```
REM ---------------------------------------------------------------
CALL AutoSpostamenti
REM ---------------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER LA STAMPA DEI RISULTATI
REM ---------------------------------------------------------------
TIME2 = TIME2 + TIMER - TIME1
CALL UscitaRisultati
REM ---------------------------------------------------------------
REM CHIUDE TUTTI I BUFFER TEMPORANEI
REM ---------------------------------------------------------------
CLOSE
END

SUB AutoSpostamenti
REM
REM  ************************************************************
REM *                                                            *
REM *                    A u t o S p o s t a m e n t i           *
REM *                                                            *
REM  ************************************************************
REM
REM ---------------------------------------------------------------
REM Calcolo degli spostamenti verticali connessi agli autovettori
REM ---------------------------------------------------------------
REM
FOR K = 1 TO NEIG
  SPOS(1, K) = 0
  FOR I = 2 TO T - 1
    SPOS(I, K) = SPOS(I - 1, K) - LTRAT * VECT(I - 1, K)
  NEXT I
  F1 = 0
  F2 = 0
  FOR I = 1 TO T - 2
    F1 = F1 + S1(I) * VECT(I, K)
    F2 = F2 + S2(I) * VECT(I, K)
  NEXT I
  SPOS(T, K) = SPOS(T - 1, K) - F1 * LTRAT
  SPOS(T + 1, K) = SPOS(T, K) - F2 * LTRAT
NEXT K
REM ---------------------------------------------------------------
REM Calcolo degli spostamenti orizzontali connessi agli autovettori
REM ---------------------------------------------------------------
```

```
FOR K = 1 TO NEIG
  SBAN(1, K) = 0
  FOR I = 2 TO T - 1
    SBAN(I, K) = SBAN(I - 1, K) - (Q(I) - Q(I - 1)) * VECT(I - 1, K)
  NEXT I
  F1 = 0
  F2 = 0
  FOR I = 1 TO T - 2
    F1 = F1 + S1(I) * VECT(I, K)
    F2 = F2 + S2(I) * VECT(I, K)
  NEXT I
  SBAN(T, K) = SBAN(T - 1, K) - F1 * (Q(T) - Q(T - 1))
  SBAN(T + 1, K) = SBAN(T, K) - F2 * (Q(T + 1) - Q(T))
NEXT K
END SUB

SUB backsub (U(), V(), L(), LD(), NK, N, M, LCK)
REM
REM  ***************************************************************
REM *                                                               *
REM *                          b a c k s u b                        *
REM *                                                               *
REM  ***************************************************************
REM
REM  ***************************************************************
REM *                                                               *
REM * RISOLVE PER SOSTITUZIONE ALL'INDIETRO IL SISTEMA L(t)V=U      *
REM * PER LE COLONNE DA LOCK AD M DI V(N,M). L E' TRIANGOLARE       *
REM * BASSA IN FORMA A BANDA MOBILE, CON INDIRIZZI IN LD(N)         *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION              *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                  *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663,(1976)                   *
REM *                                                               *
REM  ***************************************************************
REM
FOR K = LCK TO M
  FOR I = 1 TO N
    V(I, K) = U(I, K) / L(LD(I))
  NEXT I
NEXT K
FOR IT = 2 TO N
```

```
  I = N + 2 - IT
  Q = LD(I) - I
  M1 = 1 + LD(I - 1) - Q
  FOR J = M1 TO I - 1
    FOR K = LCK TO M
      V(J, K) = V(J, K) - L(J + Q) * V(I, K) / L(LD(J))
    NEXT K
  NEXT J
NEXT IT
END SUB

SUB decouple (U(), V(), W(), BD(), N, M, LCK, TOLVEC)
REM
REM  ***********************************************************************
REM *                                                                       *
REM *                              d e c o u p l e                          *
REM *                                                                       *
REM  ***********************************************************************
REM
REM  ***********************************************************************
REM *                                                                       *
REM * ORDINA I VETTORI U(N,M) E V(N,M) IN MODO CHE LE COLONNE               *
REM * DA LCK AD M SIANO IN ORDINE DECRESCENTE                               *
REM * SECONDO IL MODULO DEGLI AUTOVALORI.                                   *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                         *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION                       *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                           *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663,(1976)                            *
REM *                                                                       *
REM  ***********************************************************************
REM
REM ---------------------------------------------------------------------
REM CALCOLA GLI AUTOVALORI
REM ---------------------------------------------------------------------
FOR J = LCK TO M
  EL = 0
  FOR I = 1 TO N
    EL = EL + U(I, J) * V(I, J)
  NEXT I
  BD(J) = EL
NEXT J
REM ---------------------------------------------------------------------
```

```
REM ORDINA GLI AUTOVALORI
REM ---------------------------------------------------------------
FOR J = LCK + 1 TO M
  I = J
  DO
    K = I - 1
    IF ABS(BD(K)) < ABS(BD(I)) THEN
      EL = BD(I)
      BD(I) = BD(K)
      BD(K) = EL
      FOR II = 1 TO N
        EL = U(II, I)
        EL1 = V(II, I)
        U(II, I) = U(II, K)
        U(II, K) = EL
        V(II, I) = V(II, K)
        V(II, K) = EL1
      NEXT II
    END IF
    I = I - 1
  LOOP WHILE I > LCK + 1
NEXT J
REM ---------------------------------------------------------------
REM DISACCOPPIA I VETTORI
REM ---------------------------------------------------------------
FOR I = 1 TO N
  FOR J = LCK TO M
    W(I, J) = V(I, J)
  NEXT J
NEXT I
FOR I = LCK TO M
  FOR J = LCK TO I - 1
    EL = 0
    FOR K = 1 TO N
      EL = EL + U(K, I) * V(K, J)
    NEXT K
    EL = -2 * EL
    Q = BD(I) - BD(J)
    IF ABS(Q / BD(J)) > TOLVEC OR ABS(EL / BD(J)) > TOLVEC THEN
      ELL = SQR(Q * Q + 4 * EL * EL)
      IF Q < 0 THEN ELL = -ELL
      EL = EL / (Q + ELL)
```

```
      FOR K = 1 TO N
        W(K, I) = W(K, I) - EL * V(K, J)
        W(K, J) = W(K, J) + EL * V(K, I)
      NEXT K
    ELSE
      EL = 0
    END IF
  NEXT J
NEXT I
END SUB

SUB errore (U(), W(), BD(), ERRO(), N, M, NRQD, LCK, TOLVEC, LT)
REM
REM  ************************************************************
REM *                                                            *
REM *                          e r r o r e                       *
REM *                                                            *
REM  ************************************************************
REM
REM  ************************************************************
REM *                                                            *
REM * STIMA GLI ERRORI ASSOCIATI ALLE PREVISIONI U(N,M) E W(N,M) *
REM * CON I CORRISPONDENTI AUTOVALORI IN BD(M).                  *
REM * SOLO GLI ERRORI DA LCK AD M VENGONO STIMATI.               *
REM * GLI ERRORI SONO ANCHE CONFRONTATI CON TOLVEC, E SE         *
REM * NRQD VETTORI SODDISFANO QUESTO CRITERIO, SI                *
REM * PONE LT = 1, INDICANDO CHE L'ITERAZIONE E' COMPLETA.       *
REM *                                                            *
REM  ************************************************************
REM
REM  ************************************************************
REM *                                                            *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN              *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION            *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, 1976)                 *
REM *                                                            *
REM  ************************************************************
REM
FOR J = LCK TO M
  ER = 0
  IF BD(J) > 0 THEN
```

```
    FOR I = 1 TO N
      EL = U(I, J) - W(I, J)
      ER = ER + EL * EL
    NEXT I
  ELSE
    FOR I = 1 TO N
      EL = U(I, J) + W(I, J)
      ER = ER + EL * EL
    NEXT I
  END IF
  ERRO(J) = SQR(ER / N)
NEXT J
12 IF TOLVEC < ERRO(LCK) THEN EXIT SUB
ERRO(LCK) = -ERRO(LCK)
LCK = LCK + 1
IF NRQD >= LCK THEN GOTO 12
LT = 1
END SUB

SUB forsub (U(), V(), L(), LD(), NK, N, M, LCK)
REM
REM  ************************************************************************
REM *                                                                        *
REM *                                  f o r s u b                           *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM * RISOLUZIONE PER SOSTITUZIONE IN AVANTI DEL SISTEMA LV=U                *
REM * PER LE COLONNE DA LOCK AD M DI V(N,M). L E' TRIANGOLARE                *
REM * BASSA IN FORMA A BANDA MOBILE, CON INDIRIZZI IN LD(N).                 *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                         *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION                       *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                           *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                           *
REM *                                                                        *
REM  ************************************************************************
REM
FOR K = LCK TO M
  V(1, K) = U(1, K) / L(1)
NEXT K
```

```
FOR I = 2 TO N
  Q = LD(I) - I
  M1 = 1 + LD(I - 1) - Q
  FOR K = LCK TO M
    EL = 0
    FOR J = M1 TO I - 1
      EL = EL + L(J + Q) * V(J, K)
    NEXT J
  V(I, K) = (U(I, K) - EL) / L(LD(I))
  NEXT K
NEXT I
END SUB

SUB Geometria
REM
REM  ************************************************************************
REM *                                                                        *
REM *                          G e o m e t r i a                             *
REM *                                                                        *
REM  ************************************************************************
REM ------------------------------------------------------------------------
REM Riempimento degli array delle inerzie
REM ------------------------------------------------------------------------
IF SEZIONE$ = "PARABOLA" OR SEZIONE$ = "COSENOW" THEN
  FOR I = 1 TO T + 1
    Z = LUCE / T * (I - 1)
    IF SEZIONE$ = "PARABOLA" THEN INERZ(I) = FNINERZ1(Z)
    IF SEZIONE$ = "COSENOW" THEN INERZ(I) = FNINERZ2(Z)
