHANGE DIMENSION

5.9 Program for the reconstruction of a polynomial over $Z$ from its rational zeros

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (WD = 10)
DIMENSION NUM (ND), IDEN (ND), ICOF (11)

10 WRITE (6, '(" TYPE THE NUMBER OF ZEROS IN FORMAT (12)"
1/,1X, "FOR STOPPING PLEASE TYPE : 0")')
20 READ (4, '(12)0 NZ
WRITE (6, '(IX, 12)') NZ
IF (NZ .EQ. 0) STOP
IF (NZ .LT. 0) THEN
  WRITE (6, "(" DATA ILLEGAL, KINDLY TYPE AGAIN")")
  GO TO 20
ELSEIF (NZ .GT. ND) THEN
  WRITE (6, "(" THE GIVEN NUMBER WILL CROSS THE DIMENSION,"
1,1X,"CHANGE DIMENSION")")
  STOP
ENDIF
WRITE (6, "(" TYPE ALL NUMERATOR OF ZEROS IN FORMAT (1015)")")
READ (4, '(1015)') (NUM (K), K = 1, NZ)
WRITE (6, '(IX, 1015)') (NUM (K), K = 1, NZ)
WRITE (6, "(" TYPE ALL DENOMINATOR OF ZEROS IN FORMAT (1015)")")
30 READ (4, '(1515)') (IDEN (K), K = 1, NZ)
WRITE (6, '(1X, 1015)') (IDEN (K), K = 1, NZ)
DO 44 K = 1, NZ
  IF (IDEN (K) .EQ. 0) THEN
    WRITE (6, "(" NO DENOMINATORS CAN BE ZERO, KINDLY TYPE"
1 ,1X,"ALL THE DENOMINATORS AGAIN")")
  GO TO 30
ENDIF
44 CONTINUE
CALL RECONQ (NUM, IDEN, NZ, ICOF, M)
WRITE (6, "(" COEFFICIENTS OF THE REQUIRED POLYNOMIAL,"
1 ,1X,"ARE AS FOLLOWS")")
WRITE (6, '(1X, 10(17, IX))') (ICOF (K), K = 1, M)
GO TO 10
END

C PURPOSE - RECONSTRUCTION OF A POLYNOMIAL OVER $Z$ FROM ITS
SUBROUTINE RECQNQ (NUM, IDEN, NZ, ICOF, M)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION NUM (NZ), IDEN (NZ), ICOF (11)
C Validate the data
DO 55 K = 1, NZ
   ICN = NUM (K)
   ICD = IDEN (K)
   IF (ICN .EQ. 0) THEN
      ICD = 1
   ELSE
      IF (ICD .LT. 0) THEN
         ICD = - ICD
         ICN = - ICN
      ENDIF
      IHCF = HCF (ICN, ICD)
      IF (IHCF .NE. 1) THEN
         ICN = ICN / IHCF
         ICD = ICD / IHCF
      ENDIF
   ENDIF
   IF (K .NE. 1) THEN
      CALL LINQ (ICN, ICD, ICOF, M)
   ELSE
      Form a linear polynomial with the first zero
      ICOF (1) = ICD
      ICOF (2) = - ICN
      M = 2
   ENDIF
55 CONTINUE
RETURN
END

C PURPOSE - MULTIPLY A POLYNOMIAL BY A LINEAR FACTOR
C INPUT PARAMETERS
C MLNZ : THE NUMERATOR OF THE CONSTANT TERM OF THE
C LINEAR POLYNOMIAL
C MLNZ : THE DENOMINATOR OF THE CONSTANT TERM OF THE
C LINEAR POLYNOMIAL
C ICOF : COEFFICIENT OF THE POLYNOMIAL TO BE MULTIPLIED
C M : DEGREE OF THE POLYNOMIAL
SUBROUTINE LINQ (MLNZ, MLDZ, ICOF, M)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ICOF (M)
M = M + 1
ICOF (M) = - MLNZ * ICOF (M - 1)
IF (M .NE. 2) THEN
    MN2 = M - 2
    DO 11 K = 1, MN2
        ICOF (M - K) = MLDZ * ICOF (M - K) - MLNZ * ICOF (M - K - 1)
    11 CONTINUE
ENDIF
ICOF (1) = MLDZ * ICOF (1)
RETURN
END

TYPE THE NUMBER OF ZEROS IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
1
TYPE ALL NUMERATOR OF ZEROS IN FORMAT (10I5)
21245
TYPE ALL DENOMINATOR OF ZEROS IN FORMAT (10I5)
42182
COEFFICIENTS OF THE REQUIRED POLYNOMIAL, ARE AS FOLLOWS
6026  -3035
TYPE THE NUMBER OF ZEROS IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
2
TYPE ALL NUMERATOR OF ZEROS IN FORMAT (10I5)
652   100
TYPE ALL DENOMINATOR OF ZEROS IN FORMAT (10I5)
267   200
COEFFICIENTS OF THE REQUIRED POLYNOMIAL, ARE AS FOLLOWS
534  -1571   652
TYPE THE NUMBER OF ZEROS IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
3
TYPE ALL NUMERATOR OF ZEROS IN FORMAT (10I5)
25   50   60
TYPE ALL DENOMINATOR OF ZEROS IN FORMAT (10I5)
32   24   56
COEFFICIENTS OF THE REQUIRED POLYNOMIAL, ARE AS FOLLOWS
5376 -21160  25250  -9375
TYPE THE NUMBER OF ZEROS IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
4
5.10 Program for the reconstruction of polynomial over $\mathbb{Z}$ from its complex and surd zeros

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 20)
DIMENSION A (ND), B (ND), ICOF (ND)

WRITE (6, '('
1/IX, 'TYPE THE NUMBER OF ZEROS IN FORMAT (I2)''
1 /1X, 'FOR STOPPING PLEASE TYPE : 0''))
READ (4, '(I2)') NZ
WRITE (6,'(I1X, I2)') NZ
IF (NZ .EQ. 0) STOP
IF (NZ .LT. 0) THEN
  WRITE (6, '('
 DATA ILLEGAL, KINDLY TYPE AGAIN''
  GO TO 20
ELSEIF (NZ .GT. ND) THEN
  WRITE (6, '
 THE GIVEN NUMBER WILL CROSS THE DIMENSION,''
1,1X, 'CHANGE DIMENSION''
  STOP
ENDIF
WRITE (6, '('
1/IX, 'TYPE ALL ZEROS IN FORMAT (10i4)''
READ (4, '(10i4)') (A (K), B (K), K = 1, NZ)
WRITE (6, '
1,1X, 'CHANGE DIMENSION''
STOP

END
CALL RCONSC (A, B, NZ, ICOF, N)
WRITE (6, '(" COEFFICIENTS OF THE REQUIRED MONIC POLYNOMIAL,\n1/IX, 'STARTING FROM THE 2ND COEFFICIENT, ARE AS FOLLOWS")')
WRITE (6, '(IX, 10(17, IX))') (ICOF (K), K = 1, N)
IF (NZ .EQ. 8) STOP
GO TO 10
END

C PURPOSE - RECONSTRUCTION OF POLYNOMIAL OVER Z FROM ITS
C COMPLEX AND SURDS ZEROS
C INPUT PARAMETERS
C A : LINEAR ARRAY, THE INTEGER PART OF THE ZEROS
C B : LINEAR ARRAY, THE IMAGINARY/IRRATIONAL PART OF THE ZEROS
C NZ : THE NUMBER OF ZEROS
C OUTPUT PARAMETER
C ICOF : LINEAR ARRAY, THE COEFFICIENTS OF THE POLYNOMIAL

SUBROUTINE RCONSC (A, B, NZ, ICOF, N)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION A (NZ), B (NZ), ICOF (N)
C Form the quadratic polynomial with the first pair of zeros
ICOF (1) = - 2 * A (1)
ICOF (2) = (A (1)) ** 2 - B (1)
N = 2
IF (NZ .GT. 1) THEN
C Multiply the existing polynomial by quadratic polynomial
DO 33 I = 2, NZ
   C = - 2 * A (I)
   D = (A (I)) ** 2 - B (1)
   CALL QMUL (ICOF, N, C, D)
33 CONTINUE
ENDIF
RETURN
END

C PURPOSE - MULTIPLY A POLYNOMIAL BY A QUADRATIC FACTOR
C INPUT PARAMETERS
C P : LINEAR ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C TO BE MULTIPLIED
C N : THE DEGREE OF THE POLYNOMIAL
C C : THE SECOND COEFFICIENT OF THE QUADRATIC POLYNOMIAL,
C D : THE CONSTANT TERM OF THE QUADRATIC POLYNOMIAL
C OUTPUT PARAMETERS
C ICOF : COEFFICIENTS OF THE POLYNOMIAL AFTER MULTIPLICATION
C N : NEW DEGREE OF THE POLYNOMIAL

SUBROUTINE QMUL (P, N, C, D)
IMPLICIT INTEGER*4 (P - Z)
DIMENSION P (N)
P (N + 2) = 0
P (N + 1) = 0
DO 11 K = 0, N - 1
   P (N+2-K) = P (N+2-K) + C * P (N+1-K) + D * P (N - K)
11 CONTINUE
P (2) = P (2) + C * P (1) + D
P (1) = P (1) + C
N = N + 2
RETURN
END

5.11 Given a monic polynomial over Z, this program computes its integer zeros

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION PR (26379), ICOF (ND), ITZ (ND)
COMMON /A1/PR
CALL PRIME (304678)

10 WRITE(6,'(" TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (12)"/
   11X, "FOR STOPPING PLEASE TYPE : 0")')
20 READ(4, '(12)') M
   WRITE (6, '(IX, 12)') M
   IF (M .EQ. 0) STOP
   IF (M .LT. 0) THEN
      WRITE(6, '(" DATA ILLEGAL, KINDLY TYPE AGAIN")')
      GO TO 20
   ELSEIF (M .GT. ND) THEN
      WRITE (6,'(" THE GIVEN NUMBER WILL CROSS THE DIMENSION,"/
         1 ,IX, "CHANGE THE DIMENSION")')
      STOP
   ENDIF
   WRITE(6,'(" TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND"/
1/1X,"ONWARDS IN FORMAT (10I8)")')
   READ (4, '(10I8)') (ICOF (I), I = 1, M)
   WRITE(6, '(" THE",IX," INTEGRAL ROOTS OF THIS POLYNOMIAL"/
      11X,"WHICH ARE NOT INTEGERS")') NUMZ
   WRITE (6,'(" THE",IX," INTEGER ROOTS OF THIS POLYNOMIAL")')
C PURPOSE - TO FIND INTEGER ZEROS (OTHER THAN 1,0,-1) OF A MONIC
C POLYNOMIAL OVER THE SET OF INTEGERS
C
C INPUT PARAMETERS
C ICOF : ONE-DIMENSIONAL ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C NC : THE DEGREE OF THE POLYNOMIAL
C
C OUTPUT PARAMETERS
C ITZ : ONE-DIMENSIONAL ARRAY, CONTAINS ALL INTEGER ZEROS OF
C THE POLYNOMIAL
C NZ : NUMBER OF INTEGER ZEROS OF THE POLYNOMIAL

SUBROUTINE ZEROS (ICOF, NC, ITZ, NZ)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 20)
DIMENSION ICOF (NC), ITZ (ND), IFAC (500), IVAL (3)
COMMON /Bl/IFAC, NF /Dl/KK
IND = 0
NZ = 0
CALL TELIN (ICOF, NC, ITZ, NZ, IVAL, IND)
IF (IND .EQ. 1) RETURN
IF (NC .NE. 1) THEN
C Compute zeros of nonlinear polynomial
NEFC = 1
KK = IABS (ICOF (NO))
C Find all the factors of the last non-zero coefficient
CALL FAC
10 NEFC = NEFC + 1
IF (NEFC .GT. NF) RETURN
IC = IFAC (NEFC)
20 ISUM = 1
IF (IVAL (2) / (IC + 1) * (IC + 1) .EQ. IVAL (2)) THEN
IF (IVAL (1) / (IC - 1) * (IC - 1) .EQ. IVAL (1)) THEN
C Test a particular factor satisfies the polynomial
DO 33 K = 1, NC
ISUM = ISUM * IC + ICOF (K)
33 CONTINUE
ENDIF
ENDIF
IF (ISUM .NE. 0) THEN
C Determine the factor is positive or negative
IF (IC .LT. 0) GO TO 10
ELSE
WRITE(6,'(" ALL THE ROOTS OF THIS POLYNOMIAL ARE"'
1 ,1X," INTEGERS AND ARE"'/1X,10I7')@(ITZ(KK), KK=1,M)
ENDIF
GO TO 10
END
IC = - IC
GO TO 20
ENDIF
NZ = NZ + 1
C The integer zeros of the polynomial
ITZ (NZ) = IC
CALL POLDIV (ICOF, NC, IC)
IF (NC .NE. 1) GO TO 20
ENDIF
C For linear polynomial
NZ = NZ + 1
ITZ (NZ) = - ICOF (1)
RETURN
END

C PURPOSE - TO FIND THE INTEGER ZEROS 1,0,-1 TOGETHER WITH THEIR
C MULTIPLICITIES IF THEY EXIST AND TO DETERMINE THE
C INTEGER ZEROS OTHER THAN THESE
C
C INPUT PARAMETERS
C ICOF : ONE DIMENSIONAL ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C ID : THE DEGREE OF THE POLYNOMIAL
C
C OUTPUT PARAMETERS
C ICOF : ONE DIMENSIONAL ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C AFTER REMOVING THE ZEROS 1,0,1 (IF THEY EXIST)
C ID : THE DEGREE OF THE POLYNOMIAL AFTER REMOVING THE ZEROS 1,0,-1
C ITZ : ONE DIMENSIONAL ARRAY, CONTAINS ZEROS 1,0,-1 (IF THEY EXIST)
C OF THE POLYNOMIAL
C NZ : NUMBER OF ZEROS 1,0,-1 OF THE POLYNOMIAL
C IVAL : ONE DIMENSIONAL ARRAY, THE NON-ZERO VALUES OF THE DEFLATED
C POLYNOMIAL (AFTER REMOVING THE ZEROS 1,0,-1) AT 1,0,-1
C IND : 0 INITIALLY
C : 1 IF THE POLYNOMIAL DOES NOT HAVE ANY INTEGER ZEROS
C : OTHER THAN 1,0,-1
C
SUBROUTINE TELIN (ICOF, ID, ITZ, NZ, IVAL, IND)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 20)
DIMENSION ICOF (ID), IVAL (3), IFAC (500), ITZ (ND)
C Test whether 0 is one of the zeros of the polynomial
NC = ID
DO 11 J = 1, ID
   IF (ICOF (NC) .NE. 0) GO TO 20
   NZ = NZ + 1
   ITZ (NZ) = 0
   NC = NC - 1
11 CONTINUE
IND = 1
RETURN
20 IC = 1
ID = NC
C Test 1 and -1 satisfy the polynomial
DO 55 I = 1, ID
   ISUM = 1
   DO 44 K = 1, NC
      ISUM = ISUM * IC + ICOF (K)
   CONTINUE
   IF (ISUM .NE. 0) GO TO 60
   CALL POLDIV (ICOF, NC, IC)
   NZ = NZ + 1
   ITZ (NZ) = IC
55 CONTINUE
IND = 1
RETURN
60 IF (IC .EQ. 1) THEN
   IC = -1
   GO TO 30
ENDIF
ID = NC
C Compute the values of the polynomial at -1,1,0
IVAL (2) = ISUM
ISUM = 1
DO 77 K = 1, ID
   ISUM = ISUM + ICOF (K)
77 CONTINUE
IVAL (1) = ISUM
IVAL (3) = ICOF (ID)
DO 88 I = 1, 3
   IF (IVAL (I) .EQ. IVAL (I) / 3 * 3) RETURN
88 CONTINUE
C Linear factors do not exist
IND = 1
RETURN
END

C PURPOSE - DEFLATION OF THE MONIC POLYNOMIAL BY A LINEAR FACTOR
C INPUT PARAMETERS
C ICOF : ONE DIMENSIONAL ARRAY, CONTAINS THE COEFFICIENTS OF
C THE POLYNOMIAL TO BE DEFLATED
C ID : THE DEGREE OF THE POLYNOMIAL TO BE DEFLATED
C IC : THE CONSTANT TERM OF THE LINEAR FACTOR OF THE POLYNOMIAL
C OUTPUT PARAMETERS
C ICOF : ONE-DIMENSIONAL ARRAY, COEFFICIENTS OF THE
C DEFLATED POLYNOMIAL
C ID : THE DEGREE OF THE DEFLATED POLYNOMIAL
SUBROUTINE POLDIV (ICOF, ID, IC)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ICOF (ID)

C Reduce the degree of the polynomial by unity
   ID = ID - 1
   ICOF (1) = ICOF (1) + IC
   IF (ID .GT. 1) THEN
   C Synthetic division of the polynomial by the linear factor
   DO 10 I = 2, ID
       ICOF (I) = ICOF (I) + IC * ICOF (I-1)
   10 CONTINUE
   ENDIF
   RETURN
END

C PURPOSE - THE SUBROUTINE CALCULATES ALL FACTORS OF
C A POSITIVE INTEGER
C INPUT PARAMETER
C NUMBER : THE POSITIVE INTEGER, FACTORS OF WHICH ARE REQUIRED
C OUTPUT PARAMETERS
C FACTOR : ONE-DIMENSIONAL ARRAY, CONTAINS ALL THE FACTORS
C N : NUMBER OF FACTORS
SUBROUTINE FAC
IMPLICIT INTEGER*4 (A - Z)
DIMENSION PRIME (26379), FACTOR (500), FACT (8), POWER (8)
COMMON/Al/PRIME /Bl/ FACTOR, N /C1/FACT, POWER, NOF /D1/NUMBER

C Special treatment for unity
IF (NUMBER .EQ. 1) THEN
   N = 1
   FACTOR (1) = 1
ELSE
   C Find all prime factors of an integer>1 with their powers
   N1 = NUMBER
   J = 1
   NOF = 0
   DIV = 2
   10 I = 1
   CALL SQRUT (N1, LIMIT)
   20 IF (DIV .GT. LIMIT) THEN
       NOF = NOF + 1
       POWER (NOF) = 1
       FACT (NOF) = N1
   ELSE
       QUOT = N1 / DIV
       IF (N1 .GT. QUOT * DIV) THEN
           J = J + 1
           DIV = PRIME (J)
       ENDIF
   ENDIF
   GO TO 10
ENDIF
RETURN
END
GO TO 20
ENDIF
IF (I .EQ. 2) THEN
   POWER (NOF) = POWER (NOF) + 1
ELSE
   NOF = NOF + 1
   POWER (NOF) = 1
   FACT (NOF) = DIV
   I = 2
ENDIF
N1 = QUOT
C Test for unity
IF (N1 .GT. 1) GO TO 30
ENDIF
M = 1
N = 1
FACTOR (1) = 1
PROD = 1
C Find all the factors of the integer
40 FACTM = FACT (M)
POWERM = POWER (M)
LIMIT = PROD * POWERM
DO 55 J1 = 1, LIMIT
   N = N + 1
   FACTOR (N) = FACTOR (N - PROD) * FACTM
55 CONTINUE
IF (M .LT. NOP) THEN
   PROD = PROD + LIMIT
   M = M + 1
   GO TO 40
ENDIF
C Arrange the factors in ascending order
IF (NOF .GT. 1) CALL SORT (FACTOR, N)
ENDIF
RETURN
END
C PURPOSE :- TO ARRANGE A GIVEN SET OF NUMBERS IN ASCENDING
C ORDER
C INPUT AND OUTPUT PARAMETER
C FAC INTEGER, ONE DIMENSIONAL ARRAY CONTAINS THE NUMBERS
C L THE NUMBER OF NUMBERS
SUBROUTINE SORT (FAC, L)
IMPLICIT INTEGER*4 (A-Z)
DIMENSION FAC (500)
LM1 = L-1
C Compare and rearrange the numbers if required
DO 22 I = 1,LM1
   IP1 = I + 1
   NSMALL = IP1
   SMALL = FAC (NSMALL)
C Check last but one number is reached
IF (I .NE. LM1) THEN
   DO 11 K = I + 2,L
      IF (SMALL .GT. FAC (K)) THEN
         NSMALL = K
         SMALL = FAC (NSMALL)
      ENDIF
   11 CONTINUE
C Compare and rearrange if required the last two numbers
ELSEIF (SMALL .LT. FAC(I)) THEN
   COPY = FAC(I)
   FAC (I) = SMALL
   FAC (NSMALL) = COPY
ENDIF
22 CONTINUE
RETURN
END

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
1
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
   -1
ALL THE ROOTS OF THIS POLYNOMIAL ARE INTEGERS AND ARE
1
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
2
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
   -4   4
ALL THE ROOTS OF THIS POLYNOMIAL ARE INTEGERS AND ARE
2
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
3
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
   -6  11  -6
ALL THE ROOTS OF THIS POLYNOMIAL ARE INTEGERS AND ARE
1  2  3
5.12 The program to calculate the H.C.F. of two polynomials over the unique factorization domain \( \mathbb{Z} \)

```plaintext
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION U (ND), V (ND), POHCF (ND)
```

10 WRITE (6, '("TYPE THE DEGREES OF THE TWO POLYNOMIALS")')

```
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
4
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
-10 35 -50 24
ALL THE ROOTS OF THIS POLYNOMIAL ARE INTEGERS AND ARE
1 2 4 3
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
5
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
-15 85 -225 274 -120
ALL THE ROOTS OF THIS POLYNOMIAL ARE INTEGERS AND ARE
1 2 4 3 5
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
5
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
2 -3 -15 32 -12
THIS POLYNOMIAL HAS NO INTEGRAL ROOTS
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
8
TYPE THE COEFFICIENTS OF THE POLYNOMIAL FROM 2ND
ONWARDS IN FORMAT (10I8)
-2 -4 8 5 -14 -2 16 8
THIS POLYNOMIAL HAS 2 ROOTS WHICH ARE NOT INTEGERS
THE 6 INTEGRAL ROOTS OF THIS POLYNOMIAL ARE
-1 -1 -1 -1 2 2
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
-2
DATA ILLEGAL, KINDLY TYPE AGAIN
25
THE GIVEN NUMBER WILL CROSS THE DIMENSION, CHANGE THE DIMENSION
```
READ (4, '(I)') NCF, NCS
WRITE (6, '(IX, 1I2)') NCF, NCS
NCF = NCF + 1
NCS = NCS + 1

IF (NCF .EQ. 0 .AND. NCS .EQ. 0) STOP
IF (NCF .LE. 0 .OR. NCS .LE. 0) THEN
  WRITE (6, '(" THE DEGREES OF THE POLYNOMIALS CANNOT"
1 1X, "BE NEGATIVE")')
  GO TO 10
ELSEIF (NCF .GE. 10 .OR. NCS .GE. 10) THEN
  WRITE (6, '( " THE DEGREE(S) OF SOME POLYNOMIAL(S)"
1 1X, "EXCEED(S) THE GIVEN DIMENSION")')
  STOP
ELSE
  WRITE (6, '(" TYPE THE COEFFICIENTS OF THE FIRST"
1 1X, "POLYNOMIAL IN FORMAT (10I8)")')
  READ (4, '(10I8)') (U (I), I = NCF, 1, -1)
  WRITE (6, '(IX, 10I8)') (U (I), I = NCF, 1, -1)
  IF (U (NCF) .EQ. 0) THEN
    WRITE (6, '(" THE LEADING COEFFICIENT OF THE"
1 1X, "FIRST POLYNOMIAL CANNOT BE ZERO")')
    GO TO 20
  ENDIF
20 WRITE (6, '(" TYPE THE COEFFICIENTS OF THE SECOND"
1 1X, "POLYNOMIAL IN FORMAT (10I8)")')
  READ (4, '(10I8)') (V (I), I = NCS, 1, -1)
  WRITE (6, '(IX, 10I8)') (V (I), I = NCS, 1, -1)
  IF (V (NCS) .EQ. 0) THEN
    WRITE (6, '(" THE LEADING COEFFICIENT OF THE"
1 1X, "SECOND POLYNOMIAL CANNOT BE ZERO")')
    GO TO 30
  ENDIF
30 ENDIF

CALL POLHCF (U, NCF, V, NCS, POHCF, ICS2)
WRITE (6, '(" HCF OF THE POLYNOMIALS IS")')
WRITE (6, '(IX, 10I8)') (POHCF (I), I = ICS2, 1, -1)
GO TO 10
END

C PURPOSE - THIS PROGRAM CALCULATES THE HCF OF TWO POLYNOMIALS OVER
C THE UNIQUE FACTORIZATION DOMAIN Z
C
C U : LINEAR ARRAY, THE FIRST POLYNOMIAL
C N : NUMBER OF COEFFICIENTS IN THE FIRST POLYNOMIAL
C V : LINEAR ARRAY, THE SECOND POLYNOMIAL
C NV : NUMBER OF COEFFICIENTS IN THE SECOND POLYNOMIAL
SUBROUTINE POLHCF (U, N, V, NV, POHCF, ICS2)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION U (N), V (NV), R (ND), POHCF (ND), DEND (ND)
 IF (NV .GT. N) THEN
     DO 11 I = 1, NV
         DEND (I) = V (I)
    11 CONTINUE
     ICF1 = NV
     DO 22 I = 1, N
         POHCF (I) = U (I)
    22 CONTINUE
     ICS2 = N
 ELSE
     DO 44 I = 1, N
         DEND (I) = U (I)
    44 CONTINUE
     ICF1 = N
     DO 55 I = 1, NV
         POHCF (I) = V (I)
    55 CONTINUE
     ICS2 = NV
END IF
CALL REMCOM (DEND, ICF1)
60 CALL POLDI (DEND, ICF1, POHCF, ICS2, R, ICR)
 IF (ICR .GT. 1) THEN
     C Change the divisor and dividend
     DO 77 I = ICS2, 1, -1
         DEND (I) = POHCF (I)
    77 CONTINUE
     DO 88 I = ICR, 1, -1
         POHCF (I) = R (I)
    88 CONTINUE
     ICF1 = ICS2
     ICS2 = ICR
     GO TO 60
ELSEIF (ICR .EQ. 1) THEN
     POHCF (1) = 1
     ICS2 = 1
ENDIF
RETURN
END
C PURPOSE - THIS SUBROUTINE CALCULATES THE REMAINDER POLYNOMIAL BY
C PSEUDO-DIVISION OF A POLYNOMIAL BY ANOTHER POLYNOMIAL
C
C INPUT PARAMETERS
C U : LINEAR ARRAY, THE DIVIDEND
C N : NUMBER OF COEFFICIENTS IN THE DIVIDEND
C V : LINEAR ARRAY, THE DIVISOR
C NV : NUMBER OF COEFFICIENTS IN THE DIVISOR
C
C OUTPUT PARAMETERS
C R : LINEAR ARRAY, THE REMAINDER AFTER THE PSEUDO-DIVISION
C NR : THE NUMBER OF COEFFICIENTS IN THE REMAINDER

SUBROUTINE POLDI (U, N, V, NV, R, NR)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION U (N), V (NV), R (N)
DO 11 J = 1, N
   R (J) = U (J)
11 CONTINUE
DO 44 K = N - NV, 0, -1.
   NVPK = NV + K
   IHCF = HCF (R (NVPK), V (NV))
   LCDEND = R (NVPK) / IHCF
   LCISOR = V (NV) / IHCF
   DO 22 I = NVPK - 1, 1, -1
      R (I) = LCISOR * R (I)
22 CONTINUE
   DO 33 J = NVPK - 1, K + 1, -1
      R (J) = R (J) - LCDEND * V (J - K)
33 CONTINUE
   DO 77 I = NVPK - 1, 1, -1
      IF (R (I) .NE. 0) THEN
         CALL REMCOM (R, I)
         IF (I .LT. NV) THEN
            NR = I
            RETURN
         END IF
      END IF
77 CONTINUE
   NR = 0
44 CONTINUE
RETURN
END

TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
   1 1
TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
724 2172
HCF OF THE POLYNOMIALS IS
1
3
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
2 1

TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
112 560 672
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
172 344
HCF OF THE POLYNOMIALS IS
1
2
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
4 3

TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
2 -40 300 -1000 1250
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
5 -75 375 -625
HCF OF THE POLYNOMIALS IS
1 -15 475 -125
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
3 4

TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
5 -75 375 -625
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
2 -40 300 -1000 1250
HCF OF THE POLYNOMIALS IS
1 -15 475 -125
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
3 8

TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
1 0 2 -2
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
1 -2 -4 0 -15 -4 9 2 5
HCF OF THE POLYNOMIALS IS
1
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
8 4

TYPE THE COEFFICIENTS OF THE FIRST POLYNOMIAL IN FORMAT (10I8)
8 -16 -32 64 40 -112 -16 128 64
TYPE THE COEFFICIENTS OF THE SECOND POLYNOMIAL IN FORMAT (10I8)
12 -48 36 48 -48
HCF OF THE POLYNOMIALS IS
1 -3 0 4
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
-2 1

THE DEGREES OF THE POLYNOMIALS CANNOT BE NEGATIVE
TYPE THE DEGREES OF THE TWO POLYNOMIALS IN FORMAT (2I2)
5.13 Program to find the quadratic factors of a polynomial over Z

IMPLICIT INTEGER*4 (A-Z)
PARAMETER (ND = 10)
DIMENSION PR (26379), ICOF (ND), ICQF (20)
COMMON /A1/PR
CALL PRIME (304678)
10 WRITE(6,'(1X,12)') N
READ (4, '(12)') N
WRITE (6, '(1X,12)') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
WRITE(6, '12 DATA ILLEGAL, KINDLY TYPE AGAIN')
GO TO 20
ELSEIF (N .GT. ND) THEN
WRITE (6, '12 THE GIVEN NUMBER WILL CROSS DIMENSION,')
STOP
ENDIF
READ (4, '(1017)') (ICOF (I), I = 1, N)
WRITE(6, '12 THE POLYNOMIAL IS GIVEN BELOW')
WRITE(6, '(1X, 10I8)') (ICOF (I), I = 1, N)
CALL REIMA (ICOF, N, ICQF, NQF)
IF (NQF .EQ. 0) THEN
WRITE (6, '12 THE POLYNOMIAL DOES NOT HAVE A QUADRATIC FACTOR')
ELSE
WRITE (6, '12 THE COEFFICIENTS OF THE QUADRATIC FACTORS STARTING FROM THE 2ND ARE GIVEN BELOW')
DO 33 I = 1, NQF, 2
WRITE (6, '(1X, 10I4/)') ICQF (I), ICQF (I+1)
33 CONTINUE
ENDIF
GO TO 10
END

C PURPOSE- COMPUTATION OF QUADRATIC FACTOR OF A POLYNOMIAL
C INPUT PARAMETERS
C ICOF : LINEAR ARRAY, THE COEFFICIENT OF THE POLYNOMIAL

THE DEGREE(S) OF SOME POLYNOMIAL(S) EXCEED(S) THE GIVEN DIMENSION
C N : THE DEGREE OF THE POLYNOMIAL
C OUTPUT PARAMETERS
C ICQF : LINEAR ARRAY, THE 2ND AND 3RD COEFFICIENTS OF ALL THE
C QUADRATIC FACTORS
C NQF : THE TOTAL NUMBER OF 2ND AND 3RD COEFFICIENTS