  NEXT I
END IF
REM ------------------------------------------------------------------------
REM Riempimento dell'array delle quote, se descritte da funzioni
REM ------------------------------------------------------------------------
IF ASSE$ = "FUNC" THEN
  FOR I = 1 TO T + 1
    Z = LUCE / T * (I - 1)
    Q(I) = FNQ(Z)
  NEXT I
END IF
REM ------------------------------------------------------------------------
REM Riempimento degli array dei carichi, se descritti da funzioni,
REM e relativo calcolo della spinta
```

```
REM ------------------------------------------------------------------------
IF CARICHI$ = "FUNC" THEN
  FOR I = 2 TO T
    FVERT(I) = QUNIF * LUCE / T
  NEXT I
  FVERT(1) = 0: FVERT(T + 1) = 0
  HSPINTA = QUNIF * LUCE ^ 2 / 8 / FREC
END IF
REM ------------------------------------------------------------------------
REM Riempimento degli array delle inclinazioni dei tratti
REM ------------------------------------------------------------------------
INCL(1) = ATN(Q(2) / LUCE * T)
FOR I = 2 TO T
  INCL(I) = ATN((Q(I + 1) - Q(I - 1)) / LUCE / 2 * T)
NEXT I
INCL(T + 1) = ATN((Q(T + 1) - Q(T)) / LUCE * T)
END SUB

SUB IngressoDati
REM
REM ************************************************************************
REM *                                                                      *
REM *                          I n g r e s s o D a t i                     *
REM *                                                                      *
REM ************************************************************************
REM
REM ************************************************************************
REM *                                                                      *
REM *          Questa subroutine si occupa dell'ingresso dei dati          *
REM *                    non organizzati in array                          *
REM *                                                                      *
REM ************************************************************************
REM ************************************************************************
REM *                                                                      *
REM *   LEGGE IL TITOLO DELLA STRUTTURA                                    *
REM *   IL FLAG DI STAMPA PER LE VARIABILI DI INGRESSO                     *
REM *   IL FLAG DI STAMPA PER LE VARIABILI DI USCITA                       *
REM *     - CARTA → I DATI VENGONO STAMPATI                                *
REM *     - VIDEO → I DATI VENGONO INVIATI A VIDEO                         *
REM *     - FILES → I DATI VENGONO MEMORIZZATI SU                          *
REM *         FILES DA SPECIFICARE                                         *
REM *     - VIDEOFILES → I DATI VENGONO INVIATI A                          *
```

```
REM *          VIDEO E MEMORIZZATI SU FILES                          *
REM *      - CARTAFILES → I DATI VENGONO STAMPATI                    *
REM *          CARTA E MEMORIZZATI SU FILES                          *
REM *      - TUTTO → I DATI VENGONO INVIATI A VIDEO,                 *
REM *          STAMPATI SU CARTA E MEMORIZZATI SU FILES              *
REM *                                                                *
REM ************************************************************************
REM
INPUT #7, TITLE$
INPUT #7, STAMPA$
INPUT #7, STAMPARIS$
IF STAMPA$ = "FILES" OR STAMPA$ = "VIDEOFILES" OR STAMPA$ = "CAR-
   TAFILES" OR STAMPA$ = "TUTTO" THEN
  INPUT #7, FILEINGR$
END IF
IF STAMPARIS$ = "FILES" OR STAMPARIS$ = "VIDEOFILES" OR STAM-
   PARIS$ = "CARTAFILES" OR STAMPARIS$ = "TUTTO" THEN
  INPUT #7, FILEUSC$
END IF
CALL OpenFiles
REM
REM ************************************************************************
REM *                                                                *
REM *   LEGGE:                                                       *
REM *     LUCE    Luce dell'arco                                     *
REM *     T        Numero di coordinate lagrangiane                  *
REM *     E        Modulo di Young                                   *
REM *     CS       Cedibilita' angolare a sinistra                   *
REM *     CD       Cedibilita' angolare a destra                     *
REM *     NEIG   Numero di autovalori da calcolare                   *
REM *     NVEC    Numero di vettori per condurre                     *
REM *              l'iterazione sul sottospazio (NVEC > NEIG         *
REM *     NOI      Numero massimo di iterazioni                      *
REM *     TOLVEC   Errore ammesso sugli autovettori                  *
REM *     ASSE$    Se l'asse dell'arco e' descritto da una           *
REM *              funzione, ASSE$ = FUNZ, se invece e' dato         *
REM *              per punti, allora ASSE$ = DATI                    *
REM *     VINCOLO$   Può assumere i seguenti valori:                 *
REM *              INCASTRO                                          *
REM *              1 CERNIERA                                        *
REM *              2 CERNIERE                                        *
REM *              3 CERNIERE                                        *
```

```
REM *      SEZIONE$   Se la variazione del momento di inerzia              *
REM *          della sezione retta e' data per punti,                      *
REM *          allora SEZIONE$ = DATI. Altrimenti sono                     *
REM *          disponibili le seguenti opzioni:                            *
REM *              = PARABOLA se le inerzie variano                        *
REM *              con legge parabolica                                    *
REM *              = COSENOW se le inerzie variano                         *
REM *              con la legge del coseno                                 *
REM *      CARICHI$   Se i carichi sull'arco sono descritti da             *
REM *              funzione, CARICHI$ = FUNZ, se invece                    *
REM *              sono dati per punti, allora CARICHI$ = DATI             *
REM *      Se l'asse e' definito da funzione:                              *
REM *      FREC   Freccia dell'arco                                        *
REM *      RIAL   Rialzo a destra                                          *
REM *      Se l'inerzia della sezione varia secondo PARABOLA:              *
REM *      INERZS   Momento di inerzia a sinistra                          *
REM *      INERZC   Momento di inerzia al centro                           *
REM *      INERZD   Momento di inerzia a destra                            *
REM *      Se l'inerzia della sezione varia secondo COSENOW:               *
REM *      W          Esponente del coseno                                 *
REM *      INERZC   Momento di inerzia al centro                           *
REM *      Se i carichi sono descritti da funzione:                        *
REM *      QUNIF   Carico verticale uniforme                               *
REM *                                                                      *
REM ************************************************************************
REM
INPUT #7, LUCE, T, E, CS, CD, NEIG, NVEC, NOI, TOLVEC
INPUT #7, ASSE$, VINCOLO$, SEZIONE$, CARICHI$
IF ASSE$ = "FUNC" THEN
  INPUT #7, FREC, RIAL
END IF
SELECT CASE VINCOLO$
CASE "INCASTRO"
  CERN = 0
CASE "1 CERNIERA"
  CERN = 1
CASE "2 CERNIERE"
  CERN = 2
CASE "3 CERNIERE"
  CERN = 3
CASE ELSE
  CLS
```

```
  PRINT "OPZIONE NON VALIDA PER LA STRINGA VINCOLO$"
END SELECT
FOR I = 1 TO CERN
  INPUT #7, DIVCERN(I)
NEXT I
IF SEZIONE$ = "PARABOLA" THEN INPUT #7, INERZS, INERZC, INERZD
IF SEZIONE$ = "COSENOW" THEN INPUT #7, W, INERZC
IF CARICHI$ = "FUNC" THEN
  INPUT #7, QUNIF
END IF
END SUB

SUB IngressoDatiArray
REM
REM  ************************************************************************
REM *                                                                        *
REM *                        I n g r e s s o D a t i A r r a y               *
REM *                                                                        *
REM  ************************************************************************
REM
REM
REM  ************************************************************************
REM *                                                                        *
REM *              Questa subroutine si occupa dell'ingresso dei dati        *
REM *                              organizzati in array                      *
REM *                                                                        *
REM  ************************************************************************
REM
REM ------------------------------------------------------------------------
REM Legge le quote dell'arco
REM ------------------------------------------------------------------------
IF ASSE$ = "DATI" THEN
  FOR I = 1 TO T + 1
    INPUT #7, Q(I)
  NEXT I
  RIAL = Q(T + 1)
  FREC = Q(T / 2 + 1) - RIAL / 2
END IF
REM ------------------------------------------------------------------------
REM Legge i carichi verticali e la spinta
REM ------------------------------------------------------------------------
IF CARICHI$ = "DATI" THEN
```

```
  FOR I = 1 TO T + 1
    INPUT #7, FVERT(I)
  NEXT I
  INPUT #7, HSPINTA
END IF
REM ---------------------------------------------------------------
REM Legge i momenti di inerzia nelle varie sezioni
REM ---------------------------------------------------------------
IF SEZIONE$ = "DATI" THEN
  FOR I = 1 TO T + 1
    INPUT #7, INERZ(I)
  NEXT I
END IF
END SUB

SUB orthog (W(), N, M, LCK)
REM
REM  ********************************************************************
REM *                                                                    *
REM *                              o r t h o g                           *
REM *                                                                    *
REM  ********************************************************************
REM
REM  ********************************************************************
REM *                                                                    *
REM * ORTONORMALIZZA I VETTORI W(N,M) TRAMITE                            *
REM * PROCEDIMENTO DI SCHMIDT, ASSUMENDO CHE LE COLONNE                  *