SUBROUTINE REIMA (ICOF, N, ICQF, NQF)
  IMPLICIT INTEGER*4 (A-Z)
  PARAMETER (ND = 10)
  DIMENSION IFAC (500), ICOF (N), ITZ (ND), ICO (ND), ICQF (20),
  IQUOT (ND), IREM (ND)
  COMMON /Bl/IFAC, NF /D1/KK1
  KK1 = IABS (ICOF (N))
  CALL FAC
  IQ = IFAC (1)
  JJ = 1
  NQF = 0
  DO 11 I = 1, N
     ICO (I) = ICOF (I)
  11 CONTINUE
  NC = N

C Test the necessity of computing the polynomials in
C another indeterminant
20 IF (NC .GE. 3) THEN
   CALL CPDP (ICO, NC, IQ, IQUOT, MINUS1, IREM, MINUS2)
C Check whether all the coefficients of a particular
C polynomial obtained in the above process are zero
   DO 22 J = 1, MINUS2
      IF (IREM (J) .NE. 0) GO TO 40
   22 CONTINUE
   DO 33 J = 1, MINUS1
      IREM (J) = IQUOT (J)
   33 CONTINUE
   MINUS2 = MINUS1
   GO TO 50
40 CALL POLHCF (IQUOT, MINUS1, IREM, MINUS2, IREM, MINUS2)
C Determine whether the HCF of the polynomials is other
C than the constant polynomial
50 IF (MINUS2 .GT. 1) THEN
   IF (MINUS2 .EQ. 2) THEN
      ITZ (1) = - IREM (2) / IREM (1)
      NZ = 1
   ELSE
      MINUS2 = MINUS2 - 1
   DO 66 J = 1, MINUS2
      IREM (J) = IREM (J + 1)
  66 CONTINUE
CALL ZEROS (IREM, MINUS2, ITZ, NZ)
ENDIF

IF (NZ .GE. 1) THEN
C The 2nd and 3rd coefficients of the quadratic
C factor (if exists)
DO 77 I = 1, NZ
   IP = - ITZ (I)
   ICQF (NQF + 1) = IP
   NQF = NQF + 2
   ICQF (NQF) = IQ
   CALL DIVPMQ (ICO, NC, IP, IQ)
   IF (NC .LT. 3) GO TO 80
77 CONTINUE
ENDIF
ENDIF

IF (JJ .LE. NF) THEN
IF (IQ .GT. 0) THEN
   IQ = - IFAC (JJ)
   JJ = JJ + 1
ELSE
   IQ = IFAC (JJ)
ENDIF
GO TO 20
ENDIF
RETURN
ENDIF
RETURN
ENDIF

80 IF (NC .EQ. 2) THEN
C The 2nd and 3rd coefficients of the last
C quadratic factor
   ICQF (NQF + 1) = ICO (1)
   NQF = NQF + 2
   ICQF (NQF) = ICO (2)
ENDIF
RETURN
END

C PURPOSE- DIVISION OF A POLYNOMIAL BY A MONIC QUADRATIC
C FACTOR
C INPUT PARAMETERS
C ICO : LINEAR ARRAY, THE COEFFICIENT OF THE POLYNOMIAL
C NC : THE NUMBER OF THE COEFFICIENT OF THE POLYNOMIAL
C IP : THE 2ND COEFFICIENT OF THE QUADRATIC FACTOR
C IQ : THE LAST COEFFICIENT OF THE QUADRATIC FACTOR
C OUTPUT PARAMETERS
C ICO : LINEAR ARRAY, THE COEFFICIENT OF THE POLYNOMIAL
C AFTER DIVISION
C NC : THE NUMBER OF THE COEFFICIENT OF THE POLYNOMIAL
C AFTER DIVISION
SUBROUTINE DIVPMQ (ICO, NC, IP, IQ)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ICO (NC)
ICO (1) = ICO (1) - IP
ICO (2) = ICO (2) - IQ - ICO (1) * IP
DO 11 I = 3, NC - 2
  ICO (I) = - ICO (I-2) * IQ - ICO (I-1) * IP + ICO (I)
11 CONTINUE
NC = NC - 2
RETURN
END

C PURPOSE - THIS PROGRAM READS IN ALL THE COEFFICIENTS OF A MONIC
C POLYNOMIAL OF ANY DEGREE > 2 AND < N+1, DIVIDES IT BY
C A QUADRATIC POLYNOMIAL OF THE FORM X**2 - P*X + Q (Q
C BEING A FACTOR OF THE CONSTANT TERM OF THE ORIGINAL
C POLYNOMIAL), AND PRINTS OUT ALL THE COEFFICIENTS OF THE
C 2 POLYNOMIALS IN P, WHOSE VALUES MUST SIMULTANEOUSLY
C VANISH FOR THOSE VALUES OF P FOR WHICH THE ABOVE DIVISION
C LEAVES NO REMAINDER
C
C INPUT PARAMETERS
C ICO : LINEAR ARRAY, THE COEFFICIENT OF THE POLYNOMIAL
C NC : NUMBER OF COEFFICIENT OF THE POLYNOMIAL
C LAST : THE FACTOR OF THE LAST NON-ZERO COEFFICIENT
C
C OUTPUT PARAMETERS
C IQUOT : LINEAR ARRAY, THE COEFFICIENTS OF THE POLYNOMIAL C (P)
C MINUS1 : NUMBER OF COEFFICIENTS OF THE POLYNOMIAL C (P)
C IREM : LINEAR ARRAY, THE COEFFICIENTS OF THE POLYNOMIAL D (P)
C MINUS2 : NUMBER OF COEFFICIENTS OF THE POLYNOMIAL D (P)

SUBROUTINE CPDP (ICO, NC, LAST, IQUOT, MINUS1, IREM, MINUS2)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION ICO (NC), IQUOT (NC), IREM (NC)
ICARE = ICO (NC) / LAST
IQUOT (1) = ICO (1)
IF (NC .EQ. 3) THEN
  IREM (1) = ICARE
  IREM (2) = - LAST + ICO (2)
  MINUS2 = NC - 1
ELSE
  IQUOT (1) = ICO (1)
  MINUS2 = NC - 2
  IREM (1) = - LAST
  IQUOT (2) = - LAST + ICO (2)
  IREM (2) = - LAST * ICO (1) + ICO (3)
  DO 22 K = 1, NC - 4
    IREM (K + 2) = - LAST * IQUOT (K + 1) + ICO (K + 3)
    IQUOT (K + 2) = IREM (K + 1)
22 CONTINUE
IF (K .EQ. 1) THEN
    IREM (2) = - LAST * IQUOT (1)
ELSE
    IF (K .GE. 3) THEN
        DO 11 J = K , 3 , - 1
            IREM (J + 1) = - LAST * IQUOT (J)
            IQUOT (J + 1) = IQUOT (J + 1) + IREM (J)
        11 CONTINUE
    ENDIF
    IQUOT (3) = IQUOT (3) + IREM (2)
    IREM (3) = - LAST * IQUOT (2)
ENDIF
    IQUOT (2) = IQUOT (2) + IREM (1)
22 CONTINUE
    IREM (NC - 3) = IREM (NC - 3) + ICARE
ENDIF
    IQUOT (NC - 2) = IQUOT (NC - 2) - ICARE
    MINUS1 = NC - 1
    DO 33 I = MINUS1 - 1, 1, -1
        IQUOT (I + 1) = IQUOT (I)
    33 CONTINUE
    IQUOT (1) = 1
RETURN
END

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
1
THE POLYNOMIAL IS GIVEN BELOW
    2
THE POLYNOMIAL DOES NOT HAVE A QUADRATIC FACTOR
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
2
THE POLYNOMIAL IS GIVEN BELOW
    2 4
THE COEFFICIENTS OF THE QUADRATIC FACTORS STARTING FROM THE 2ND ARE GIVEN BELOW
    2 4
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
3
THE POLYNOMIAL IS GIVEN BELOW
    0 61 24
THE POLYNOMIAL DOES NOT HAVE A QUADRATIC FACTOR
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
5

THE POLYNOMIAL IS GIVEN BELOW
\[ 4 \quad 6 \quad 1 \quad 4 \quad 6 \]

THE COEFFICIENTS OF THE QUADRATIC FACTORS
STARTING FROM THE 2ND ARE GIVEN BELOW
-1 1
4 6

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

6

THE POLYNOMIAL IS GIVEN BELOW
\[ 0 \quad -18 \quad 16 \quad 28 \quad -32 \quad 8 \]

THE COEFFICIENTS OF THE QUADRATIC FACTORS
STARTING FROM THE 2ND ARE GIVEN BELOW
-4 2
0 -2
4 -2

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

8

THE POLYNOMIAL IS GIVEN BELOW
\[ -2 \quad -4 \quad 8 \quad 5 \quad -14 \quad -2 \quad 16 \quad 8 \]

THE COEFFICIENTS OF THE QUADRATIC FACTORS
STARTING FROM THE 2ND ARE GIVEN BELOW
2 1
2 1
-2 2
-4 4

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

15

THE GIVEN NUMBER WILL CROSS DIMENSION, CHANGE THE DIMENSION

5.14 This program computes the rational zeros of a polynomial over \( \mathbb{Z} \)

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION PR (26379), ICOF (ND), ITZNUM (ND), ITZDEN (ND)
COMMON /A1/PR
CALL PRIME (304678)
10 WRITE(6,'(" TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)"/
11X,"' FOR STOPPING PLEASE TYPE : 0"'))
20 READ(4, '(I2)') M
WRITE (6, '(I1X, I2)') M
IF (M .EQ. 0) STOP
IF (M .LT. 0) THEN
WRITE(6, '(" DATA ILLEGAL, KINDLY TYPE AGAIN")')
GO TO 20
ELSEIF (M + 1 .GT. NO) THEN
WRITE(6, '(" THE GIVEN NUMBER WILL CROSS THE DIMENSION,")'
1, 1X, 'CHANGE THE DIMENSION")')
STOP
ENDIF
NC = M + 1
WRITE(6, '(" TYPE THE COEFFICIENTS OF THE POLYNOMIAL")'
1, 1X, 'IN FORMAT (10I8)")')
READ (4, '(10I8)') (ICOF (I), I = 1, NC)
WRITE(6, '(1X, 10I8)') (ICOF (I), I = 1, NC)
IF (ICOF (1) .EQ. 0) THEN
WRITE(6, '(" THE FIRST COEFFICIENT OF THE POLYNOMIAL")'
1, 1X, 'CANNOT BE ZERO/
1, 1X, 'TYPE THE COEFFICIENT OF THE")'
1, 1X, 'POLYNOMIAL AGAIN")
GO TO 30
ENDIF
CALL ZEROSQ (ICOF, NC, ITZNUM, ITZDEN, NZ)
IF (NZ .EQ. 0) THEN
WRITE(6, '(" THIS POLYNOMIAL HAS NO RATIONAL ZEROS")')
ELSE
IF (NZ .LT. M) THEN
WRITE(6, '(" ONLY", 12, " ZEROS OF THE POLYNOMIAL")'
1, 1X, 'ARE RATIONAL")') NZ
ELSE
WRITE(6, '(" ALL ZEROS OF THE POLYNOMIAL")'
1, 1X, 'ARE RATIONAL")')
ENDIF
WRITE(6, '(" THE NUMERATOR OF THE ZEROS")'
1, 1X, 'ARE")' /1X, 10I8)') (ITZNUM (KK), KK = 1, NZ)
WRITE(6, '(" THE DENOMINATOR OF THE ZEROS")'
1, 1X, 'ARE")' /1X, 10I8)') (ITZDEN (KK), KK = 1, NZ)
ENDIF
GO TO 10
END

C PURPOSE - THIS PROGRAM COMPUTES THE RATIONAL ZEROS OF THE
C POLYNOMIAL
C INPUT PARAMETERS
C ICOF : LINEAR ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C NC : THE NUMBER OF THE COEFFICIENT OF THE POLYNOMIAL
C OUTPUT PARAMETERS
C ITZNUM : LINEAR ARRAY, THE NUMERATOR OF THE ZEROS
C ITZDEN : LINEAR ARRAY, THE DENOMINATOR OF THE ZEROS
C NZ : THE NUMBER OF ZEROS
SUBROUTINE ZEROSQ (ICOF, NC, ITZNUM, ITZDEN, NZ)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION ICOF (NC), ITZNUM (NC), ITZDEN (NC), NEWCOF (ND)
CALL LEASTK (ICOF, NC, LN)
DO 11 I = 2, NC
   ICOF (I) = ICOF (I) * LN ** (I - 1)
11 CONTINUE
KDIV = ICOF (1)
DO 22 I = 1, NC - 1
   NEWCOF (I) = ICOF (I + 1) / KDIV
22 CONTINUE
CALL ZEROS (NEWCOF, NC-1, ITZNUM, NZ)
IF (NZ .EQ. 0) RETURN
DO 33 I = 1, NZ
   NHCF = HCF (ITZNUM (I), LN)
   ITZNUM (I) = ITZNUM (I) / NHCF
   ITZDEN (I) = LN / NHCF
33 CONTINUE
RETURN
END

C PURPOSE - THE PROGRAM FINDS THE LEAST NATURAL NUMBER BY
C WHICH ALL ZEROS OF A POLYNOMIAL SHOULD BE MULTIPLIED
C SO THAT THE POLYNOMIAL RECONSTRUCTED FROM NEW ZEROS
C MIGHT BE MONIC
C INPUT PARAMETERS
C ICOF : LINEAR ARRAY, COEFFICIENTS OF THE POLYNOMIAL
C NC : THE NUMBER OF THE COEFFICIENT OF THE POLYNOMIAL
C OUTPUT PARAMETER
C LN : THE REQUIRED LEAST NUMBER
SUBROUTINE LEASTK (ICOF, NC, LN)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION PR (26379), ICOF (NC), INCO (ND), IFAC (8), POWER (8)
COMMON /Al/PR /Cl/IFAC, POWER, NT /Dl/KK
LN = 1
C If the leading coefficient is 1 or - 1, then LN = 1
IF (IABS (ICOF (1)) .EQ. 1) RETURN
DO 11 J = 1, NC
   INCO (J) = ICOF (J)
11 CONTINUE
CALL REMCOM (INCO, NC)
IF (I .EQ. IABS (INCO(1))) RETURN
KK = IABS (INCO (1))
CALL FAC
C For a particular prime factor, calculate the least power
DO 44 K = 1, NT
KF = 1
IP = IFAC (K)
IE = POWER (K)
DO 22 I = 2, NC
   ICAR = I - 1
   IF (ICAR .EQ. IE) GO TO 30
   IF (INCO (I) .NE. 0) THEN
      M = 0
      ITEST = INCO (I)
      ICHEK = ITEST / IP
      IF (ITEST .NE. IP * ICHEK) THEN
         C Calculate the highest power of the current
         C prime factor
         L = (IE - M) / (I - 1)
         IF (L * (I - 1) .NE. IE - M) L = L + 1
         IF (L .GT. KF) KF = L
      ELSE
         ITEST = ICHEK
         M = M + 1
         C Check whether the coefficient plus the number of
         C times it is already divisible is equal to the power
         C of the factor
         IF (M + ICAR .NE. IE) GO TO 10
      ENDIF
   END IF
22 CONTINUE
30 LN = LN * IP ** KF
44 CONTINUE
RETURN
END

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (12)
FOR STOPPING PLEASE TYPE : 0
2
TYPE THE COEFFICIENTS OF THE POLYNOMIAL IN FORMAT (10I8)
   60 -414 156
ALL ZEROS OF THE POLYNOMIAL ARE RATIONAL
THE NUMERATOR OF THE ZEROS ARE
   2 13
THE DENOMINATOR OF THE ZEROS ARE
   5 2
TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0
3
TYPE THE COEFFICIENTS OF THE POLYNOMIAL IN FORMAT (10I8)
   50 175 175 125
ONLY 1 ZEROS OF THE POLYNOMIAL ARE RATIONAL
THE NUMERATOR OF THE ZEROS ARE
-5
THE DENOMINATOR OF THE ZEROS ARE
2

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

4
TYPE THE COEFFICIENTS OF THE POLYNOMIAL IN FORMAT (10I8)
24 14 -56 24 0
ALL ZEROS OF THE POLYNOMIAL ARE RATIONAL
THE NUMERATOR OF THE ZEROS ARE
0 2 -2 3
THE DENOMINATOR OF THE ZEROS ARE
1 3 1 4

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

5
TYPE THE COEFFICIENTS OF THE POLYNOMIAL IN FORMAT (10I8)
4 -72 64 112 -128 32
THIS POLYNOMIAL HAS NO RATIONAL ZEROS

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

7
TYPE THE COEFFICIENTS OF THE POLYNOMIAL IN FORMAT (10I8)
2 -1 -21 1 55 0 -36 0
ALL ZEROS OF THE POLYNOMIAL ARE RATIONAL
THE NUMERATOR OF THE ZEROS ARE
0 1 -1 -2 -2 3 3
THE DENOMINATOR OF THE ZEROS ARE
1 1 1 1 1 2 1

TYPE THE DEGREE OF THE POLYNOMIAL IN FORMAT (I2)
FOR STOPPING PLEASE TYPE : 0

-2
DATA ILLEGAL, KINDLY TYPE AGAIN

25
THE GIVEN NUMBER WILL CROSS THE DIMENSION, CHANGE THE DIMENSION

5.15 Given a square matrix, this program computes the integer eigenvalues of the matrix

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), P (ND), PR (26379), ITZ (ND)
COMMON /Al/PR
CALL PRIME (304678)

WRITE(6,(' TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)','
1/1X,'FOR STOPPING PLEASE TYPE : 0''))

READ(4, '(I2)') N
WRITE (6, '(1X, I2)') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
  WRITE(6, '('' DATA ILLEGAL, KINDLY TYPE AGAIN'')')
  GO TO 20
ELSEIF (N .GT. ND) THEN
  WRITE(6, '('' THE GIVEN NUMBER WILL CROSS THE DIMENSION,''
1 1X, ''CHANGE THE DIMENSION'')')
  STOP
ENDIF
WRITE (6, '('' TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)'')')
DO 33 I = 1, N
  READ (4, '(10I5)') (MAT (I, JJ), JJ = 1, N)
33 CONTINUE
DO 44 MM = 1, N
  WRITE (6, '(IX, 10I5)') (MAT (MM, KK), KK = 1, N)
44 CONTINUE
CALL CHPOLY (MAT, N, P)
WRITE (6, '('' THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL''
1/1X, ''STARTING FROM THE 2ND ARE GIVEN BELOW'')')
WRITE(6, '(10(1X, I7))') (P (I), I = 1, N)
M = N
CALL ZEROS (P, M, ITZ, NZ)
IF (NZ .EQ. 0) THEN
  WRITE(6, '('' THIS MATRIX HAS NO INTEGER EIGENVALUES''))
ELSEIF (NZ .LT. N) THEN
  NUMZ = N - NZ
  WRITE (6, '('' THIS MATRIX HAS '',I2,'' EIGENVALUES''
1 ,1X, ''WHICH ARE NOT INTEGERS'')') NUMZ
  WRITE(6, '('' THE '',I2,'' INTEGRAL EIGENVALUES OF THIS''
1 ,1X, ''MATRIX ARE''/1X,10I7)') NZ, (ITZ(KK), KK = 1, NZ)
ELSE
  WRITE(6, '('' ALL THE EIGENVALUES OF THIS MATRIX ARE''
1 ,1X, ''INTEGERS AND ARE''/1X,10I7)') (ITZ(KK), KK = 1, NZ)
ENDIF
GO TO 10
END

C PURPOSE- GIVEN A SQUARE MATRIX, THIS SUBROUTINE COMPUTES
C THE CHARACTERISTIC POLYNOMIAL OF THE MATRIX BY
C LEVERRIER-FADEEV METHOD
C INPUT PARAMETERS
C MAT: TWO DIMENSIONAL ARRAY, THE GIVEN MATRIX
C N : THE ORDER OF THE MATRIX
C OUTPUT PARAMETERS
C P : ONE DIMENSIONAL ARRAY, THE CHARACTERISTIC POLYNOMIAL
SUBROUTINE CHPOLY (MAT, N, P)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), B (ND, ND), P (ND)
DO 22 I = 1, N
   DO 11 J = 1, N
      B (I, J) = MAT (I, J)
   11 CONTINUE
22 CONTINUE
DO 55 K = 1, N
   SIGMA = 0
   DO 33 I = 1, N
      SIGMA = SIGMA - B (I, I)
   33 CONTINUE
   P (K) = SIGMA / K
   DO 44 I = 1, N
      B (I, I) = B (I, I) + P (K)
   44 CONTINUE
   CALL MULT (MAT, B, N)
55 CONTINUE
RETURN
END

C PURPOSE - TO MULTIPLY TWO SQUARE MATRICES
C INPUT PARAMETERS
C X : TWO DIMENSIONAL ARRAY, THE FIRST MATRIX
C Y : TWO DIMENSIONAL ARRAY, THE SECOND MATRIX
C N : THE ORDER OF THE MATRIX
C OUTPUT PARAMETER
C Y : TWO DIMENSIONAL ARRAY, THE PRODUCT OF THE MATRIX

SUBROUTINE MULT (X, Y, N)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION X (ND, ND), Y (ND, ND), Z (ND, ND)
DO 33 I = 1, N
   DO 22 J = 1, N
      SUM = 0
      DO 11 K = 1, N
         SUM = SUM + X (I, K) * Y (K, J)
      11 CONTINUE
      Z (I, J) = SUM
   22 CONTINUE
33 CONTINUE
DO 55 I = 1, N
   DO 44 J = 1, N
      Y (I, J) = Z (I, J)
   44 CONTINUE
55 CONTINUE
RETURN
END

TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
1

TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
  100
THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
   -100
ALL THE EIGENVALUES OF THIS MATRIX ARE INTEGERS AND ARE
   100
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
2

TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
  5   -3
    2   10
THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
   -15    56
ALL THE EIGENVALUES OF THIS MATRIX ARE INTEGERS AND ARE
     8    7
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
3

TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
  18   28   4
    6   18   16
   -20  -44  -26
THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
   -10    4   -40
THIS MATRIX HAS 2 EIGENVALUES WHICH ARE NOT INTEGERS
THE 1 INTEGRAL EIGENVALUES OF THIS MATRIX ARE
   10
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
4

TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
  3   6   2   5
    6   3   2   5
    2   2  10   2
    5   5   2   4
THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
-20  35  440  384
ALL THE EIGENVALUES OF THIS MATRIX ARE INTEGERS AND ARE
   -1  8  16  -3
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
   5
TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
   7  4  3  -2  -3
   0  5  0  0  0
-2 -4  2  2  3
   2  4  3  3  -3
-2 -4 -3  2  8

THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
-25  250  -1250  3125  -3125
ALL THE EIGENVALUES OF THIS MATRIX ARE INTEGERS AND ARE
   5  5  5  5  5
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
   -5
DATA ILLEGAL, KINDLY TYPE AGAIN
   6
TYPE THE ENTRIES OF THE MATRIX IN FORMAT (10I5)
   0  0  3  4  5  2
   0  0  4  5  1  1
   0  0  1  3  4  4
   4  0  2  4  8  5
   5  0  7  1  3  3
   6  7  1  2  3  2

THE COEFFICIENTS OF THE CHARACTERISTIC POLYNOMIAL
STARTING FROM THE 2ND ARE GIVEN BELOW
-10  -90  -570  -997  -1851  -10080
THIS MATRIX HAS NO INTEGRAL EIGENVALUES
TYPE THE ORDER OF THE MATRIX IN FORMAT(I2)
FOR STOPPING PLEASE TYPE : 0
   0

5.16 This program computes the eigenvectors of a given matrix

IMPLICIT INTEGER*4 (A-Z)
PARAMETER (ND = 15)
DIMENSION B (ND, ND), EV (ND), U (ND), V (ND)
COMMON A (ND, 16), ADEN (ND, 16), X (ND, ND), XDEN (ND, ND),
   IM, N, P, IND
10 WRITE (6, '(" TYPE THE ORDER OF THE MATRIX AND THE NUMBER",
1/1X,"OF DISTINCT EIGENVALUES IN FORMAT (2I2)")')
READ (4, '(2I2)') N, NEQ
WRITE (6, '(1X, 2I2)') N, NEQ
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
    WRITE (6, '('' DATA ILLEGAL, KINDLY TYPE AGAIN'')')
    GO TO 10
ELSEIF (N .GT. ND) THEN
    WRITE (6, '('' ORDER OF THE MATRIX, TOO LARGE KINDLY'')')
    GO TO 10
ENDIF
M = N
NS = 0
NP1 = N + 1
WRITE (6,'('' TYPE THE COEFFICIENTS OF THE MATRIX IN''
1,1X,''FORMAT (15I4)'')')
DO 22 I = 1, N
    READ (4, '(15I4)') (B (I, J), J = 1, N)
22 CONTINUE
DO 33 I = 1, N
    WRITE (6, '(1X, 15I4)') (B (I, J), J = 1, N)
33 CONTINUE
WRITE (6,'('' TYPE THE DISTINCT EIGENVALUES IN FORMAT (15I4)'')')
READ (4, '(15I4)') (EV (I), I = 1, NEQ)
WRITE (6, '(1X, 15I4)') (EV (I), I = 1, NEQ)
40 NS = NS + 1
IF (NS .GT. NEQ) GO TO 10
CALL FECEIV (B, EV, N, NS, A)
DO 77 I = 1, N
    DO 66 J = 1, N + 1
        ADEN (I, J) = 1
66 CONTINUE
77 CONTINUE
CALL SOLVE
IF (IND .EQ. 1) THEN
    WRITE (6, '('' THERE IS NO SOLUTION'')')
ELSE
    WRITE (6, '('' THE CORRESPONDING EIGENVECTORS ARE''')
    P = P + 1
    DO 122 K = 2, P
        U (J) = XDEN (J, K)
        V (J) = X (J, K)
122 CONTINUE
CALL MULLCM (U, V, N)
CALL REMCOM (V, N)
DO 111 I = 1, N
   IF (V (I) .GT. 0) GO TO 120
   IF (V (I) .LT. 0) THEN
      DO 99 J = I, N
         V (J) = - V (J)
      99 CONTINUE
   END IF
111 CONTINUE
120 WRITE (6, '(IX, 1514)') (V (I), I = 1, N)
122 CONTINUE

C PURPOSE- TO FORM AN EQUATION AX = LX
C INPUT PARAMETERS
C B : TWO-DIMENSIONAL ARRAY, THE GIVEN MATRIX
C EV : LINEAR ARRAY, THE EIGENVALUES
C N : THE ORDER OF THE MATRIX
C NS : THE NUMBER OF THE EIGENVALUE
C OUTPUT PARAMETER
C A : TWO-DIMENSIONAL ARRAY, THE COEFFICIENTS OF THE MATRIX
C EQUATIONS

SUBROUTINE FECEIV (B, EV, N, NS, A)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 15)
DIMENSION B (ND, ND), A (ND, 16), EV (ND)
DO 22 I = 1, N
   DO 11 J = 1, N
      IF (I .EQ. J) THEN
         A (I, J) = B (I, J) - EV (NS)
      ELSE
         A (I, J) = B (I, J)
      ENDIF
11 CONTINUE
22 CONTINUE
DO 33 I = 1, N
   A (I, N + 1) = 0
33 CONTINUE
RETURN
END

C PURPOSE - TO SOLVE A SYSTEM OF M LINEAR EQUATIONS IN N UNKNOWNS
C INPUT PARAMETERS
C A : TWO-DIMENSIONAL ARRAY, NUMERATOR OF THE COEFFICIENTS
C ADEN : TWO-DIMENSIONAL ARRAY, DENOMINATOR OF THE COEFFICIENTS
C M : THE NUMBER OF EQUATIONS
SUBROUTINE SOLVE
IMPLICIT INTEGER*4 (A-Z)
PARAMETER (ND = 15)
COMMON A (ND, 16), ADEN (ND, 16), X (ND, ND), XDEN (ND, ND)
DIMENSION LOC (ND), ROW (ND), COL (ND), T (ND)
DO 22 II=1, ND
   LOC (II) = 0
22 CONTINUE
NPl = N + 1
RANK = 0
P = 0
C Test consistency condition by determining rank
DO 88 I = 1, NPl
   DO 33 L = 1, M
      IF (LOC (L) .NE. 1) THEN
         IF (A (L, I) .NE. 0) GO TO 40
      ENDIF
33 CONTINUE
   IF (I .EQ. NPl) GO TO 120
   P = P + 1
   T (P) = I
   GO TO 88
40 IF (I .EQ. NPl) THEN
   C The system is not consistent
   IND = 1
   RETURN
ENDIF
RANK = RANK + 1
ROW (RANK) = L
COL (RANK) = I
LOC (L) = 1
CALL RECIP (A(L,I), ADEN(L,I), RECNUM, RECDEN)
IP1 = I + 1
DO 55 K = IP1, NPl
   IF (A (L, K) .NE. 0) THEN
      CALL MULTN (A (L, K), ADEN (L, K), RECNUM, RECDEN,
                   1
                   A (L, K), ADEN (L, K))
   ENDF
The system is consistent, find the solution

IF (RANK .EQ. M) GO TO 90

DO 77 J = L+1, M
   IF (LOC (J) .NE. 1 .AND. A(J, I) .NE. 0) THEN
      DO 66 K = IP1, NP1
         IF (A (L, K) .NE. 0) THEN
            CALL MULTN (A (J, I), ADEN (J, I), A (L, K),
                        ADEN (L,K), TEMP1, TEMP2)
            CALL SUBTN (A (J,K), ADEN(J,K), TEMP1, TEMP2,
                        A (J, K), ADEN (J, K))
         END IF
      END DO 66
   END IF
END DO 77

88 CONTINUE

90 IF (I .NE. N) THEN
   G Initialize the identity matrix
   DO 111 INDEX = IP1, N
      P = P + 1
      T (P) = INDEX
   END DO 111
   END IF

120 IND = 0

IF (RANK .EQ. 0) THEN
   DO 144 I = 1, N
      X (I, I) = 1
      XDEN (I, I) = 1
      IF (N .NE. 1) THEN
         DO 133 J = 1, N
            IF (J .NE. I) THEN
               X (J, I) = 0
               XDEN (J, I) = 1
            END IF
         END DO 133
      END IF
   END DO 144
END IF

DO 166 I = 1, RANK
   NMI = RANK + 1 - I
   L = ROW (NMI)
   Y = COL (NMI)
   IF (I .NE. 1) THEN
      IM1 = I - 1
      DO 155 K = 1, IM1
         CALL MULTN (A (I, K), ADEN (I, K), A (L, K),
                      ADEN (L,K), TEMP1, TEMP2)
         CALL SUBTN (A (I,K), ADEN(I,K), TEMP1, TEMP2,
                      A (I, K), ADEN (I, K))
      END DO 155
   END IF
END DO 166

RETURN
Z = COL (NMI + K)
IF (A (L, Z) .NE. 0 .AND. X(Z, 1) .NE. 0) THEN
  CALL MULTN (A (L, Z), ADEN (L, Z), X (Z, 1),
  XDEN (Z, 1), TEMP1, TEMP2)
  CALL SUBTN (A (L, NP1), ADEN (L, NP1), TEMP1,
  TEMP2, A (L, NP1), ADEN (L, NP1))
ENDIF