REM * PRIMA DI LCK NON VADANO MODIFICATE.                                *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                      *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION                    *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                        *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                        *
REM *                                                                    *
REM  ********************************************************************
FOR I = LCK TO M
  FOR J = 1 TO I
    EL = 0
    FOR K = 1 TO N
      EL = EL + W(K, J) * W(K, I)
    NEXT K
    IF I = J THEN
      D = 1 / SQR(EL)
```

```
      FOR K = 1 TO N
        W(K, I) = D * W(K, I)
      NEXT K
    ELSE
      FOR K = 1 TO N
        W(K, I) = W(K, I) - EL * W(K, J)
      NEXT K
    END IF
  NEXT J
NEXT I
END SUB

SUB premult (U(), V(), L(), LD(), NA, N, M, LCK, IRENT)
REM
REM  ***************************************************************
REM *                                                               *
REM *                          p r e m u l t                        *
REM *                                                               *
REM  ***************************************************************
REM
REM  ***************************************************************
REM *                                                               *
REM * OPERA LA MOLTIPLICAZIONE V = AU PER LE COLONNE                *
REM * DA LCK AD M DI V. A() E' MEMORIZZATA IN L() IN                *
REM * BANDA MOBILE. SE IRENT=2, ALLORA A() E' CONSIDERATA           *
REM * TRIANGOLARE BASSA.                                            *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                 *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION               *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                   *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                   *
REM *                                                               *
REM  ***************************************************************
REM
FOR I = 1 TO N
  LK = L(LD(I))
  FOR K = LCK TO M
    V(I, K) = U(I, K) * LK
  NEXT K
NEXT I
FOR I = 2 TO N
  Q = LD(I) - I
  M1 = 1 + LD(I - 1) - Q
```

```
  FOR K = LCK TO M
    IF IRENT <> 2 THEN
      FOR J = M1 TO I - 1
        V(I, K) = V(I, K) + L(J + Q) * U(J, K)
      NEXT J
    END IF
    FOR J = M1 TO I - 1
      V(J, K) = V(J, K) + L(J + Q) * U(I, K)
    NEXT J
  NEXT K
NEXT I
END SUB

SUB rand (U(), N, M, IRENT)
REM
REM  ************************************************************************
REM *                                                                        *
REM *                                  r a n d                               *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM * GENERAZIONE RANDOM DEI VETTORI IN TUTTE LE COLONNE                     *
REM * DI U, SE IRENT = 0, OPPURE SOLO NELLA COLONNA M-MA                     *
REM * SE IRENT E' DIVERSO DA ZERO.                                           *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                          *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION                        *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                            *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                            *
REM *                                                                        *
REM  ************************************************************************
REM
K = 1
IF IRENT <> 0 THEN K = M
IRENT = 1
FOR I = 1 TO N
  FOR J = K TO M
    U(I, J) = RND
  NEXT J
NEXT I
END SUB
```

```
SUB reduce (L(), LD(), NK, N)
REM
REM  ************************************************************************
REM *                                                                        *
REM *                              r e d u c e                               *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM * FATTORIZZAZIONE ALLA CHOLESKI DI UNA MATRICE L()                       *
REM * IN BANDA MOBILE, CON MATRICE DI INDIRIZZI LD()                         *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN                          *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION                        *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                            *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                            *
REM *                                                                        *
REM  ************************************************************************
REM
L(1) = SQR(L(1))
FOR I = 2 TO N
  Q = LD(I) - I
  M1 = 1 + LD(I - 1) - Q
  FOR J = M1 TO I
    NN = 0
    EL = L(J + Q)
    IF J > 1 THEN
      KK = LD(J) - J
      NN = 1 + LD(J - 1) - KK
      IF NN < M1 THEN NN = M1
      FOR K = NN TO J - 1
        EL = EL - L(Q + K) * L(K + KK)
      NEXT K
    END IF
    L(J + Q) = EL / L(LD(J))
  NEXT J
  IF EL >= 0 THEN
    L(LD(I)) = SQR(EL)
  ELSE
    CLS
    PRINT "REDUCTION FAILURE : PIVOT"; I; "NOT POSITIVE"
    STOP
```

```
  END IF
NEXT I
END SUB

SUB RigidezzeGeometriche
REM
REM  ***************************************************************
REM *                                                               *
REM *                  R i g i d e z z e G e o m e t r i c h e      *
REM *                                                               *
REM  ***************************************************************
REM
REM  ***************************************************************
REM *                                                               *
REM *      CALCOLA LA MATRICE INSTABILIZZANTE DEL LAVORO            *
REM *           DEL SECONDO ORDINE, SECONDO 6.23 E 6.24             *
REM *                                                               *
REM  ***************************************************************
REM
KAUX = (Q(T) - Q(T + 1)) / LTRAT
FOR I = 1 TO T - 2
  FSUM = 0
  FOR IT = I + 1 TO T
    FSUM = FSUM + FVERT(IT)
  NEXT IT
  FSUM = FSUM - KAUX * HSPINTA
  IK = I * (I + 1) / 2
  B(IK) = (Q(I + 1) - Q(I)) * FSUM + LTRAT * HSPINTA
  FOR J = I TO T - 2
    IK = J * (J + 1) / 2 - J + I
    B(IK) = B(IK) - (Q(T - 1) - Q(T)) * S1(I) * S1(J) * (FVERT(T) - KAUX *
       HSPINTA)
    B(IK) = B(IK) + LTRAT * S1(I) * S1(J) * HSPINTA
    B(IK) = B(IK) - (Q(T) - Q(T + 1)) * S2(I) * S2(J) * (-KAUX * HSPINTA)
    B(IK) = B(IK) + LTRAT * S2(I) * S2(J) * HSPINTA
  NEXT J
NEXT I
END SUB

SUB RigidezzeGlobali
REM
REM  ***************************************************************
```

```
REM *                                                                *
REM *                    RigidezzeGlobali                            *
REM *                                                                *
REM  ******************************************************************
REM
REM  ******************************************************************
REM *                                                                *
REM *       CALCOLA LA MATRICE GLOBALE DELLE RIGIDEZZE               *
REM *                                                                *
REM  ******************************************************************
REM ------------------------------------------------------------------
REM COSTRUZIONE DEGLI ARRAY AUSILIARI PER LE
REM ROTAZIONI DEGLI ULTIMI DUE TRATTI
REM ------------------------------------------------------------------
FOR I = 1 TO T - 2
  DEN = 2 * Q(T) - Q(T - 1) - Q(T + 1)                    'cfr.6.9
  S1(I) = (Q(T + 1) - Q(T) + Q(I) - Q(I + 1)) / DEN       'cfr.6.9
  S2(I) = (Q(T - 1) - Q(T) - Q(I) + Q(I + 1)) / DEN       'cfr.6.9
NEXT I
REM ------------------------------------------------------------------
REM COSTRUZIONE DELLA DIAGONALE PRINCIPALE                 cfr. 6.16
REM ------------------------------------------------------------------
FOR I = 1 TO T - 2
  IK = I * (I + 1) / 2
  K(IK) = LOCSTIF(I) + LOCSTIF(I + 1)
  K(IK) = K(IK) + (LOCSTIF(T - 1) + LOCSTIF(T)) * S1(I) ^ 2
  K(IK) = K(IK) + (LOCSTIF(T) + LOCSTIF(T + 1)) * S2(I) ^ 2
  K(IK) = K(IK) - 2 * LOCSTIF(T) * S1(I) * S2(I)
  IF I = T - 2 THEN K(IK) = K(IK) - 2 * LOCSTIF(T - 1) * S1(T - 2)
NEXT I
REM ------------------------------------------------------------------
REM COSTRUZIONE DEI TERMINI FUORI DIAGONALE
REM Primi tre termini della 6.17
REM ------------------------------------------------------------------
FOR I = 1 TO T - 2
  FOR J = I + 1 TO T - 2
    IK = J * (J + 1) / 2 - J + I
    K(IK) = LOCSTIF(T - 1) * S1(I) * S1(J)
    K(IK) = K(IK) + LOCSTIF(T) * (S2(I) - S1(I)) * (S2(J) - S1(J))
    K(IK) = K(IK) + LOCSTIF(T + 1) * S2(I) * S2(J)
  NEXT J
NEXT I
```

```
REM ------------------------------------------------------------
REM MODIFICA DELLE POSIZIONI (I,I+1)
REM Penultimo termine della 6.17
REM ------------------------------------------------------------
FOR I = 1 TO T - 3
  J = I + 1
  IK = J * (J + 1) / 2 - J + I
  K(IK) = K(IK) - LOCSTIF(J)
NEXT I
REM ------------------------------------------------------------
REM MODIFICA DELL'ULTIMA COLONNA
REM Ultimo termine della 6.17
REM ------------------------------------------------------------
FOR I = 1 TO T - 3
  J = T - 2
  IK = J * (J + 1) / 2 - J + I
  K(IK) = K(IK) - LOCSTIF(T - 1) * S1(I)
NEXT I
REM ------------------------------------------------------------
REM COSTRUISCE LA MATRICE DEGLI INDIRIZZI
REM ------------------------------------------------------------
FOR I = 1 TO T - 2
  INDK(I) = I * (I + 1) / 2
NEXT I
END SUB

SUB RigidezzeLocali
REM
REM  *********************************************************************
REM *                                                                     *
REM *                        R i g i d e z z e L o c a l i                *
REM *                                                                     *
REM  *********************************************************************
REM
REM  *********************************************************************
REM *                                                                     *
REM *            Questa subroutine si occupa della costruzione del        *
REM *                 vettore delle rigidezze locali nelle celle          *
REM *                                                                     *
REM  *********************************************************************
REM
FOR I = 2 TO T
```

```
  Q1 = (Q(I) + Q(I - 1)) / 2
  Q2 = (Q(I) + Q(I + 1)) / 2