155 CONTINUE

1 ENDIF

166 CONTINUE

155 CONTINUE

C Test for the existence of unique solution
IF (P .EQ. 0) RETURN
DO 177 I = 1, P
  X (T (I), 1) = 0
  XDEN (T (I), 1) = 1
177 CONTINUE
DO 222 INDEX = 1, P
  INDP1 = INDEX + 1
  R = T (INDEX)
  DO 199 I = 1, RANK
    NMI = RANK + 1 - I
    L = ROW (NMI)
    Y = COL (NMI)
    IF (Y .GE. R) THEN
      X (Y, INDP1) = 0
      XDEN (Y, INDP1) = 1
      GO TO 199
    ENDIF
    IF (I .NE. 1) THEN
      IM1 = I - 1
      DO 188 K = 1, IM1
        Z = COL (NMI + K)
        IF (A (L, Z) .NE. 0 .AND. X(Z, INDP1) .NE. 0) THEN
          CALL MULTN (A (L, Z), ADEN (L, Z), X (Z, INDP1),
          XDEN (Z, INDP1), TEMP1, TEMP2)
          CALL SUBTN (A (L, R), ADEN (L, R), -TEMP1, TEMP2,
          TEMP2, A (L, NP1), ADEN (L, NP1))
        ENDIF
      188 CONTINUE
    ENDIF
  X (Y, INDP1) = - A (L, R)
  XDEN (Y, INDP1) = ADEN (L, R)
199 CONTINUE

199 CONTINUE

X (R, INDP1) = 1
XDEN (R, INDPl) = 1
IF (P .NE. 1) THEN
  DO 211 J = 1, P
    IF (J .NE. INDEX) THEN
      X (T(J), INDPl) = 0
      XDEN (T(J), INDPl) = 1
    ENDIF
  211 CONTINUE
ENDIF
  222 CONTINUE
RETURN
END

C PURPOSE - TO REDUCE A VECTOR WITH RATIONAL COEFFICIENTS
C TO A VECTOR WITH INTEGER COEFFICIENTS
C INPUT PARAMETERS
C U : LINEAR ARRAY, THE RATIONAL VECTOR
C NCF : THE NUMBER OF COMPONENTS OF THE VECTOR
C OUTPUT PARAMETER
C V : LINEAR ARRAY, THE INTEGER VECTOR
SUBROUTINE MULLCM (U, V, NCF)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION U (NCF), V (NCF)
ICOHCF = IABS (U (1))
LCM = U (1)
DO 22 I = 2, NCF
  ICOHCF = HCF (ICOHCF, U (I))
  LCM = IABS (LCM * U (I) / ICOHCF)
22 CONTINUE
DO 33 I = 1, NCF
  V (I) = V (I) * LCM / U (I)
33 CONTINUE
RETURN
END

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF DISTINCT EIGENVALUES IN FORMAT (2I2)
  2 1

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
  8 9
  -4 -4

TYPE THE DISTINCT EIGENVALUES IN FORMAT (15I4)
  2
THE CORRESPONDING EIGENVECTORS ARE
  3 -2

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF DISTINCT EIGENVALUES IN FORMAT (2I2)
  3 2
TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
   6  -2   2
   -2   3  -1
    2  -1   3
TYPE THE DISTINCT EIGENVALUES IN FORMAT (15I4)
     2   8
THE CORRESPONDING EIGENVECTORS ARE
   1   2   0
   1   0  -2
THE CORRESPONDING EIGENVECTORS ARE
   2  -1   1
TYPE THE ORDER OF THE MATRIX AND THE NUMBER OF DISTINCT EIGENVALUES IN FORMAT (2I2)
    4   4
TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
   10   3   5   2
    8   3   4   5
    8   2   1   9
    8   2   1   9
TYPE THE DISTINCT EIGENVALUES IN FORMAT (15I4)
   20   2   1   0
THE CORRESPONDING EIGENVECTORS ARE
   1   1   1   1
THE CORRESPONDING EIGENVECTORS ARE
   5  -4  -4  -4
THE CORRESPONDING EIGENVECTORS ARE
  13 -25  -6  -6
THE CORRESPONDING EIGENVECTORS ARE
  1  51 -29  -9
TYPE THE ORDER OF THE MATRIX AND THE NUMBER OF DISTINCT EIGENVALUES IN FORMAT (2I2)
   21   1
ORDER OF THE MATRIX, TOO LARGE KINDLY TYPE AGAIN
TYPE THE ORDER OF THE MATRIX AND THE NUMBER OF DISTINCT EIGENVALUES IN FORMAT (2I2)
    5   1
TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
   7    4    3   -2  -3
    0    5    0    0    0
   -2  -4    2    2    3
    2    4    3    3  -3
   -2  -4  -3    2    8
TYPE THE DISTINCT EIGENVALUES IN FORMAT (15I4)
     5
THE CORRESPONDING EIGENVECTORS ARE
   2  -1   0   0   0
   3   0  -2   0   0
   1   0   0   1   0
   3   0   0   0   2
TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF DISTINCT EIGENVALUES IN FORMAT (2I2)
0 0

5.17 The program computes the Jordan canonical
form of a given matrix

IMPLICIT INTEGER*4 (A-Z)
PARAMETER (ND = 15)
DIMENSION B (ND, ND), U (ND), V (ND)
COMMON A (ND, 16), ADEN (ND, 16), X (ND, ND), XDEN (ND, ND),
1M, N, P, IND
10 WRITE (6, \"('' TYPE THE ORDER OF THE MATRIX AND THE NUMBER\',
1/1X,\'(OF EIGENVECTOR IN FORMAT (2I2)')\")'
READ (4, \'(2I2)\') N, NEQ
WRITE (6, \'(1X, 2I2)\') N, NEQ
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
   WRITE (6, \"DATA ILLEGAL, KINDLY TYPE AGAIN\")
   GO TO 10
ELSEIF (N .GT. ND) THEN
   WRITE (6, \"ORDER OF THE MATRIX, TOO LARGE KINDLY\'
1 ,1X,\'TYPE AGAIN\")
   GO TO 10
ENDIF
M = N
NS = 0
NP1 = N + 1
WRITE (6, \"(OF THE COEFFICIENTS OF THE MATRIX IN\',
1,1X,\'FORMAT (15I4)')\")
DO 22 I = 1, N
   READ (4, \'(20I4)\') (B (I, J), J = 1, N)
22 CONTINUE
DO 33 I = 1, N
   WRITE (6, \'(1X, 20I4)\') (B (I, J), J = 1, N)
33 CONTINUE
NS = NS + 1
IF (NS .GT. NEQ) GO TO 10
WRITE (6, \"(TYPE THE EIGENVALUE IN FORMAT (I4)')\")
READ (4, \'(I4)\') EV
WRITE (6, \'(1X, I4)\') EV
WRITE (6, \"(TYPE THE CORRESPONDING EIGENVECTOR\',
1,1X,\'IN FORMAT (15I4)')\")
READ (4, \'(15I4)\') (A (I, NP1), I = 1, N)
WRITE (6, \'(1X, 15I4)\') (A (I, NP1), I = 1, N)
DO 55 I = 1, N
    ADEN (I, N + 1) = 1
55  CONTINUE
60  CALL FEJORD (B, N, EV, A)
DO 88 I = 1, N
    DO 77 J = 1, N + 1
        ADEN (I, J) = 1
77  CONTINUE
88  CONTINUE
M = N
CALL SOLVE
IF (IND .EQ. 1) THEN
    WRITE (6, '('' THERE IS NO SOLUTION'')')
    GO TO 40
ELSE
    WRITE (6, '('' THE PRINCIPAL VECTOR IS'')')
    WRITE (6, '(IX, 10(14,"/",12))') (X (I, 1), XDEN (I, 1), I = 1, N)
END IF
DO 99 I = 1, N
    A (I, N+1) = X (I, 1)
    ADEN (I, N+1) = XDEN (I, 1)
99  CONTINUE
GO TO 60
END

C PURPOSE- TO FORM AN EQUATION AX = LX
C INPUT PARAMETERS
C B : TWO DIMENSIONAL ARRAY, THE GIVEN MATRIX
C EV : THE EIGENVALUE
C N : THE ORDER OF THE MATRIX
C OUTPUT PARAMETER
C A : TWO DIMENSIONAL ARRAY, THE COEFFICIENTS OF THE MATRIX
C EQUATIONS
SUBROUTINE FEJORD (B, N, EV, A)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 15)
DIMENSION B (ND, ND), A (ND, 15)
DO 22 I = 1, N
    DO 11 J = 1, N
        IF (I .EQ. J) THEN
            A (I, J) = B (I, J) - EV
        ELSE
            A (I, J) = B (I, J)
        ENDIF
11  CONTINUE
22  CONTINUE
RETURN
END

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
3 3

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
-7 -21 -15
  6  16  10
-3  -7  -3

TYPE THE EIGENVALUE IN FORMAT (I4)
  2

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
-7  3  0
THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
  2

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
-5  0  3
THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
  2

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
  3  -2  1
THE PRINCIPAL VECTOR IS
  -1/ 3  0/ 1  0/ 1
THERE IS NO SOLUTION

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
4 2

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
  2  -1  -2  1
-4  -3  2  1
  17  -3  -10  2
-6  1  2  -5

TYPE THE EIGENVALUE IN FORMAT (I4)
-4

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
  0  -2  1  0
THE PRINCIPAL VECTOR IS
  -1/ 1  -6/ 1  0/ 1  0/ 1
THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
-4

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
-1  -5  0  1
THE PRINCIPAL VECTOR IS
-3/ 1  -17/ 1  0/ 1  0/ 1
THERE IS NO SOLUTION
TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
   4 3

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
   10  -1  -2   0
   -8   7   3  -1
   18  -4  -2   1
   -5   1   2   5

TYPE THE EIGENVALUE IN FORMAT (I4)
   5

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
   1   1   2   0

THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
   5

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
   1   5   0   2

THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
   5

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
   1  -1   3  -1

THE PRINCIPAL VECTOR IS
   1/2  3/2 0/1 0/1

THERE IS NO SOLUTION

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
   5 3

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
   5   8   2   4   2
   2   7   4   2   4
   3   6   6   6   9
   1   2   1   3   4
   2   4   2   4   9

TYPE THE EIGENVALUE IN FORMAT (I4)
   3

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
  -60   8  12   4   8

THE PRINCIPAL VECTOR IS
  38/1 -17/1 0/1 0/1 0/1

THE PRINCIPAL VECTOR IS
  -21/2 19/2 -17/2 0/1 0/1

THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
   20

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
   4   4   6   2   4

THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
1

199

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
84 -30 12 -34 8
THERE IS NO SOLUTION

TYPES THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
4 2

TYPE THE COEFFICIENTS OF THE MATRIX IN FORMAT (15I4)
9 -1 -2 0
-8 6 3 -1
18 -4 -3 1
-5 1 2 4

TYPE THE EIGENVALUE IN FORMAT (I4)
4

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
1 1 2 0
THERE IS NO SOLUTION

TYPE THE EIGENVALUE IN FORMAT (I4)
4

TYPE THE CORRESPONDING EIGENVECTOR IN FORMAT (15I4)
1 5 0 2
THERE IS NO SOLUTION

TYPE THE ORDER OF THE MATRIX AND THE NUMBER
OF EIGENVECTOR IN FORMAT (2I2)
0 0

5.18 Program to find the entries of a matrix with
given determinant

PARAMETER (ND = 10)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION MAT (ND, ND)
OPEN (4, FILE = 'ZGDET.IN', STATUS = 'OLD')

WRITE (6, ' (" TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)"')
READ (4, '(I2)') N
WRITE (6, '(IX, I2)') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
  WRITE (6, ' (" ORDER OF THE MATRIX CANNOT BE NEGATIVE"')
  GO TO 10
ELSEIF (N .GT. ND) THEN
  WRITE (6, ' (" DESIRED MATRIX TOO LARGE, TYPE",IX,
1   " A SMALLER NUMBER" /", AS THE ORDER OF THE MATRIX")
  GO TO 10
ELSEIF (N .EQ. 1) THEN
  IO = 1
ELSE
IO = 2
ENDIF
WRITE (6, '(" TYPE THE INTEGER ENTRIES OF THE MATRIX"'
1,1X,'"OF ORDER",1X,I2,1X,'"WITH DETERMINANT 1")') IO
20 DO 33 I = 1, IO
   READ (4, '(1016)') (MAT (I, J), J = 1, IO)
33 CONTINUE
DO 44 I = 1, IO
   WRITE (6, '(1X,1016)') (MAT (I, J), J = 1, IO)
44 CONTINUE
IND = 1
50 CALL MUD (MAT, IO, IND)
IF (IND .EQ. 2) THEN
   WRITE (6, '(" ELEMENTS ARE WRONG, KINDLY TYPE AGAIN")')
   GO TO 20
ENDIF
IF (IO .LT. N) THEN
   IO = IO + 1
   WRITE (6,'(" TYPE ANY",13,IX," INTEGERS IN")'
1 ,1X,'"FORMAT (1016)")') IO - 1
   READ (4, '(1016)') (MAT (IO, LL-1), LL = 2, IO)
   WRITE (6, '(1X,1016)') (MAT (IO, LL-1), LL = 2, IO)
   WRITE (6, '(" AGAIN TYPE ANY",13,IX," INTEGERS IN")'
1 ,1X,'"FORMAT (1016)")') IO - 1
   READ (4, '(1016)') (MAT (LL-1, IO), LL = 2, IO)
   WRITE (6, '(1X,1016)') (MAT (LL-1, IO), LL = 2, IO)
   GO TO 50
ENDIF
WRITE (6, '(" TYPE THE DETERMINANT OF THE MATRIX"
1,1X,"IN FORMAT (16)")')
READ (4, '(16)') IDETR
WRITE (6, '(16)') IDETR
C Special treatment for order 1 and 2
IF (N .LE. 2) THEN
   DO 55 I = 1, N
      MAT (1, I) = IDETR * MAT (1, I)
   55 CONTINUE
ELSE
   MAT (N, N) = MAT (N, N) + (IDETR - 1)
ENDIF
WRITE (6, '(" REQUIRED MATRIX IS")')
DO 66 I = 1, N
   WRITE (6, '(1X,1016)') (MAT (I, J), J = 1, N)
66 CONTINUE
GO TO 10
END
C PURPOSE :- TO GENERATE A MATRIX WITH UNIT DETERMINANT AND INTEGER
C ENTRIES
C INPUT PARAMETERS
C MAT : TWO DIMENSIONAL ARRAY, NON DIAGONAL ENTRIES OF THE MATRIX
C TO BE GENERATED
C N : ORDER OF THE MATRIX
C IND = 1 INITIALLY
C = 2 MATRIX WITH UNIT DETERMINANT IS NOT POSSIBLE
C OUTPUT PARAMETER
C MAT : TWO DIMENSIONAL ARRAY, INTEGER MATRIX WITH UNIT DETERMINANT

SUBROUTINE MUD (MAT, N, IND)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), NMAT (ND, ND)
IF (N .LE. 2) THEN
  C Special treatment for orders 1 and 2
  IF (N .EQ. 1) THEN
    IF (MAT (1, 1) .NE. 1) IND = 2
  ELSE
    IF (MAT(1, 1)*MAT (2,2) - MAT(l,2) * MAT(2,l).NE. 1) IND=2
  ENDIF
  RETURN
ELSE
  C Calculate the entry MAT (N, N), N>=3
  MN1 = N - 1
  NEW = 1
  DO 44 JJ = 1, MN1
    IF (JJ .EQ. 1) THEN
      C The minor of an element
      DO 22 I = 1, MN1
        DO 11 K = 1, MN1
          NMAT (I, K) = MAT (I, K + 1)
        11 CONTINUE
      22 CONTINUE
    ENDIF
    C Calculation of the new entry
    NEW = NEW + (-1)**(JJ+N+1)*MAT (N, JJ) * LDET (NMAT,MN1)
    IF (JJ .NE. MN1) THEN
      DO 33 I = 1, MN1
        NMAT (I, JJ) = MAT (I, JJ)
      33 CONTINUE
    ENDIF
  44 CONTINUE
  C The required element of the matrix
  MAT (N, N) = NEW
ENDIF
RETURN
**C**

**PURPOSE** - TO EVALUATE DETERMINANT USING GAUSSIAN ELIMINATION METHOD

**C**

**INPUT PARAMETERS**

**C**

**A** : TWO DIMENSIONAL ARRAY, ELEMENTS OF THE DETERMINANT

**C**

**N** : ORDER OF THE DETERMINANT

**FUNCTION LDET (A, N)**

**IMPLICIT INTEGER*4 (A - Z)**

**PARAMETER (ND = 10)**

**INTEGER A (ND, ND), COPYA (ND, ND), COPYD (ND, ND)**

**C Initialize the arrays**

**DO 22 MM=1,N**

**DO 11 NN=1,N**

**COPYA (MM, NN) = A (MM, NN)**

**COPYD (MM, NN) = 1**

**11 CONTINUE**

**22 CONTINUE**

**C Special treatment for order 1**

**IF (N .NE. 1) THEN**

**DO 88 I = 1, N-1**

**IPl = I + 1**

**IF (COPYA (1,1) .Eq. 0) THEN**

**C Search for non-zero entry in the column**

**DO 33 J = IPl, N**

**IF (COPYA (J,I) .NE. 0) GO TO 40**

**33 CONTINUE**

**C The above condition is true when det is 0**

**LDET = 0**

**RETURN**

**C Interchange the rows**

**DO 55 K = I, N**

**EXTRA =COPYA (I, K)**

**COPYA (I, K) = COPYA (J, K)**

**COPYA (J, K) = - EXTRA**

**EXTRA = COPYD (I, K)**

**COPYD (I, K) = COPYD (J, K)**

**COPYD (J, K) = EXTRA**

**55 CONTINUE**

**ENDIF**

**C Carry out the process of elimination**

**CALL RECIP (COPYA (I, I),COPYD (I, I), RECNUM, RECDEN)**

**DO 77 J = IP1, N**

**IF (COPYA (J, I) .NE. 0) THEN**

**CALL MULTN (COPYA (J, I),COPYD (J, I),RECNUM,RECDEN, 1 FACNUM,FACDEN)**

**77 CONTINUE**

**END**
C Replace a (j,k) by a (j,k)+a (j,i)/a (i,i)*a (i,k)
DO 66 K = IP1, N
   IF (COPYA (I, K) .NE. 0) THEN
      CALL MULTN (FACNUM, FACDEN,COPYA (I, K),
      COPYD (I, K), TEMP1, TEMP2)
      CALL SUBTN (COPYA (J, K),COPYD (J,K),
      TEMP1, TEMP2, COPYA (J,K), COPYD (J,K))
   END IF
66 CONTINUE
END IF
77 CONTINUE
88 CONTINUE
ENDIF
NUM = 1
DEN = 1
C Multiply all the diagonal elements
DO 99 I = 1, N
   CALL MULTN (NUM, DEN, COPYA (I, I),COPYD (I, I), NUM, DEN)
99 CONTINUE
IF (DEN .EQ. 1) THEN
   LDET = NUM
ELSE
   WRITE (*, *) ' THERE IS SOME MISTAKE IN THE CALCULATIONS '
ENDIF
RETURN
END

TYPE THE ORDER OF THE MATRIX IN FORMAT (12)
1
TYPE THE INTEGER ENTRIES OF THE MATRIX OF ORDER 1 WITH DETERMINANT 1
1
TYPE THE DETERMINANT OF THE MATRIX IN FORMAT (16)
452046
REQUIRED MATRIX IS
452046
TYPE THE ORDER OF THE MATRIX IN FORMAT (12)
2
TYPE THE INTEGER ENTRIES OF THE MATRIX OF ORDER 2 WITH DETERMINANT 1
1 -1
-3 4
TYPE THE DETERMINANT OF THE MATRIX IN FORMAT (16)
1000
REQUIRED MATRIX IS
1000 -1000
-3 4
TYPE THE ORDER OF THE MATRIX IN FORMAT (12)
25
DESIRED MATRIX TOO LARGE, TYPE A SMALLER NUMBER

AS THE ORDER OF THE MATRIX

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)

5

TYPE THE INTEGER ENTRIES OF THE MATRIX OF ORDER 2 WITH DETERMINANT 1

\[
\begin{pmatrix}
24 & 1 \\
1 & 1
\end{pmatrix}
\]

ELEMENTS ARE WRONG, KINDLY TYPE AGAIN

\[
\begin{pmatrix}
-2 & 1 \\
1 & -1
\end{pmatrix}
\]

TYPE ANY 2 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
0 & -1
\end{pmatrix}
\]

AGAIN TYPE ANY 2 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
-2 & 1
\end{pmatrix}
\]

TYPE ANY 3 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
1 & 4 & -1
\end{pmatrix}
\]

AGAIN TYPE ANY 3 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
2 & -2 & -2
\end{pmatrix}
\]

TYPE ANY 4 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
1 & 2 & 3 & -1
\end{pmatrix}
\]

AGAIN TYPE ANY 4 INTEGERS IN FORMAT (10I6)

\[
\begin{pmatrix}
-4 & 2 & 0 & 2
\end{pmatrix}
\]

TYPE THE DETERMINANT OF THE MATRIX IN FORMAT (I6)

105

REQUIRED MATRIX IS

\[
\begin{pmatrix}
-2 & 1 & -4 & 2 & -4 \\
1 & -1 & 2 & -2 & 2 \\
0 & -1 & 1 & -2 & 0 \\
1 & 4 & -1 & 9 & 2 \\
1 & 2 & 3 & -1 & 107
\end{pmatrix}
\]

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)

0

5.19 Program for generation of integer matrices (diagonalisable) with pre-assigned integer spectra

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), INVM (ND, ND), IMT (ND, ND), NMAT (ND, ND)
1,EV (ND)

10 WRITE (6,
'(" TYPE THE ORDER OF THE MATRIX")''
1,1X,"IN FORMAT (I2)")')
READ (4, '(I2') N
WRITE (6, '(1X, I2') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
    WRITE (6,'(" ORDER OF THE MATRIX CANNOT BE NEGATIVE")')
END IF
READ (4, '(12)') N
WRITE (6, '(1X, 12)') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
    WRITE (6,'(" ORDER OF THE MATRIX CANNOT BE NEGATIVE")')
END IF

EXAMPLE:

ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
24 & 1 \\
1 & 1
\end{pmatrix}
\]

AGAIN ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
0 & -1
\end{pmatrix}
\]

AGAIN ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
-2 & 1
\end{pmatrix}
\]

ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
1 & 4 & -1 \\
2 & -2 & -2 \\
1 & 2 & 3 & -1
\end{pmatrix}
\]

ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
-4 & 2 & 0 & 2
\end{pmatrix}
\]

ORDER OF THE MATRIX IS

\[
\begin{pmatrix}
-2 & 1 & -4 & 2 & -4 \\
1 & -1 & 2 & -2 & 2 \\
0 & -1 & 1 & -2 & 0 \\
1 & 4 & -1 & 9 & 2 \\
1 & 2 & 3 & -1 & 107
\end{pmatrix}
\]
GO TO 10
ELSEIF (N .GT. ND) THEN
   WRITE (6, '(" ORDER OF THE DESIRED MATRIX TOO LARGE, TYPE'"
   1 ,IX,'"A SMALLER NUMBER'"/" AS THE ORDER OF THE MATRIX")')
   GO TO 10
ENDIF
DO 44 I = 1, N
   READ (4, '(1016)') (MAT (I, J), J = 1, N)
44 CONTINUE
WRITE (6, '(" TYPE THE INTEGER EIGENVALUES IN FORMAT (1016)")')
READ (4, '(1016)') (EV (I), I = 1, N)
WRITE (6, '(1X, 1016)') (EV (I), I = 1, N)
WRITE (6, '(" TYPE A MATRIX WITH UNIT DETERMINANT IN"'
1,1X,"FORMAT (1016)")')
DO 55 I = 1, N
   WRITE (6, '(1X, 1016)') (MAT (I, J), J = 1, N)
55 CONTINUE
CALL MATINV (MAT, INVM, N, IND)
C IND = 1, if the matrix is singular
IF (IND .EQ. 1) GO TO 10
WRITE (6, '(" INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS")')
DO 66 I = 1, N
   WRITE (6, '(1X, 1017)') (INVM (I,J), J = 1, N)
66 CONTINUE
CALL DMM (MAT, EV, NMAT, N)
CALL MATMUL (NMAT, INVM, IMT, N)
WRITE (6, '(" THE REQUIRED MATRIX IS")')
DO 77 I = 1, N
   WRITE (6, '(1X, 1017)') (IMT (I, J), J = 1, N)
77 CONTINUE
GO TO 10
END

C PURPOSE - TO FIND THE PRODUCT OF A MATRIX WITH A DIAGONAL
C MATRIX
C INPUT PARAMETERS
C EV : ONE DIMENSIONAL ARRAY, CONTAINS THE DIAGONAL ELEMENTS
C KMAT: TWO DIMENSIONAL ARRAY, CONTAINS THE MATRIX
C N : THE ORDER OF THE MATRIX
C OUTPUT PARAMETERS
C NMAT: TWO DIMENSIONAL ARRAY, CONTAINS THE PRODUCT MATRIX

SUBROUTINE DMM (KMAT, EV, NMAT, N)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION KMAT (ND, ND), EV (N), NMAT (ND, ND)
DO 22 J = 1, N
   DO 11 I = 1, N
      READ (4, '(1016)') (KMAT (I, J), J = 1, N)
11 CONTINUE
READ (4, '(1016)') (EV (I), I = 1, N)
WRITE (6, '(1X, 1016)') (EV (I), I = 1, N)
WRITE (6, '(IX, 1016)') (KMAT (I, J), J = 1, N)
22 CONTINUE
DO 33 I = 1, N
   WRITE (6, '(IX, 1016)') (KMAT (I, J), J = 1, N)
33 CONTINUE
WRITE (6, '(" TYPE A MATRIX WITH UNIT DETERMINANT IN")')
DO 44 I = 1, N
   WRITE (6, '(IX, 1016)') (KMAT (I, J), J = 1, N)
44 CONTINUE
CALL MATINV (KMAT, INVM, N, IND)
C IND = 1, if the matrix is singular
IF (IND .EQ. 1) GO TO 10
WRITE (6, '(" INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS")')
DO 55 I = 1, N
   WRITE (6, '(1X, 1016)') (INVM (I,J), J = 1, N)
55 CONTINUE
CALL DMM (KMAT, EV, NMAT, N)
CALL MATMUL (NMAT, INVM, IMT, N)
WRITE (6, '(" THE REQUIRED MATRIX IS")')
DO 66 I = 1, N
   WRITE (6, '(1X, 1016)') (INVM (I,J), J = 1, N)
66 CONTINUE
GO TO 10
END
NMAT (J, I) = KMAT (J, I) * EV (I)
11 CONTINUE
22 CONTINUE
RETURN
END

C PURPOSE - TO MULTIPLY TWO SQUARE MATRICES
C INPUT PARAMETERS
C F : TWO DIMENSIONAL ARRAY, CONTAINS THE FIRST MATRIX
C G : TWO DIMENSIONAL ARRAY, CONTAINS THE SECOND MATRIX
C K : ORDER OF THE MATRIX
C OUTPUT PARAMETERS
C H : TWO DIMENSIONAL ARRAY, CONTAINS THE PRODUCT MATRIX

SUBROUTINE MATMUL (F, G, H, K)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION F (ND, ND), G (ND, ND), H (ND, ND)
DO 33 I = 1, K
   DO 22 J = 1, K
      NSUM = 0
      DO 11 L = 1, K
         NSUM = NSUM + F (I, L) * G (L, J)
      CONTINUE
   CONTINUE
33 CONTINUE
RETURN
END

C PURPOSE - TO FIND THE INVERSE OF A MATRIX
C INPUT PARAMETERS
C A1 : TWO DIMENSIONAL ARRAY, CONTAINS THE MATRIX
C N : ORDER OF THE MATRIX
C OUTPUT PARAMETERS
C A : TWO DIMENSIONAL ARRAY, CONTAINS THE INVERSE MATRIX
C IND = 0 INITIALLY
C = 1 THE MATRIX IS SINGULAR

SUBROUTINE MATINV (A1, A, N, IND)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION A1 (ND, ND), ADEN (ND, ND), INT1 (ND), INT2 (ND), A (ND, ND)
C Initialize the arrays
DO 22 I = 1, N
   DO 11 J = 1, N
      A (I, J) = A1 (I, J)
      ADEN (I, J) = 1
11 CONTINUE
L = 0
DO 99 I = 1, N

C Search for the pivot
IF (A(I, I) .EQ. 0) THEN
  IF (I .NE. N) THEN
    DO 33 J = I+1, N
      IF (A(J, I) .NE. 0) GO TO 40
    33 CONTINUE
  ENDIF
  C The matrix is singular
  IND = 1
  RETURN
ENDIF

40 L = L + 1
INT1(L) = I
INT2(L) = J

C Interchange the rows
DO 44 K = 1, N
  EXTRA = A(I, K)
  A(I, K) = A(J, K)
  A(J, K) = EXTRA
  EXTRA = ADEN(I, K)
  ADEN(I, K) = ADEN(J, K)
  ADEN(J, K) = EXTRA
44 CONTINUE

C Carry out elimination
CALL RECIP(A(I, I), ADEN(I, I), A(I, I), ADEN(I, I))
DO 55 K = 1, N
  IF (K .NE. I .AND. A(I, K) .NE. 0) THEN
    CALL MULTN(A(I, K), ADEN(I, K), A(I, I),
    1 ADEN(I, I), A(I, k), ADEN(I, K))
  ENDIF
55 CONTINUE

C Replace a(j,k) by a(j,k) - a(j,i)/a(i,i) * a(i,k)
DO 77 J = 1, N
  IF (J .NE. I .AND. A(J, I) .NE. 0) THEN
    DO 66 K = 1, N
      IF (K .NE. I .AND. A(I, K) .NE. 0) THEN
        CALL MULTN(A(J, I), ADEN(J, I),
        1 A(I, K), ADEN(I, K), TEMP1, TEMP2)
        CALL SUBTN(A(J, K), ADEN(J, K),
        1 TEMP1, TEMP2, A(J, K), ADEN(J, K))
      ENDIF
    66 CONTINUE
  ENDIF
77 CONTINUE

ENDIF
CONTINUE
DO 88 J = 1, N
   IF (J .NE. I .AND. A (J, I) .NE. 0) THEN
      TEMP = - A (I, I)
      CALL MULTN (A (J, I), ADEN (J, I), TEMP,
      1   ADEN (I, I), A (J, I), ADEN (J, I))
   ENDIF
88 CONTINUE
99 CONTINUE
IND = 0
IF (L .EQ. 0) RETURN
C Inter change the columns corresponding to the rows
C already interchanged, but in the reverse order
DO 122 I = 1, L
   LMI = L + 1 - I
   K1 = INT1 (LMI)
   K2 = INT2 (LMI)
   DO 111 J = 1, N
      EXTRA = A (J, K1)
      A (J, K1) = A (J, K2)
      A (J, K2) = EXTRA
      EXTRA = ADEN (J, K1)
      ADEN (J, K1) = ADEN (J, K2)
      ADEN (J, K2) = EXTRA
111 CONTINUE
122 CONTINUE
RETURN
END