  Q = Q2 - Q1
  LUNGH = SQR(LTRAT ^ 2 + Q ^ 2)
  IN = (INERZ(I - 1) + 2 * INERZ(I) + INERZ(I + 1)) / 4
  LOCSTIF(I) = E * IN / LUNGH                                    'cfr 6.2
NEXT I
Q = Q(2) / 2
LUNGH = SQR((LTRAT / 2) ^ 2 + Q ^ 2)
LOCSTIF(1) = E * INERZ(1) / LUNGH
LOCSTIF(1) = LOCSTIF(1) / (1 + LOCSTIF(1) * CS)                  'cfr. 6.5
Q = (Q(T) - Q(T + 1)) / 2
LUNGH = SQR((LTRAT / 2) ^ 2 + Q ^ 2)
LOCSTIF(T + 1) = E * INERZ(T + 1) / LUNGH
LOCSTIF(T + 1) = LOCSTIF(T + 1) / (1 + LOCSTIF(T + 1) * CD)      'cfr. 6.5
REM
REM ---------------------------------------------------------------
REM Si introducono eventuali cerniere
REM ---------------------------------------------------------------
FOR I = 1 TO CERN
  LOCSTIF(DIVCERN(I)) = 0
NEXT I
END SUB

SUB sivib2 (G(), GD(), L(), LD(), BD(), ERRO(), U(), V(), W(), NA, NK, N, M,
    NRQD, NOI, TOLVEC)
REM
REM ***************************************************************
REM *                                                             *
REM *                          S I V I B 2                        *
REM *                                                             *
REM ***************************************************************
REM
REM ***************************************************************
REM *                                                             *
REM * CALCOLO DEI PRIMI AUTOVALORI ED AUTOVETTORI DEL             *
REM * PROBLEMA GENERALIZZATO CON IL METODO DI                     *
REM * ITERAZIONE SIMULTANEA DI A. JENNINGS.                       *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN               *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION             *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                 *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                 *
```

```
REM *                                                                *
REM  ************************************************************************
REM
IRENT = 0
REM FATTORIZZAZIONE DI K ALLA CHOLESKI
CALL reduce(L(), LD(), NK, N)
LT = 0
LCK = 1
INTER = 1
IF IRENT = 2 THEN
  CALL premult(W(), U(), L(), LD(), NK, M, N, 1, IRENT)
END IF
REM GENERA VETTORI U RANDOM
CALL rand(U(), N, M, IRENT)
REM ORTONORMALIZZA I VETTORI U
CALL orthog(U(), N, M, 1)
REM SOSTITUZIONE ALL'INDIETRO L(T)X = U
DO WHILE LT = 0
  CALL backsub(U(), V(), L(), LD(), NK, N, M, LCK)
  REM PREMOLTIPLICAZIONE Y = MX
  CALL premult(V(), W(), G(), GD(), NA, N, M, LCK, 0)
  REM SOSTITUZIONE IN AVANTI LV=Y
  CALL forsub(W(), V(), L(), LD(), NK, N, M, LCK)
  REM ORDINA I VETTORI
  CALL decouple(U(), V(), W(), BD(), N, M, LCK, TOLVEC)
  REM GENERA I VETTORI W RANDOM
  CALL rand(W(), N, M, IRENT)
  REM ORTONORMALIZZA I VETTORI W
  CALL orthog(W(), N, M, LCK)
  REM STIMA DEI VETTORI DELL'ERRORE
  CALL errore(U(), W(), BD(), ERRO(), N, M, NRQD, LCK, TOLVEC, LT)
  REM MONITOR PRINTINGS
  PRINT "AUTOVALORI DOPO "; INTER; " ITERAZIONI"
  PRINT : FOR I = 1 TO M: PRINT BD(I): NEXT I: PRINT
  PRINT "ERRORI": PRINT : FOR I = 1 TO M: PRINT ERRO(I): NEXT I
  FOR I = 1 TO N
    FOR J = LCK TO M
      U(I, J) = W(I, J)
    NEXT J
  NEXT I
  IF NOI < INTER THEN
    CLS
```

```
    PRINT "CONVERGENZA NON RAGGIUNTA"
    STOP
  END IF
  INTER = INTER + 1
LOOP
REM SOSTITUZIONE ALL'INDIETRO
CALL backsub(U(), W(), L(), LD(), NK, N, M, 1)
FOR J = 1 TO M
  EL = 0
  FOR I = 1 TO N
    IF ABS(W(I, J)) > EL THEN EL = ABS(W(I, J))
  NEXT I
  EL = 1 / EL
  FOR I = 1 TO N
    W(I, J) = W(I, J) * EL
  NEXT I
NEXT J
FOR I = 1 TO NRQD
  BD(I) = 1 / BD(I)
NEXT I
END SUB

SUB UscitaDati  REM
REM  ************************************************************************
REM *                                                                        *
REM *                              U s c i t a D a t i                       *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM *        Questa subroutine si occupa dell'uscita (su video, su stampa    *
REM *                          e su file) dei dati di ingresso               *
REM *                                                                        *
REM  ************************************************************************
REM
FOR ITER = 1 TO ITER1
  PRINT #ITER,"  ================================="
  PRINT #ITER, "PROGRAMMA AINS - ANALISI DI STABILITA'
  PRINT #ITER, TITLE$
  PRINT #ITER, "CALCOLO CARICHI CRITICI
  PRINT #ITER,"  ================================="
```

```
PRINT #ITER,
IF CERN = 0 THEN PRINT #ITER, "ARCO INCASTRATO"
IF CERN = 1 THEN PRINT #ITER, "ARCO AD 1 CERNIERA"
IF CERN = 2 THEN PRINT #ITER, "ARCO A 2 CERNIERE"
IF CERN = 3 THEN PRINT #ITER, "ARCO A 3 CERNIERE"
PRINT #ITER, "FILE DEI DATI ="; FILEINPUT$
PRINT #ITER, "LUCE DELL'ARCO = "; LUCE
PRINT #ITER, "FRECCIA DELL'ARCO = "; FREC
PRINT #ITER,
PRINT #ITER, "FILE DEI DATI ="; FILEINPUT$
PRINT #ITER, "LUCE DELL'ARCO = "; LUCE
PRINT #ITER, "FRECCIA DELL'ARCO = "; FREC
PRINT #ITER, "RIALZO A DESTRA ="; RIAL
PRINT #ITER, "MODULO DI YOUNG ="; E
PRINT #ITER, "CEDIBILITA' ANGOLARE A SINISTRA = "; CS
PRINT #ITER, "CEDIBILITA' ANGOLARE A DESTRA = "; CD
PRINT #ITER, "NUMERO DI AUTOVALORI DA CALCOLARE = "; NEIG
PRINT #ITER, "ERRORE AMMESSO SUGLI AUTOVETTORI = "; TOLVEC
PRINT #ITER, "DIMENSIONE DEL SOTTOSPAZIO = "; NVEC
PRINT #ITER, "NUMERO MASSIMO DI ITERAZIONI = "; NOI
PRINT #ITER, "NUMERO DI TRATTI = "; T
IF CERN >= 1 THEN
  PRINT #ITER, "PRIMA CERNIERA ALLA DIVIDENTE "; DIVCERN(1)
END IF
IF CERN >= 2 THEN
  PRINT #ITER, "SECONDA CERNIERA ALLA DIVIDENTE "; DIVCERN(2)
END IF
IF CERN = 3 THEN
  PRINT #ITER, "TERZA CERNIERA ALLA DIVIDENTE "; DIVCERN(3)
END IF
IF SEZIONE$ = "PARABOLA" THEN
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO A SINIS-
     TRA = "; INERZS
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO AL CEN-
     TRO = "; INERZC
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO A DE-
     STRA = "; INERZD
END IF
IF SEZIONE$ = "COSENOW" THEN
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO AL CEN-
     TRO = "; INERZC
END IF
```

```
PRINT #ITER,
IF CARICHI$ = "FUNC" THEN
  PRINT #ITER, "CARICO VERTICALE UNIFORME = "; QUNIF
END IF
REM ------------------------------------------------------------
REM STAMPA LE QUOTE DELL'ARCO PER CIASCUNA DIVIDENTE
REM ------------------------------------------------------------
IF ASSE$ = "DATI" THEN
PRINT #ITER,
  PRINT #ITER," ==================================="
  PRINT #ITER," =                                 ="
  PRINT #ITER, "=      TABELLA DELLE QUOTE DELL'ARCO      ="
  PRINT #ITER," =                                 ="
  PRINT #ITER," ==================================="
  PRINT #ITER,
  PRINT #ITER, " ---------------------------------------------"
  PRINT #ITER, " DIVIDENTE                QUOTA               "
  PRINT #ITER, " ---------------------------------------------"
  PRINT #ITER,      FOR I = 1 TO T + 1
    PRINT #ITER, TAB(1); I; TAB(34); Q(I)
  NEXT I
END IF
IF CARICHI$ = "DATI" THEN
  REM ----------------------------------------------------------
  REM STAMPA I CARICHI PER CIASCUNA DIVIDENTE
  REM ----------------------------------------------------------
  PRINT #ITER, "SPINTA ORIZZONTALE = "; HSPINTA
  PRINT #ITER,
  PRINT #ITER," ==================================="
  PRINT #ITER," =                                 ="
  PRINT #ITER, "=      CARICHI VERTICALI SULL'ARCO      ="
  PRINT #ITER," =                                 ="
  PRINT #ITER," ==================================="
  PRINT #ITER,
  PRINT #ITER, " ---------------------------------------------"
  PRINT #ITER, " DIVIDENTE          CARICO VERTICALE          "
  PRINT #ITER, " ---------------------------------------------"
  PRINT #ITER,
  FOR I = 1 TO T + 1
    PRINT #ITER, TAB(2); I; TAB(20); FVERT(I)
  NEXT I
END IF
```

```
  IF SEZIONE$ = "DATI" THEN
    REM ---------------------------------------------------------
    REM STAMPA I MOMENTI DI INERZIA PER CIASCUNA DIVIDENTE
    REM ---------------------------------------------------------
    PRINT #ITER," ==================================="
    PRINT #ITER," =                                 ="
    PRINT #ITER, "=  MOMENTI DI INERZIA DELLA SEZIONE RETTA  ="
    PRINT #ITER," =                                 ="
    PRINT #ITER," ==================================="
    PRINT #ITER,
    PRINT #ITER, " ------------------------------------------"
    PRINT #ITER, " DIVIDENTE          MOMENTO DI INERZIA     "
    PRINT #ITER, " ------------------------------------------"
    PRINT #ITER,
    FOR I = 1 TO T + 1
      PRINT #ITER, TAB(2); I; TAB(20); INERZ(I)
    NEXT I
  END IF
NEXT ITER
END SUB

SUB UscitaRisultati
REM
REM  ********************************************************************
REM *                                                                    *
REM *                        U s c i t a R i s u l t a t i               *
REM *                                                                    *
REM  ********************************************************************
REM
REM  ********************************************************************
REM *                                                                    *
REM *       Questa subroutine si occupa dell'uscita (su video, su stampa *
REM *                          e su file) dei risultati                  *
REM *                                                                    *
REM  ********************************************************************
FOR ITER = ITER1 + 1 TO ITER2
  PRINT #ITER, "CARICHI CRITICI"
  PRINT #ITER,
  FOR I = 1 TO NEIG
    PRINT #ITER, TAB(2); I; TAB(15); EIG(I)
  NEXT I
  FOR I = 1 TO NEIG
```

```
    PRINT #ITER,
    PRINT #ITER, "AUTOVETTORE N. "; I
    PRINT #ITER,
    FOR K = 1 TO T - 2
      PRINT #ITER, TAB(2); K; TAB(15); VECT(K, I)
    NEXT K
  NEXT I
  PRINT #ITER,
  FOR I = 1 TO NEIG
    PRINT #ITER,
    PRINT #ITER, "SPOSTAMENTI VERTICALI CONNESSI ALL'AUTOVET-
       TORE N. "; I
    PRINT #ITER,
    FOR K = 1 TO T + 1
      PRINT #ITER, TAB(2); K; TAB(15); SPOS(K, I)
    NEXT K
  NEXT I
  PRINT #ITER,
  PRINT #ITER,
  FOR I = 1 TO NEIG
    PRINT #ITER,
    PRINT #ITER, "SPOSTAMENTI ORIZZONTALI CONNESSI ALL'AUTO-
       VETTORE N. "; I
    PRINT #ITER,
    FOR K = 1 TO T + 1
      PRINT #ITER, TAB(2); K; TAB(15); SBAN(K, I)
    NEXT K
  NEXT I
  PRINT #ITER,
  PRINT #ITER, "TEMPO DI ESECUZIONE = "; TIME2; " SEC."