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   1
TYPE THE INTEGER EIGENVALUES IN FORMAT (10I6)
   5000
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
   1
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
   1
THE REQUIRED MATRIX IS
   5000
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   3
TYPE THE INTEGER EIGENVALUES IN FORMAT (10I6)
   10 4 1
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
   1 -1 2
   -3 4 3
   1 -1 3
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
15 1 -11
12 1 -9
-1 0 1
THE REQUIRED MATRIX IS
100 6 -72
-261 -14 189
99 6 -71
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
15
ORDER OF THE DESIRED MATRIX TOO LARGE, TYPE A SMALLER NUMBER
AS THE ORDER OF THE MATRIX
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
5
TYPE THE INTEGER EIGENVALUES IN FORMAT (10I6)
-8 7 -5 4 2
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
-2 1 -4 2 -4
1 -1 2 -2 2
0 -1 1 -2 0
1 4 -1 9 2
1 2 3 -1 3
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
-27 -41 -30 -10 -2
-5 -8 -6 -2 0
-1 -2 1 0 0
2 3 3 1 0
14 22 14 5 1
THE REQUIRED MATRIX IS
-583 -904 -590 -206 -40
301 468 304 106 20
24 42 13 6 0
199 290 241 80 20
237 366 213 78 22
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
0

5.20 Program for generation of integer matrices starting from block-diagonal matrices

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), INVM (ND, ND), NMAT (ND, ND),
IBLOM (ND, ND), MKEI (ND, ND), P (ND)
10 WRITE (6, '('*'TYPE THE ORDER OF THE MATRIX IN'',''
1,1X,''FORMAT (I2)'')'))
READ (4, '(I2)') N
WRITE (6, '(1X, I2)') N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
    WRITE (6, '(/" ORDER OF THE MATRIX CANNOT BE NEGATIVE,")')
    GO TO 10
ELSEIF (N .GT. ND) THEN
    WRITE (6, '(/" ORDER OF THE DESIRED MATRIX TOO LARGE, TYPE")
    GO TO 10
ENDIF
WRITE (6, '(/" TYPE A MATRIX WITH UNIT DETERMINANT IN")
1,1X,' FORMAT (1016)')
DO 22 I = 1, N
    READ (4, '(1016)') (MAT (I, J), J = 1, N)
22 CONTINUE
DO 33 I = 1, N
    WRITE (6, '(1X, 1016)') (MAT (I, J), J = 1, N)
33 CONTINUE
CALL MATINV (MAT, INVM, N, IND)
C IND = 1 if the matrix is singular
IF (IND .EQ. 1) GO TO 10
WRITE (6, '(/" INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS")
1,1X,' FORMAT (1016)')
DO 44 I = 1, N
    WRITE (6, '(1X, 1016)') (INVM (I, J), J = 1, N)
44 CONTINUE
WRITE (6, '(/" TYPE A BLOCK DIAGONAL MATRIX IN")
1,1X,' FORMAT (1016)')
DO 55 I = 1, N
    READ (4, '(1016)') (BLOM (I, J), J = 1, N)
55 CONTINUE
DO 66 I = 1, N
    WRITE (6, '(1X, 1016)') (BLOM (I, J), J = 1, N)
66 CONTINUE
CALL MATMUL (MAT, BLOM, NMAT, N)
CALL MATMUL (NMAT, INVM, MKEI, N)
WRITE (6, '(/" THE REQUIRED MATRIX IS")
1,1X,' FORMAT (1017)')
DO 77 I = 1,N
    WRITE (6, '(1X, 1017)') (MKEI (I, J), J = 1, N)
77 CONTINUE
CALL CHPOLY (MKEI, N, P)
WRITE (6, '(/" THE COEFFICIENTS OF CHARACTERISTIC POLYNOMIAL/
1IX, OF THE MATRIX, STARTING FROM THE 2ND ARE GIVEN BELOW")
1IX, 1018)') (P (I), I = 1, N)
GO TO 10
END
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   1
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
   1
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
   1
TYPE A BLOCK DIAGONAL MATRIX IN FORMAT (10I6)
   50
THE REQUIRED MATRIX IS
   50
THE COEFFICIENTS OF CHARACTERISTIC POLYNOMIAL
OF THE MATRIX, STARTING FROM THE 2ND ARE GIVEN BELOW
   -50
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   2
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
   3 -1
   -2 1
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
   1 1
   2 3
TYPE A BLOCK DIAGONAL MATRIX IN FORMAT (10I6)
   2 1
   0 2
THE REQUIRED MATRIX IS
   8 9
   -4 -4
THE COEFFICIENTS OF CHARACTERISTIC POLYNOMIAL
OF THE MATRIX, STARTING FROM THE 2ND ARE GIVEN BELOW
   -4 4
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   20
ORDER OF THE DESIRED MATRIX TOO LARGE, TYPE A SMALLER NUMBER
AS THE ORDER OF THE MATRIX
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
   3
TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
   3 -1 4
   -2 1 -1
   1 -1 -1
INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
   -2 -5 -3
   -3 -7 -5
   1 2 1
TYPE A BLOCK DIAGONAL MATRIX IN FORMAT (10I6)
   2 1 0
   0 2 1
   0 0 2
THE REQUIRED MATRIX IS
THE COEFFICIENTS OF CHARACTERISTIC POLYNOMIAL
OF THE MATRIX, STARTING FROM THE 2ND ARE GIVEN BELOW
-6 12 -8

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
4

TYPE A MATRIX WITH UNIT DETERMINANT IN FORMAT (10I6)
1 0 1 1
-1 1 1 1
3 -1 2 2
-1 0 -1 0

INVERSE OF THE MATRIX WITH UNIT DETERMINANT IS
3 -1 -1 0
5 -1 -2 0
-3 1 1 -1
1 0 0 1

TYPE A BLOCK DIAGONAL MATRIX IN FORMAT (10I6)
3 1 0 0
0 3 0 0
0 0 -4 1
0 0 0 -4

THE REQUIRED MATRIX IS
23 -8 -9 1
10 -3 -5 1
45 -17 -17 2
-27 8 9 -5

THE COEFFICIENTS OF CHARACTERISTIC POLYNOMIAL
OF THE MATRIX, STARTING FROM THE 2ND ARE GIVEN BELOW
2 -23 -24 144

5.21 Program to generate a matrix with addition and subtraction operations only

IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), ITRIM (ND, ND)

10 WRITE (6, '('' TYPE THE ORDER OF THE MATRIX IN''
1,IX,''FORMAT (I2)'''))
READ (4, ''(I2)'') N
WRITE (6, ''(IX, I2)'' ) N
IF (N .EQ. 0) STOP
IF (N .LT. 0) THEN
    WRITE (6, ''('' ORDER OF THE MATRIX CANNOT BE NEGATIVE''

1 /"' KINDLY TYPE AGAIN''')GO TO 10
ELSEIF (N .GT. ND) THEN
WRITE (6, (''' ORDER OF DESIRED MATRIX IS TOO LARGE, TYPE''
1 ,'' A SMALLER NUMBER''/'' AS THE ORDER OF THE MATRIX''')')
GO TO 10
ENDIF
WRITE (6, (''' TYPE THE ENTIERES OF THE UPPER TRIANGULAR''
1,'''MATRIX IN FORMAT (10I6)'')')
DO 22 I = 1, N
READ (4, '(10I6)') (ITRIM(I, J), J = 1, N)
22 CONTINUE
DO 33 I = 1, N
WRITE (6, '(1X, 10I6)') (ITRIM (I, J), J = 1, N)
33 CONTINUE
CALL GEMAS (ITRIM, MAT, N)
WRITE (6, (''' THE REQUIRED MATRIX IS''')')
DO 44 I = 1, N
WRITE (6, '(1X, 10I7)') (MAT (I, J), J = 1, N)
44 CONTINUE
CALL INTERC (MAT, N)
GO TO 10
END
C PURPOSE - TO GENERATE A MATRIX WITH ADDITION AND SUBTRACTION
C OPERATIONS ONLY
C INPUT PARAMETERS
C F : TWO DIMENSIONAL ARRAY, CONTAINS THE FIRST MATRIX
C G : TWO DIMENSIONAL ARRAY, CONTAINS THE SECOND MATRIX
C K : ORDER OF THE MATRIX
C OUTPUT PARAMETERS
C H : TWO DIMENSIONAL ARRAY, CONTAINS THE GENERATED MATRIX
SUBROUTINE GEMAS (F, H, K)
PARAMETER (ND = 10)
INTEGER F (ND, ND), H (ND, ND)
IF (K .GT. 1) THEN
   DO 11 J = 1, K - 1
      H (1, J) = F (1, J) - F (1, J+1)
   11 CONTINUE
   DO 44 I = 2, K
      H (I, I-1) = H (I-1, I-1) - F (I, I)
      ITEMP = H (I, I-1)
      IF (I .LT. K) THEN
         DO 22 M = I+1, K
            H (M, I-1) = ITEMP
         22 CONTINUE
      ENDIF
   44 CONTINUE
   CALL GEMAS (H, H, K)
   CALL INTERC (H, H, K)
END
H(I, J) = H(I-1, J) + F(I, J) - F(I, J+1)

CONTINUE
ENDIF
44 CONTINUE
H(1, K) = F(1, K)
DO 55 I = 2, K
   H(I, K) = H(I-1, K) + F(I, K)
55 CONTINUE
ELSE
   H(1, 1) = F(1, 1)
ENDIF
RETURN
END

C PURPOSE - TO APPLY SIMILARITY TRANSFORMATION OF
C INTERCHANGING ROW AND COLUMN
C INPUT PARAMETERS
C COPYA: TWO-DIMENSIONAL ARRAY, THE MATRIX
C N: ORDER OF THE MATRIX
C OUTPUT PARAMETER
C COPYA: TWO-DIMENSIONAL ARRAY, THE MATRIX OBTAINED
C BY APPLYING SIMILARITY TRANSFORMATION

SUBROUTINE INTERC(COPYA, N)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION COPYA(ND, ND)
WRITE (6, ('" TYPE THE NUMBER OF SIMILARITY TRANSFORMATION"
1,1X,'"IN FORMAT (12)"
'))
READ (4, '(12)') NS
WRITE (6, ('" TYPE THE ROW AND COLUMN NUMBER"
1,1X,'"IN FORMAT (212)"
'))
READ (4, '(212)') I, J
WRITE (6, ('" DATA ILLEGAL, KINDLY TYPE AGAIN"
'))
GO TO 10
10 WRITE (6, ('" TYPE THE ROW AND COLUMN NUMBER"
1,1X,'"IN FORMAT (212)"
'))
READ (4, '(212)') I, J
WRITE (6, ('" DATA ILLEGAL, KINDLY TYPE AGAIN"
'))
GO TO 10
ENDIF
DO 22 K = 1, N
   EXTRA = COPYA(I, K)
   COPYA(I, K) = COPYA(J, K)
   COPYA(J, K) = EXTRA
22 CONTINUE
DO 33 K = 1, N
   EXTRA = COPYA(K, I)
33 CONTINUE
COPYA (K, I) = COPYA (K, J)
COPYA (K, J) = EXTRA
CONTINUE
WRITE (6, '('' THE CHANGED MATRIX IS'')')
DO 44 II = 1, N
   WRITE (6, '(IX, 1017)') (COPYA (II, JJ), JJ = 1, N)
44 CONTINUE
D = D + 1
IF (D .EQ. NS) RETURN
GO TO 10
END

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
1
TYPE THE ENTIREs OF THE UPPER TRIANGULAR MATRIX IN FORMAT (10I6)
100
THE REQUIRED MATRIX IS
100
TYPE THE NUMBER OF SIMILARITY TRANSFORMATION IN FORMAT (I2)
1
TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)
1 1
THE CHANGED MATRIX IS
100
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
2
TYPE THE ENTIREs OF THE UPPER TRIANGULAR MATRIX IN FORMAT (10I6)
25 8
0 15
THE REQUIRED MATRIX IS
17 8
2 23
TYPE THE NUMBER OF SIMILARITY TRANSFORMATION IN FORMAT (I2)
1
TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)
1 2
THE CHANGED MATRIX IS
23 2
8 17
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
3
TYPE THE ENTIREs OF THE UPPER TRIANGULAR MATRIX IN FORMAT (10I6)
25 28 -42
0 4 3
0 0 60
THE REQUIRED MATRIX IS
-3 70 -42
-7 71 -39
\[-7 \quad 11 \quad 21\]

TYPE THE NUMBER OF SIMILARITY TRANSFORMATION IN FORMAT (I2)

1

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

1 3

THE CHANGED MATRIX IS

\[
\begin{pmatrix}
21 & 11 & -7 \\
-39 & 71 & -7 \\
-42 & 70 & -3
\end{pmatrix}
\]

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)

4

TYPE THE ENTIRE OF THE UPPER TRIANGULAR MATRIX IN FORMAT (10I6)

\[
\begin{array}{cccc}
16 & 6 & 4 & 2 \\
0 & 8 & 6 & 3 \\
0 & 0 & -1 & 1 \\
0 & 0 & 0 & -3
\end{array}
\]

THE REQUIRED MATRIX IS

\[
\begin{pmatrix}
10 & 2 & 2 & 2 \\
2 & 4 & 5 & 5 \\
2 & 5 & 3 & 6 \\
2 & 5 & 6 & 3
\end{pmatrix}
\]

TYPE THE NUMBER OF SIMILARITY TRANSFORMATION IN FORMAT (I2)

2

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

2 4

THE CHANGED MATRIX IS

\[
\begin{pmatrix}
10 & 2 & 2 & 2 \\
2 & 3 & 6 & 5 \\
2 & 6 & 3 & 5 \\
2 & 5 & 5 & 4
\end{pmatrix}
\]

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

0 1

DATA ILLEGAL, KINDLY TYPE AGAIN

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

7 8

DATA ILLEGAL, KINDLY TYPE AGAIN

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

2-1

DATA ILLEGAL, KINDLY TYPE AGAIN

TYPE THE ROW AND COLUMN NUMBER IN FORMAT (2I2)

1 3

THE CHANGED MATRIX IS

\[
\begin{pmatrix}
3 & 6 & 2 & 5 \\
6 & 3 & 2 & 5 \\
2 & 2 & 10 & 2 \\
5 & 5 & 2 & 4
\end{pmatrix}
\]

TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)

0
C PURPOSE - TO CHANGE THE ELEMENTS OF THE MATRIX BY APPLYING
C A SIMILARITY TRANSFORMATION OF PARTICULAR TYPE
C INPUT PARAMETERS
C MAT: TWO-DIMENSIONAL ARRAY, CONTAINS THE MATRIX
C N : THE ORDER OF THE MATRIX
C OUTPUT PARAMETER
C MAT: TWO-DIMENSIONAL ARRAY, CONTAINS THE CHANGED MATRIX

SUBROUTINE CEMAT (MAT, N)
IMPLICIT INTEGER*4 (A - Z)
PARAMETER (ND = 10)
DIMENSION MAT (ND, ND), MUP (ND), IHCF (ND)
IF (N .EQ. 1) THEN
   IHCF (1) = MAT (1, 1)
ELSE
   DO 22 I = 1, N
      IF (I .EQ. 1) THEN
         IHCF (I) = MAT (2, I)
      ELSE
         IHCF (I) = MAT (1, I)
      ENDIF
      DO 11 J = 2, N
         IF (I .NE. J) THEN
            IHCF (I) = HCF (MAT (J, I), IHCF (I))
         ENDIF
      11 CONTINUE
   22 CONTINUE
ENDIF
WRITE (6,'("MULTIPLIERS MAY BE THE FACTORS OF")
1/10(1X,I4))'(IHCF (I), I = 1, N)
WRITE (6, '("TYPE THE MULTIPLIERS IN FORMAT (10I6)")')
READ (4, '(10I6)') (MUP (I), I = 1, N)
WRITE (6, '(1X, 10I6)') (MUP (I), I = 1, N)
DO 44 J = 1, N
   DO 33 I = 1, N
      IF (I .NE. J) THEN
         ITEMP = MAT (I, J) * MUP (I) / MUP (J)
         IF (ITEMP * MUP(J) .NE. MAT(I, J) * MUP(I)) THEN
            WRITE (6,'(1X,"CANNOT BE A MULTIPLIER OF")
1,1X,"COLUMN",I2)') MUP(J), J
            RETURN
         ENDIF
      ENDIF
   33 CONTINUE
   44 CONTINUE
DO 66 J = 1, N
   DO 55 I = 1, N
MAT (I, J) = MAT (I, J) * MUP (I) / MUP (J)
55 CONTINUE
66 CONTINUE
WRITE (6, '(" THE MATRIX WITH CHANGED ELEMENTS IS")')
DO 77 I = 1, N
   WRITE (6, '(1X, 10I6)') (MAT (I, J), J = 1, N)
77 CONTINUE
RETURN
END

The following output is generated using the subroutine GEMAS; and then we have applied another simple similarity transformation using the routine CEMAT, instead of applying the similarity transformation of interchanging rows and columns.

**TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)**
2

**TYPE AN UPPER TRIANGULAR MATRIX IN FORMAT (10I6)**
16  11
  0  -4

**THE POSITIVE MATRIX IS**
  5  11
   9  7

**MULTIPLIERS MAY BE THE FACTORS OF**
  9  11

**TYPE THE MULTIPLIERS IN FORMAT (10I6)**
    1  1

**THE MATRIX WITH CHANGED ELEMENTS IS**
  5  11
   9  7

**TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)**
3

**TYPE AN UPPER TRIANGULAR MATRIX IN FORMAT (10I6)**
15  10  4
  0  -2  2
  0  2  -1

**THE ABOVE MATRIX IS NOT AN UPPER TRIANGULAR MATRIX**

**TYPE AN UPPER TRIANGULAR MATRIX IN FORMAT (10I6)**
15  10  4
  0  -2  2
  0  0  -1

**THE POSITIVE MATRIX IS**
  5  6  4
  7  2  6
   7  3  5

**MULTIPLIERS MAY BE THE FACTORS OF**
  7  3  2

**TYPE THE MULTIPLIERS IN FORMAT (10I6)**
  7  1  2
THE MATRIX WITH CHANGED ELEMENTS IS
\[
\begin{bmatrix}
5 & 42 & 14 \\
1 & 2 & 3 \\
2 & 6 & 5 \\
\end{bmatrix}
\]
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
4
TYPE AN UPPER TRIANGULAR MATRIX IN IN FORMAT (10I6)
\[
\begin{bmatrix}
20 & 10 & 7 & 2 \\
0 & 2 & 2 & 3 \\
0 & 0 & 1 & 4 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
THE POSITIVE MATRIX IS
\[
\begin{bmatrix}
10 & 3 & 5 & 2 \\
8 & 3 & 4 & 5 \\
8 & 2 & 1 & 9 \\
8 & 2 & 1 & 9 \\
\end{bmatrix}
\]
MULTIPLIERS MAY BE THE FACTORS OF
\[
\begin{bmatrix}
8 & 1 & 1 & 1 \\
\end{bmatrix}
\]
TYPE THE MULTIPLIERS IN FORMAT (10I6)
\[
\begin{bmatrix}
1 & 2 & 2 & 2 \\
\end{bmatrix}
\]
2 CANNOT BE A MULTIPLIER OF COLUMN 2
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
5
TYPE AN UPPER TRIANGULAR MATRIX IN IN FORMAT (10I6)
\[
\begin{bmatrix}
20 & 15 & 7 & 4 & 2 \\
0 & 3 & 4 & 1 & 2 \\
0 & 0 & 3 & 3 & 2 \\
0 & 0 & 0 & 3 & 2 \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
THE POSITIVE MATRIX IS
\[
\begin{bmatrix}
5 & 8 & 3 & 2 & 2 \\
2 & 7 & 6 & 1 & 4 \\
2 & 4 & 6 & 2 & 6 \\
2 & 4 & 3 & 3 & 8 \\
2 & 4 & 3 & 2 & 9 \\
\end{bmatrix}
\]
MULTIPLIERS MAY BE THE FACTORS OF
\[
\begin{bmatrix}
2 & 4 & 3 & 1 & 2 \\
\end{bmatrix}
\]
TYPE THE MULTIPLIERS IN FORMAT (10I6)
\[
\begin{bmatrix}
2 & 2 & 3 & 1 & 2 \\
\end{bmatrix}
\]
THE MATRIX WITH CHANGED ELEMENTS IS
\[
\begin{bmatrix}
5 & 8 & 2 & 4 & 2 \\
2 & 7 & 4 & 2 & 4 \\
3 & 6 & 6 & 6 & 9 \\
1 & 2 & 1 & 3 & 4 \\
2 & 4 & 2 & 4 & 9 \\
\end{bmatrix}
\]
TYPE THE ORDER OF THE MATRIX IN FORMAT (I2)
0

C PURPOSE - TO SUBTRACT ONE VULGAR FRACTION FROM ANOTHER
C INPUT PARAMETERS
C I : THE NUMERATOR OF THE MINUEND
C J : THE DENOMINATOR OF THE MINUEND
C K : THE NUMERATOR OF THE SUBTRAHEND
C L : THE DENOMINATOR OF THE SUBTRAHEND
C OUTPUT PARAMETERS
C M : THE NUMERATOR OF THE DIFFERENCE
C N : THE DENOMINATOR OF THE DIFFERENCE
SUBROUTINE SUBTN (I, J, K, L, M, N)
IMPLICIT INTEGER*4 (A - Z)
DEN = J / HCF (J, L) * L
NUM = DEN / J * I - DEN / L * K
FAC = HCF (NUM, DEN)
M = NUM / FAC
N = DEN / FAC
RETURN
END

C PURPOSE - TO MULTIPLY TWO VULGAR FRACTION
C INPUT PARAMETERS
C I : THE NUMERATOR OF THE MULTIPLICAND
C J : THE DENOMINATOR OF THE MULTIPLICAND
C K : THE NUMERATOR OF THE MULTIPLIER
C L : THE DENOMINATOR OF THE MULTIPLIER
C OUTPUT PARAMETERS
C M : THE NUMERATOR OF THE PRODUCT
C N : THE DENOMINATOR OF THE PRODUCT
SUBROUTINE MULTN (I, J, K, L, M, N)
IMPLICIT INTEGER*4 (A - Z)
HCFl = HCF (I, L)
HCF2 = HCF (J, K)
M = I / HCFl * (K / HCF2)
N = J / HCF2 * (L / HCFl)
RETURN
END

C PURPOSE - TO FIND THE RECIPROCAL OF A VULGAR FRACTION
C INPUT PARAMETERS
C I : THE NUMERATOR OF THE VULGAR FRACTION
C J : THE DENOMINATOR OF THE VULGAR FRACTION
C OUTPUT PARAMETERS
C K : THE NUMERATOR OF THE RECIPROCAL FRACTION
C L : THE DENOMINATOR OF THE RECIPROCAL FRACTION
SUBROUTINE RECIP (I, J, K, L)
IMPLICIT INTEGER*4 (A - Z)
ITEMP=I
K = ISIGN (J, I)
L = IABS (ITEMP)
RETURN
C PURPOSE - TO REDUCE THE GIVEN POLYNOMIAL TO A PRIMITIVE POLYNOMIAL
C INPUT PARAMETERS
C U : LINEAR ARRAY, THE COEFFICIENTS OF THE GIVEN POLYNOMIAL
C NCF : THE NUMBER OF COEFFICIENTS OF THE POLYNOMIAL
C OUTPUT PARAMETERS
C U : LINEAR ARRAY, COEFFICIENTS OF THE PRIMITIVE POLYNOMIAL

SUBROUTINE REMCOM (U, NCF)
IMPLICIT INTEGER*4 (A - Z)
DIMENSION U (NCF)
ICOHCF = IABS (U (1))
DO 22 I = 2, NCF
   ICOHCF = HCF (ICOHCF, U (I))
22 CONTINUE
DO 33 I = 1, NCF
   U (I) = U (I) / ICOHCF
33 CONTINUE
RETURN
END

C PURPOSE - TO FIND THE HCF OF TWO INTEGERS
C INPUT PARAMETERS
C I : THE FIRST NUMBER
C J : THE SECOND NUMBER

INTEGER FUNCTION HCF (I, J)
IMPLICIT INTEGER*4 (A - Z)
IF (I .EQ. 0) THEN
   HCF = IABS (J)
ELSEIF (J .EQ. 0) THEN
   HCF = IABS (I)
ELSE
   KOPY1 = IABS (I)
   KOPY2 = IABS (J)
10 REM = MOD (KOPY1, KOPY2)
   IF (REM .NE. 0) THEN
      KOPY1 = KOPY2
      KOPY2 = REM
      GO TO 10
   ELSE
      HCF = KOPY2
   ENDIF
ENDIF
RETURN
END
CONTENTS OF THE APPENDIX

5 1 The program for addition of two non-negative m, n-piece integers 117
5 2 The program for subtracting a non-negative m-piece integer from a non-negative n-piece integer 122
5 3 The program for multiplication of two non-negative m, n-piece integers 126
5 4 Program for division of an m-piece non-negative integer by an n-piece positive integer 132
5 5 Given a positive integer this program computes the exact integer part of its square root 140
5 6 The program to generate all the prime numbers 143
5 7 This program creates a nest of DO loops and finds the distinct indices of the loops and their sum 147
5 8 Program to reconstruct a polynomial from its integer zeros 154
5 9 Program for the reconstruction of a polynomial over $\mathbb{Z}$ from its rational zeros 157
5 10 Program for the reconstruction of polynomial over $\mathbb{Z}$ from its complex and surd zeros 160
5 11 Given a monic polynomial over $\mathbb{Z}$, this program computes its integer zeros 162
5 12 The program to calculate the H.C.F. of two polynomials over the unique factorization domain $\mathbb{Z}$ 169
5 13 Program to find the quadratic factors of a polynomial over $\mathbb{Z}$ 174
5 14 This program computes the rational zeros of a polynomial over $\mathbb{Z}$ 179
5 15 Given a square matrix, this program computes the integer eigenvalues of the matrix 183
5 16 This program computes the eigenvectors of a given matrix 187
5 17 The program computes the Jordan canonical form of a given matrix 195
5 18 Program to find the entries of a matrix with given determinant 199
5 19 Program for generation of integer matrices (diagonalizable) with pre-assigned integer spectra 204
5 20 Program for generation of integer matrices starting from block-diagonal matrices 209
5 21 Program to generate a matrix with addition and subtraction operations only 212
CONTENTS OF THE APPENDIX

1 SUBROUTINE BIGADD 118
2 SUBROUTINE PRINT 119
3 SUBROUTINE BIGSUB 123
4 SUBROUTINE BIGMUL 127
5 SUBROUTINE MULT 128
6 SUBROUTINE SHIFT 129
7 SUBROUTINE INTEG 130
8 SUBROUTINE BIGDIV 133
9 SUBROUTINE PARDEE 136
10 SUBROUTINE APPEND 137
11 SUBROUTINE EXACDQ 137
12 SUBROUTINE SQRUT 140
13 SUBROUTINE SIMPGV 141
14 SUBROUTINE TONRAP 142
15 SUBROUTINE PRIME 143
16 SUBROUTINE UPTHIR 145
17 SUBROUTINE START 148
18 SUBROUTINE PRE 149
19 SUBROUTINE NEST 150
20 SUBROUTINE CORE 151
21 SUBROUTINE POST 151
22 SUBROUTINE RECONZ 155
23 SUBROUTINE LINZ 155
24 SUBROUTINE RECONQ 158
<table>
<thead>
<tr>
<th>Number</th>
<th>Subroutine</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>SUBROUTINE LINQ</td>
<td>159</td>
</tr>
<tr>
<td>26</td>
<td>SUBROUTINE RCONSC</td>
<td>161</td>
</tr>
<tr>
<td>27</td>
<td>SUBROUTINE QMUL</td>
<td>161</td>
</tr>
<tr>
<td>28</td>
<td>SUBROUTINE ZEROS</td>
<td>163</td>
</tr>
<tr>
<td>29</td>
<td>SUBROUTINE TELIN</td>
<td>164</td>
</tr>
<tr>
<td>30</td>
<td>SUBROUTINE POLDIV</td>
<td>165</td>
</tr>
<tr>
<td>31</td>
<td>SUBROUTINE FAC</td>
<td>166</td>
</tr>
<tr>
<td>32</td>
<td>SUBROUTINE SORT</td>
<td>167</td>
</tr>
<tr>
<td>33</td>
<td>SUBROUTINE POLHCF</td>
<td>171</td>
</tr>
<tr>
<td>34</td>
<td>SUBROUTINE POLDI</td>
<td>172</td>
</tr>
<tr>
<td>35</td>
<td>SUBROUTINE REIMA</td>
<td>175</td>
</tr>
<tr>
<td>36</td>
<td>SUBROUTINE DIVPMQ</td>
<td>177</td>
</tr>
<tr>
<td>37</td>
<td>SUBROUTINE CPDP</td>
<td>177</td>
</tr>
<tr>
<td>38</td>
<td>SUBROUTINE ZEROSQ</td>
<td>180</td>
</tr>
<tr>
<td>39</td>
<td>SUBROUTINE LEASTK</td>
<td>181</td>
</tr>
<tr>
<td>40</td>
<td>SUBROUTINE CHPOLY</td>
<td>185</td>
</tr>
<tr>
<td>41</td>
<td>SUBROUTINE MULT</td>
<td>185</td>
</tr>
<tr>
<td>42</td>
<td>SUBROUTINE FECEIV</td>
<td>189</td>
</tr>
<tr>
<td>43</td>
<td>SUBROUTINE SOLVE</td>
<td>190</td>
</tr>
<tr>
<td>44</td>
<td>SUBROUTINE MULLCM</td>
<td>193</td>
</tr>
<tr>
<td>45</td>
<td>SUBROUTINE FEJORD</td>
<td>196</td>
</tr>
<tr>
<td>46</td>
<td>SUBROUTINE MUD</td>
<td>201</td>
</tr>
<tr>
<td>47</td>
<td>FUNCTION LDET</td>
<td>202</td>
</tr>
<tr>
<td>48</td>
<td>SUBROUTINE DMM</td>
<td>205</td>
</tr>
<tr>
<td>SUBROUTINE NAME</td>
<td>PAGE</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>49 SUBROUTINE MATMUL</td>
<td>206</td>
<td></td>
</tr>
<tr>
<td>50 SUBROUTINE MATINV</td>
<td>206</td>
<td></td>
</tr>
<tr>
<td>51 SUBROUTINE GEMAS</td>
<td>213</td>
<td></td>
</tr>
<tr>
<td>52 SUBROUTINE INTERC</td>
<td>214</td>
<td></td>
</tr>
<tr>
<td>53 SUBROUTINE CEMAT</td>
<td>217</td>
<td></td>
</tr>
<tr>
<td>54 SUBROUTINE SUBTN</td>
<td>220</td>
<td></td>
</tr>
<tr>
<td>55 SUBROUTINE MULTN</td>
<td>220</td>
<td></td>
</tr>
<tr>
<td>56 SUBROUTINE RECIP</td>
<td>220</td>
<td></td>
</tr>
<tr>
<td>57 SUBROUTINE REMCOM</td>
<td>221</td>
<td></td>
</tr>
<tr>
<td>58 INTEGER FUNCTION HCF</td>
<td>221</td>
<td></td>
</tr>
</tbody>
</table>