  PRINT #ITER, "RUN DEL "; DATE$; " ORE "; TIME$
NEXT ITER
END SUB
```

Appendice 6.2. Il programma AS2

```
DEFDBL A-Z
DECLARE SUB Momenti ()
DECLARE SUB multcc (A#(), B#(), RA#, CA#, RB#, CB#, C#())
DECLARE SUB multrc (A#(), B#(), RA#, CA#, RB#, CB#, C#())
DECLARE SUB forsub (U(), V(), L(), LD(), NK, N, M, LCK)
DECLARE SUB reduce (L(), LD(), NK, N)
DECLARE SUB backsub (U(), V(), L(), LD(), NK, N, M, LCK)
DECLARE SUB solve (A(), AD(), NA, U(), N, M)
DECLARE SUB Soluzione ()
DECLARE SUB Spostamenti2 ()
DECLARE SUB SecondoMembro ()
DECLARE SUB Trasferimento ()
DECLARE SUB RigidezzeGeometriche ()
DECLARE SUB RigidezzeGlobali ()
DECLARE SUB RigidezzeLocali ()
DECLARE SUB Geometria ()
DECLARE SUB UscitaDati ()
DECLARE SUB UscitaRisultati ()
DECLARE SUB IngressoDati ()
DECLARE SUB IngressoDatiArray ()
DECLARE SUB OpenFiles ()
REM  ************************************************************************
REM *                                                                        *
REM *                      P R O G R A M M A      AS2                        *
REM *                                                                        *
REM *                                 di                                     *
REM *                                                                        *
REM *                  V I N C E N Z O      F R A N C I O S I                *
REM *                                                                        *
REM *                Release 1.0.0. Gennaio 1988 (in BASIC HP)               *
REM *              Release 3.0.0. Agosto 1994 (in Microsoft QBasic)          *
REM *                                                                        *
```

```
REM  ************************************************************************
REM
REM
REM  ************************************************************************
REM *                                                                        *
REM *        Questo programma calcola spostamenti e momenti di un arco       *
REM *                          in teoria del secondo ordine                  *
REM *                                                                        *
REM  ************************************************************************
OPTION BASE 1
REM
REM  ************************************************************************
REM *                                                                        *
REM *                              C O M M O N                               *
REM *                                                                        *
REM  ************************************************************************
REM
REM
COMMON SHARED LUCE, T, E, CS, CD, FREC, RIAL
COMMON SHARED INERZS, INERZC, INERZD, WEXP, QUNIF, HSPINTA
COMMON SHARED LTRAT, CERN, ALPHA, DELTATN,MULT
COMMON SHARED ITER1, ITER2, TIME2, B1, C1,TT
COMMON SHARED FVERT() AS DOUBLE, Q() AS DOUBLE
COMMON SHARED INERZ() AS DOUBLE, INCL() AS DOUBLE
COMMON SHARED LOCSTIF() AS DOUBLE, DIVCERN() AS DOUBLE
COMMON SHARED S1() AS DOUBLE, S2() AS DOUBLE
COMMON SHARED K() AS DOUBLE, B() AS DOUBLE, INDK() AS DOUBLE
COMMON SHARED FVACC() AS DOUBLE, FOACC() AS DOUBLE
COMMON SHARED V() AS DOUBLE, W() AS DOUBLE, VT() AS DOUBLE
COMMON SHARED WT() AS DOUBLE, C() AS DOUBLE, CV() AS DOUBLE
COMMON SHARED CO() AS DOUBLE, CTERMICA() AS DOUBLE
COMMON SHARED CTERM2() AS DOUBLE, M() AS DOUBLE
COMMON SHARED TITLE$, STAMPA$, STAMPARIS$, FILEINGR$, FILEUSC$
COMMON SHARED ASSE$, VINCOLO$, SEZIONE$, CARICHI$, FILEINPUT$
REM
REM  ************************************************************************
REM *                                                                        *
REM *                    DIMENSIONA GLI ARRAY STATICI                        *
REM *                                                                        *
REM  ************************************************************************
REM
DIM TITLE$(100), TITLE1$(100), STAMPA$(10), STAMPARIS$(10)
```

```
DIM FILEINGR$(10), FILEUSC$(10), DIVCERN(3) AS DOUBLE
REM
REM  ************************************************************************
REM *                                                                        *
REM *              DEFINISCE ALCUNE LEGGI DI VARIAZIONE                      *
REM *                                                                        *
REM  ************************************************************************
REM DEF FNQ (Z) = RIAL * Z / LUCE + 4 * FREC / LUCE ^ 2 * Z * (LUCE -
    Z)
DEF FNINERZ1 (Z)
  FAUX = INERZS * (LUCE - Z) / LUCE + INERZD * Z / LUCE
  FAUX = FAUX - 2 * (INERZS + INERZD - 2 * INERZC) / LUCE ^ 2 * Z *
    (LUCE - Z)
  FNINERZ1 = FAUX
END DEF
DEF FNA (Z) = 4 * FREC / LUCE ^ 2 * (LUCE - 2 * Z) + RIAL / LUCE
DEF FNINERZ2 (Z) = INERZC * COS(ATN(FNA(Z))) ^ WEXP
CLS
REM
REM  ************************************************************************
REM *                                                                        *
REM *              LEGGE IL NOME DEL FILE DI INPUT DATI                      *
REM *                                                                        *
REM  ************************************************************************
REM
PRINT "NOME DEL FILE DI INPUT"
INPUT FILEINPUT$
TIME1 = TIMER
OPEN FILEINPUT$ FOR INPUT AS #7
REM ------------------------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER L'INGRESSO DEI DATI
REM ------------------------------------------------------------------------
CALL IngressoDati
TT = (T - 2) * (T - 1) / 2
LTRAT = LUCE / T
REM
REM  ************************************************************************
REM *                                                                        *
REM *                  REDIMENSIONA ARRAY DINAMICI                           *
REM *                                                                        *
REM  ************************************************************************
REM
```

```
REDIM FVERT(T + 1, 1) AS DOUBLE, Q(T + 1) AS DOUBLE
REDIM FVACC(T + 1, 1) AS DOUBLE, FOACC(T + 1, 1) AS DOUBLE
REDIM INERZ(T + 1) AS DOUBLE, LOCSTIF(T + 1) AS DOUBLE
REDIM S1(T - 2) AS DOUBLE, S2(T - 2) AS DOUBLE
REDIM INDK(T - 2) AS DOUBLE, K(TT) AS DOUBLE
REDIM B(TT) AS DOUBLE, CTERM2(T - 2) AS DOUBLE
REDIM V(T + 1, 1) AS DOUBLE, W(T + 1, 1) AS DOUBLE
REDIM C(T - 2, 1) AS DOUBLE, CV(T - 2, 1) AS DOUBLE
REDIM CO(T - 2, 1) AS DOUBLE, CTERMICA(T - 2) AS DOUBLE
REDIM VT(T + 1, T - 2) AS DOUBLE, WT(T + 1, T - 2) AS DOUBLE
REDIM M(T + 1) AS DOUBLE,, INCL(T + 1) AS DOUBLE
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER L'INGRESSO DEI DATI IN ARRAY
REM ---------------------------------------------------------
CALL IngressoDatiArray
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER RIEMPIRE ALCUNI ARRAY
REM ---------------------------------------------------------
CALL Geometria
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER STAMPARE I DATI DI INGRESSO
REM ---------------------------------------------------------
TIME2 = TIMER - TIME1
CALL UscitaDati
TIME1 = TIMER
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER RIEMPIRE LA MATRICE
REM DELLE RIGIDEZZE LOCALI DELLE CELLE
REM ---------------------------------------------------------
CALL RigidezzeLocali
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER ASSEMBLARE LA MATRICE
REM DI RIGIDEZZA GLOBALE
REM ---------------------------------------------------------
CALL RigidezzeGlobali
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER CALCOLARE LA MATRICE
REM DI RIGIDEZZA GEOMETRICA
REM ---------------------------------------------------------
CALL RigidezzeGeometriche
REM ---------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER COSTRUIRE LE MATRICI
```

```
REM DI TRASFERIMENTO
REM ---------------------------------------------------------------
CALL Trasferimento
REM ---------------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER IL CALCOLO DEI TERMINI NOTI
REM ---------------------------------------------------------------
CALL SecondoMembro
REM ---------------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER LA SOLUZIONE DELLE EQUAZIONI
REM DI EQUILIBRIO
REM ---------------------------------------------------------------
CALL Soluzione
REM ---------------------------------------------------------------
REM CHIAMA LE SUBROUTINE PER IL CALCOLO DI SPOSTAMENTI
REM E MOMENTI
REM ---------------------------------------------------------------
CALL Spostamenti2
CALL Momenti
REM ---------------------------------------------------------------
REM CHIAMA LA SUBROUTINE PER LA STAMPA DEI RISULTATI
REM ---------------------------------------------------------------
TIME2 = TIME2 + TIMER - TIME1
CALL UscitaRisultati
REM ---------------------------------------------------------------
REM CHIUDE TUTTI I BUFFER TEMPORANEI
REM ---------------------------------------------------------------
CLOSE END

SUB IngressoDati
REM
REM  ************************************************************************
REM *                                                                        *
REM *                          I n g r e s s o D a t i                       *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM *          Questa subroutine si occupa dell'ingresso dei dati            *
REM *                      non organizzati in array                          *
REM *                                                                        *
REM  ************************************************************************
```

```
REM  ************************************************************************
REM *                                                                        *
REM *    LEGGE IL TITOLO DELLA STRUTTURA                                     *
REM *    IL FLAG DI STAMPA PER LE VARIABILI DI INGRESSO                      *
REM *    IL FLAG DI STAMPA PER LE VARIABILI DI USCITA                        *
REM *      - CARTA → I DATI VENGONO STAMPATI                                 *
REM *      - VIDEO → I DATI VENGONO INVIATI A VIDEO                          *
REM *      - FILES → I DATI VENGONO MEMORIZZATI SU                           *
REM *          FILES DA SPECIFICARE                                          *
REM *      - VIDEOFILES → I DATI VENGONO INVIATI A                           *
REM *          VIDEO E MEMORIZZATI SU FILES                                  *
REM *      - CARTAFILES → I DATI VENGONO STAMPATI                            *
REM *          CARTA E MEMORIZZATI SU FILES                                  *
REM *      - TUTTO → I DATI VENGONO INVIATI A VIDEO,                         *
REM *          STAMPATI SU CARTA E MEMORIZZATI SU FILES                      *
REM *                                                                        *
REM  ************************************************************************
REM
INPUT #7, TITLE$
INPUT #7, STAMPA$
INPUT #7, STAMPARIS$
IF STAMPA$ = "FILES" OR STAMPA$ = "VIDEOFILES" OR STAMPA$ = "CAR-
    TAFILES" OR STAMPA$ = "TUTTO" THEN
  INPUT #7, FILEINGR$
END IF
IF STAMPARIS$ = "FILES" OR STAMPARIS$ = "VIDEOFILES" OR STAM-
    PARIS$ = "CARTAFILES" OR STAMPARIS$ = "TUTTO" THEN
  INPUT #7, FILEUSC$
END IF
CALL OpenFiles
REM
REM  ************************************************************************
REM *                                                                        *
REM *    LEGGE:                                                              *
REM *      LUCE    Luce dell'arco                                            *
REM *      T       Numero di coordinate lagrangiane                          *
REM *      E       Modulo di Young                                           *
REM *      CS      Cedibilita' angolare a sinistra                           *
REM *      CD      Cedibilita' angolare a destra                             *
REM *      ALPHA   Coefficiente di dilatazione termica                       *
REM *      DELTATN Variazione termica                                        *
REM *      MULT    Moltiplicatore del peso proprio                           *
```

```
REM *      ASSE$   Se l'asse dell'arco e' descritto da una                *
REM *              funzione, ASSE$ = FUNZ, se invece e' dato              *
REM *              per punti, allora ASSE$ = DATI                         *
REM *      VINCOLO$   Può assumere i seguenti valori:                     *
REM *              INCASTRO                                               *
REM *              1 CERNIERA                                             *
REM *              2 CERNIERE                                             *
REM *              3 CERNIERE                                             *
REM *      SEZIONE$   Se la variazione del momento di inerzia             *
REM *          della sezione retta e' data per punti,                     *
REM *          allora SEZIONE$ = DATI. Altrimenti sono                    *
REM *          disponibili le seguenti opzioni:                           *
REM *              = PARABOLA se le inerzie variano                       *
REM *              con legge parabolica                                   *
REM *              = COSENOW se le inerzie variano                        *
REM *              con la legge del coseno                                *
REM *      CARICHI$   Se i carichi sull'arco sono descritti da            *
REM *              funzione, CARICHI$ = FUNZ, se invece                   *
REM *              sono dati per punti, allora CARICHI$ = DATI            *
REM *      Se l'asse e' definito da funzione:                             *
REM *      FREC   Freccia dell'arco                                       *
REM *      RIAL   Rialzo a destra                                         *
REM *      Se l'inerzia della sezione varia secondo PARABOLA:             *
REM *      INERZS   Momento di inerzia a sinistra                         *
REM *      INERZC   Momento di inerzia al centro                          *
REM *      INERZD   Momento di inerzia a destra                           *
REM *      Se l'inerzia della sezione varia secondo COSENOW:              *
REM *      WEXP   Esponente del coseno                                    *
REM *      INERZC   Momento di inerzia al centro                          *
REM *      Se i carichi sono descritti da funzione:                       *
REM *      QUNIF   Carico verticale uniforme                              *
REM *                                                                     *
REM ***********************************************************************
INPUT #7, LUCE, T, E, CS, CD, ALPHA, DELTATN, MULT
INPUT #7, ASSE$, VINCOLO$, SEZIONE$, CARICHI$
IF ASSE$ = "FUNC" THEN
  INPUT #7, FREC, RIAL
END IF
SELECT CASE VINCOLO$
CASE "INCASTRO"
  CERN = 0
CASE "1 CERNIERA"
```

```
  CERN = 1
CASE "2 CERNIERE"
  CERN = 2
CASE "3 CERNIERE"
  CERN = 3
CASE ELSE
  CLS
  PRINT "OPZIONE NON VALIDA PER LA STRINGA VINCOLO$"
END SELECT
FOR I = 1 TO CERN
  INPUT #7, DIVCERN(I)
NEXT I
IF SEZIONE$ = "PARABOLA" THEN INPUT #7, INERZS, INERZC, INERZD
IF SEZIONE$ = "COSENOW" THEN INPUT #7, WEXP, INERZC
IF CARICHI$ = "FUNC" THEN
  INPUT #7, QUNIF
END IF
END SUB

SUB IngressoDatiArray  REM
REM  ************************************************************************
REM *                                                                        *
REM *                        I n g r e s s o D a t i A r r a y               *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM *              Questa subroutine si occupa dell'ingresso dei dati        *
REM *                              organizzati in array                      *
REM *                                                                        *
REM  ************************************************************************
REM ------------------------------------------------------------------------
REM Legge le quote dell'arco
REM ------------------------------------------------------------------------
IF ASSE$ = "DATI" THEN
  FOR I = 1 TO T + 1
    INPUT #7, Q(I)
  NEXT I
  RIAL = Q(T + 1)
  FREC = Q(T / 2 + 1) - RIAL / 2
END IF
```

```
REM
  REM ---------------------------------------------------------------
  REM Legge i carichi verticali fissi
  REM ---------------------------------------------------------------
IF CARICHI$ = "DATI" THEN
  REM ---------------------------------------------------------------
  REM Legge i carichi verticali fissi
  REM ---------------------------------------------------------------
  FOR I = 1 TO T + 1
    INPUT #7, FVERT(I, 1)
  NEXT I
  REM ---------------------------------------------------------------
  REM Legge i carichi verticali accidentali
  REM ---------------------------------------------------------------
  FOR I = 1 TO T + 1
    INPUT #7, FVACC(I, 1)
  NEXT I
  REM ---------------------------------------------------------------
  REM Legge i carichi orizzontali accidentali
  REM ---------------------------------------------------------------
  FOR I = 1 TO T + 1
    INPUT #7, FOACC(I, 1)
  NEXT I
  REM ---------------------------------------------------------------
  REM Legge la spinta
  REM ---------------------------------------------------------------
  INPUT #7, HSPINTA
END IF
IF SEZIONE$ = "DATI" THEN
  REM ---------------------------------------------------------------
  REM Legge i momenti di inerzia nelle sezioni
  REM ---------------------------------------------------------------
  FOR I = 1 TO T + 1
    INPUT #7, INERZ(I)
  NEXT I
END IF
END SUB

SUB Momenti
REM
REM ***************************************************************
REM *                                                             *
```

```
REM *                          M o m e n t i                          *
REM *                                                                 *
REM  *******************************************************************
REM
REM  *******************************************************************
REM *                                                                 *
REM *              CALCOLA I MOMENTI FLETTENTI                        *
REM *                                                                 *
REM  *******************************************************************
FOR I = 2 TO T - 2
  M(I) = (C(I, 1) - C(I - 1, 1)) * LOCSTIF(I)
NEXT I
M(1) = LOCSTIF(1) * C(1, 1)
F1 = 0
F2 = 0
FOR I = 1 TO T - 2
  F1 = F1 + C(I, 1) * S1(I)
  F2 = F2 + C(I, 1) * S2(I)
NEXT I
IF DELTATN <> 0 THEN
  F1 = F1 + B1 * ALPHA * DELTATN
  F2 = F2 + C1 * ALPHA * DELTATN
END IF
M(T - 1) = LOCSTIF(T - 1) * (F1 - C(T - 2, 1))
M(T) = LOCSTIF(T) * (F2 - F1)
M(T + 1) = -LOCSTIF(T + 1) * F2
END SUB

SUB multcc (A(), B(), RA, CA, RB, CB, C())
REM
REM  *******************************************************************
REM *                                                                 *
REM *                          m u l t c c                            *
REM *                                                                 *
REM  *******************************************************************
REM
REM  *******************************************************************
REM *                                                                 *
REM * OPERA IL PRODOTTO MATRICIALE, COLONNA PER COLONNA,              *
REM * TRA LA MATRICE A, AD RA RIGHE E CA COLONNE,                     *
REM * E LA MATRICE B AD RB RIGHE E CB COLONNE,                        *
REM * OTTENENDO LA MATRICE C, A CA RIGHE E CB COLONNE.                *
```

```
REM * 1. IL PRODOTTO PUO' EFFETTUARSI SOLO SE RA = RB          *
REM * 2. IL PRODOTTO EQUIVALE ALL'USUALE PRODOTTO TRA LA      *
REM * TRASPOSTA DI A E B.                                      *
REM *                                                          *
REM ************************************************************
IF RA <> RB THEN
  CLS
  PRINT "NON PUO' EFFETTUARSI IL DESIDERATO PRODOTTO MATRI-
    CIALE"
  STOP
END IF
FOR I = 1 TO CA
  FOR J = 1 TO CB
    C(I, J) = 0
    FOR K = 1 TO RA
      C(I, J) = C(I, J) + A(K, I) * B(K, J)
    NEXT K
  NEXT J
NEXT I
END SUB

SUB multrc (A(), B(), RA, CA, RB, CB, C())
REM
REM ************************************************************
REM *                                                          *
REM *                        m u l t r c                       *
REM *                                                          *
REM ************************************************************
REM
REM ************************************************************
REM *                                                          *
REM * OPERA IL PRODOTTO MATRICIALE, RIGA PER COLONNA,          *
REM * TRA LA MATRICE A, AD RA RIGHE E CA COLONNE,              *
REM * E LA MATRICE B AD RB RIGHE E CB COLONNE,                 *
REM * OTTENENDO LA MATRICE C, A CA RIGHE E CB COLONNE.         *
REM * 1. IL PRODOTTO PUO' EFFETTUARSI SOLO SE CA = RB          *
REM *                                                          *
REM ************************************************************
IF CA <> RB THEN
  CLS
  PRINT "NON PUO' EFFETTUARSI IL DESIDERATO PRODOTTO MATRI-
    CIALE"
```

```
  STOP
END IF
FOR I = 1 TO CA
  FOR J = 1 TO CB
    C(I, J) = 0
    FOR K = 1 TO RA
      C(I, J) = C(I, J) + A(I, K) * B(K, J)
    NEXT K
  NEXT J
NEXT I
END SUB

SUB SecondoMembro
REM
REM  ************************************************************************
REM *                                                                        *
REM *                          S e c o n d o M e m b r o                     *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM *        COSTRUISCE IL VETTORE DEI TERMINI NOTI DELLE                    *
REM *                        EQUAZIONI DI EQUILIBRIO                         *
REM *                                                                        *
REM  ************************************************************************
REM ------------------------------------------------------------------------
REM OPERA IL PRODOTTO MATRICIALE VT(t)*FVACC
REM ------------------------------------------------------------------------
CALL multcc(VT(), FVACC(), T + 1, T - 2, T + 1, 1, CV())
REM ------------------------------------------------------------------------
REM OPERA IL PRODOTTO MATRICIALE WT(t)*FOACC
REM ------------------------------------------------------------------------
CALL multcc(WT(), FOACC(), T + 1, T - 2, T + 1, 1, CO())
FOR I = 1 TO T - 2
  C(I, 1) = CV(I, 1) + CO(I, 1)
NEXT I
IF DELTATN <> 0 THEN
  REM ----------------------------------------------------------------------
  REM EFFETTO DELLE VARIAZIONI TERMICHE
  REM ----------------------------------------------------------------------
  DEN = 2 * Q(T) - Q(T - 1) - Q(T + 1)
```

```
  B1 = (RIAL * (Q(T + 1) - Q(T)) + LTRAT * LUCE) / LTRAT / DEN
  C1 = -(RIAL * (Q(T) - Q(T - 1)) + LTRAT * LUCE) / LTRAT / DEN
  FOR I = 1 TO T - 2
    CTERMICA(I) = LOCSTIF(T - 1) * B1 * S1(I)
    CTERMICA(I) = CTERMICA(I) + LOCSTIF(T) * (C1 - B1) * (S2(I) - S1(I))
    CTERMICA(I) = CTERMICA(I) + LOCSTIF(T + 1) * C1 * S2(I)
  NEXT I
  CTERMICA(T - 2) = CTERMICA(T - 2) - LOCSTIF(T - 1) * B1
  FOR I = 1 TO T - 2
    CTERMICA(I) = CTERMICA(I) * ALPHA * DELTATN
  NEXT I
  KAUX = (Q(T) - Q(T + 1)) / LTRAT
  FOR I = 1 TO T - 2
    CTERM2(I) = HSPINTA * LTRAT * (S1(I) * B1 + S2(I) * C1)
    CTERM2(I) = CTERM2(I) - FVERT(T, 1) * (Q(T - 1) - Q(T)) * S1(I) * B1
    CTERM2(I) = CTERM2(I) + KAUX * HSPINTA * (Q(T - 1) - Q(T)) * S1(I)
       * B1
    CTERM2(I) = CTERM2(I) + KAUX * HSPINTA * (Q(T) - Q(T + 1)) * S2(I)
       * C1
    CTERM2(I) = CTERM2(I) * MULT * ALPHA * DELTATN
  NEXT I
  FOR I = 1 TO T - 2
    C(I, 1) = C(I, 1) - CTERMICA(I) + CTERM2(I)
  NEXT I
END IF
END SUB

SUB Soluzione
REM
REM  ***************************************************************
REM *                                                               *
REM *                          S o l u z i o n e                    *
REM *                                                               *
REM  ***************************************************************
REM
REM  ***************************************************************
REM *                                                               *
REM *       ASSEMBLA E RISOLVE LE EQUAZIONI DI EQUILIBRIO           *
REM *                                                               *
REM  ***************************************************************
FOR I = 1 TO TT
  K(I) = K(I) - MULT * B(I)
```

```
NEXT I
CALL solve(K(), INDK(), TT, C(), T - 2, 1)
END SUB

SUB solve (A(), AD(), NA, U(), N, M)
REM
REM  ************************************************************
REM *                                                            *
REM *                          s o l v e                         *
REM *                                                            *
REM  ************************************************************
REM
REM  ************************************************************
REM *                                                            *
REM * SOLUZIONE DI UN SISTEMA DI EQUAZIONI ALGEBRICO             *
REM * LINEARE, CON PIU' SECONDI MEMBRI, ADOTTANDO                *
REM * L'ALGORITMO DI A. JENNINGS                                 *
REM * IN INGRESSO                                               *
REM * A()   MATRICE MEMORIZZATA IN BANDA MOBILE                  *
REM * AD() MATRICE DEGLI INDIRIZZI DI A()                        *
REM * NA LUNGHEZZA COMPLESSIVA DI A                              *
REM * U() MATRICE DEGLI M TERMINI NOTI, PER COLONNE              *
REM * N ORDINE DEL SISTEMA                                       *
REM * M NUMERO DEI SECONDI MEMBRI                               *
REM * IN USCITA:                                                *
REM * A() MATRICE TRIANGOLARE BASSA ALLA CHOLESKI                *
REM * U() SOLUZIONI DEI SISTEMI LINEARI, PER COLONNE             *
REM *                                                          *
REM * TRASCRIZIONE BASIC DEL PROGRAMMA CONTENUTO IN              *
REM * A.JENNINGS, R.B.CORR - A SIMULTANEOUS ITERATION            *
REM * ALGORITHM FOR SYMMETRIC EIGENVALUE PROBLEMS                *
REM * INT.J.NUM.METHODS ENGNG. 10,647-663, (1976)                *
REM *                                                            *
REM  ************************************************************
REM
REM FATTORIZZAZIONE DI L ALLA CHOLESKI
CALL reduce(A(), AD(), NA, N)
REM SOSTITUZIONE IN AVANTI LV=Y
CALL forsub(U(), U(), A(), AD(), NA, N, M, 1)
REM SOSTITUZIONE ALL'INDIETRO
CALL backsub(U(), U(), A(), AD(), NA, N, M, 1)
END SUB
```

```
SUB Spostamenti2
REM
REM  ************************************************************************
REM *                                                                        *
REM *                          S p o s t a m e n t i 2                       *
REM *                                                                        *
REM  ************************************************************************
REM
REM  ************************************************************************
REM *                                                                        *
REM *    CALCOLA GLI SPOSTAMENTI VERTICALI ED ORIZZONTALI                    *
REM *                                                                        *
REM  ************************************************************************
REM ------------------------------------------------------------------------
REM OPERA IL PRODOTTO MATRICIALE VT*C
REM ------------------------------------------------------------------------
CALL multrc(VT(), C(), T - 2, T + 1, T + 1, 1, V())
REM ------------------------------------------------------------------------
REM OPERA IL PRODOTTO MATRICIALE WT*C
REM ------------------------------------------------------------------------
CALL multrc(WT(), C(), T - 2, T + 1, T + 1, 1, W())
IF DELTATN <> 0 THEN
  REM ----------------------------------------------------------------------
  REM EFFETTO DELLE VARIAZIONI TERMICHE
  REM ----------------------------------------------------------------------
  FOR I = 1 TO T + 1
    Z = (I - 1) * LTRAT
    V(I, 1) = V(I, 1) - ALPHA * DELTATN * Q(I)
    W(I, 1) = W(I, 1) + ALPHA * DELTATN * Z
  NEXT I
  V(T, 1) = V(T, 1) - LTRAT * B1 * ALPHA * DELTATN
  W(T, 1) = W(T, 1) - (Q(T) - Q(T - 1)) * B1 * ALPHA * DELTATN
  W(T + 1, 1) = W(T + 1, 1) - (Q(T) - Q(T - 1)) * B1 * ALPHA * DELTATN
  W(T + 1, 1) = W(T + 1, 1) - (Q(T + 1) - Q(T)) * C1 * ALPHA * DELTATN
END IF
END SUB

SUB Trasferimento
REM
REM  ************************************************************************
REM *                                                                        *
REM *                          T r a s f e r i m e n t o                     *
```

```
REM *                                                                *
REM  ************************************************************
REM
REM  ************************************************************
REM *                                                                *
REM *      COSTRUISCE LE MATRICI DI TRASFERIMENTO PER GLI            *
REM *                          SPOSTAMENTI                           *
REM *                                                                *
REM  ************************************************************
FOR I = 2 TO T - 1
  FOR J = 1 TO I - 1
    VT(I, J) = -LTRAT
    WT(I, J) = -(Q(J + 1) - Q(J))
  NEXT J
NEXT I
FOR J = 1 TO T - 2
  VT(T, J) = -LTRAT * (1 + S1(J))
  VT(T + 1, J) = -LTRAT * (1 + S1(J) + S2(J))
  WT(T, J) = -(Q(J + 1) - Q(J) + (Q(T) - Q(T - 1)) * S1(J))
  WT(T + 1, J) = WT(T, J) - (Q(T + 1) - Q(T)) * S2(J)
NEXT J
END SUB

SUB UscitaDati  REM
REM  ************************************************************
REM *                                                                *
REM *                        U s c i t a D a t i                     *
REM *                                                                *
REM  ************************************************************
REM
REM  ************************************************************
REM *                                                                *
REM *      Questa subroutine si occupa dell'uscita (su video, su stampa  *
REM *                  e su file) dei dati di ingresso               *
REM *                                                                *
REM  ************************************************************
REM
FOR ITER = 1 TO ITER1
  PRINT #ITER," ==================================="
  PRINT #ITER, "PROGRAMMA AS2 - ANALISI DEL SECONDO ORDINE"
  PRINT #ITER, TITLE$
  PRINT #ITER, "CALCOLO SPOSTAMENTI E MOMENTI"
```

```
PRINT #ITER,“ ==================================”
PRINT #ITER,
IF CERN = 0 THEN PRINT #ITER, "ARCO INCASTRATO"
IF CERN = 1 THEN PRINT #ITER, "ARCO AD 1 CERNIERA"
IF CERN = 2 THEN PRINT #ITER, "ARCO A 2 CERNIERE"
IF CERN = 3 THEN PRINT #ITER, "ARCO A 3 CERNIERE"
PRINT #ITER, “FILE DEI DATI =”; FILEINPUT$
PRINT #ITER, “LUCE DELL'ARCO = ”; LUCE
PRINT #ITER, “FRECCIA DELL'ARCO = ”; FREC
PRINT #ITER,
PRINT #ITER, "FILE DEI DATI =”; FILEINPUT$
PRINT #ITER, "LUCE DELL'ARCO = ”; LUCE
PRINT #ITER, "FRECCIA DELL'ARCO = ”; FREC
PRINT #ITER, "RIALZO A DESTRA =”; RIAL
PRINT #ITER, "MODULO DI YOUNG =”; E
PRINT #ITER, "CEDIBILITA' ANGOLARE A SINISTRA = ”; CS
PRINT #ITER, "CEDIBILITA' ANGOLARE A DESTRA = ”; CD
PRINT #ITER, "COEFFICIENTE DI DILATAZIONE TERMICA = ”; ALPHA
PRINT #ITER, "VARIAZIONE TERMICA = ”; DELTAT
PRINT #ITER, "NUMERO DI TRATTI = ”; T
PRINT #ITER, "SPINTA = ”; HSPINTA
IF CERN >= 1 THEN
  PRINT #ITER, "PRIMA CERNIERA ALLA DIVIDENTE ”; DIVCERN(1)
END IF
IF CERN >= 2 THEN
  PRINT #ITER, "SECONDA CERNIERA ALLA DIVIDENTE ”; DIVCERN(2)
END IF
IF CERN = 3 THEN
  PRINT #ITER, "TERZA CERNIERA ALLA DIVIDENTE ”; DIVCERN(3)
END IF
IF SEZIONE$ = "PARABOLA" THEN
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO A SINIS-
    TRA = ”; INERZS
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO AL CEN-
    TRO = ”; INERZC
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO A DE-
    STRA = ”; INERZD
END IF
IF SEZIONE$ = "COSENOW" THEN
  PRINT #ITER, "INERZIA DELLA SEZIONE RETTA DELL'ARCO AL CEN-
    TRO = ”; INERZC
END IF
```

```
  PRINT #ITER,
  IF CARICHI$ = "FUNC" THEN
    PRINT #ITER, "CARICO VERTICALE UNIFORME = "; QUNIF
  END IF
  REM ---------------------------------------------------------------
  REM STAMPA LE QUOTE DELL'ARCO PER CIASCUNA DIVIDENTE
  REM ---------------------------------------------------------------
  IF ASSE$ = "DATI" THEN
  PRINT #ITER,
    PRINT #ITER," ==========================================="
    PRINT #ITER," =                                         ="
    PRINT #ITER, "=        TABELLA DELLE QUOTE DELL'ARCO    ="
    PRINT #ITER," =                                         ="
    PRINT #ITER," ==========================================="
    PRINT #ITER,
    PRINT #ITER, " ---------------------------------------------"
    PRINT #ITER, " DIVIDENTE                QUOTA            "
    PRINT #ITER, " ---------------------------------------------"
    PRINT #ITER,       FOR I = 1 TO T + 1
      PRINT #ITER, TAB(1); I; TAB(34); Q(I)
    NEXT I
  END IF
  IF CARICHI$ = "DATI" THEN
    REM ---------------------------------------------------------------
    REM STAMPA I CARICHI PER CIASCUNA DIVIDENTE
    REM ---------------------------------------------------------------
    PRINT #ITER, "SPINTA ORIZZONTALE = "; HSPINTA
    PRINT #ITER,
    PRINT #ITER," ==========================================="
    PRINT #ITER," =                                         ="
    PRINT #ITER, "=        CARICHI VERTICALI SULL'ARCO      ="
    PRINT #ITER," =                                         ="
    PRINT #ITER," ==========================================="
    PRINT #ITER,
    PRINT #ITER, " ---------------------------------------------"
    PRINT #ITER, " DIVIDENTE  VERT. FISSO  VERT. ACC.  ORIZ. ACC "
    PRINT #ITER, " ---------------------------------------------"
    PRINT #ITER,
    FOR I = 1 TO T + 1
      PRINT #ITER, TAB(2); I; TAB(7); FVERT(I, 1); TAB(27); FVACC(I, 1);
       TAB(47); FOACC(I, 1)
    NEXT I
```

```
  END IF
  IF SEZIONE$ = "DATI" THEN
    REM ---------------------------------------------------------------
    REM STAMPA I MOMENTI DI INERZIA PER CIASCUNA DIVIDENTE
    REM ---------------------------------------------------------------
    PRINT #ITER," ==================================="
    PRINT #ITER," =                                 ="
    PRINT #ITER, "=  MOMENTI DI INERZIA DELLA SEZIONE RETTA  ="
    PRINT #ITER," =                                 ="
    PRINT #ITER," ==================================="
    PRINT #ITER,
    PRINT #ITER, " ------------------------------------------------"
    PRINT #ITER, " DIVIDENTE          MOMENTO DI INERZIA          "
    PRINT #ITER, " ------------------------------------------------"
    PRINT #ITER,
    FOR I = 1 TO T + 1
      PRINT #ITER, TAB(2); I; TAB(20); INERZ(I)
    NEXT I
  END IF
NEXT ITER
END SUB

SUB UscitaRisultati
REM
REM  ***************************************************************
REM *                                                               *
REM *                      U s c i t a R i s u l t a t i            *
REM *                                                               *
REM  ***************************************************************
REM
REM  ***************************************************************
REM *                                                               *
REM *       Questa subroutine si occupa dell'uscita (su video, su stampa  *
REM *                          e su file) dei risultati             *
REM *                                                               *
REM  ***************************************************************
FOR ITER = ITER1 + 1 TO ITER2
  PRINT #ITER, "DIVIDENTE SPOST. VERTICALI SPOST. ORIZZONTALI"
  PRINT #ITER,
  FOR K = 1 TO T + 1
    PRINT #ITER, TAB(2); K; TAB(7); V(K, 1); TAB(35); W(K, 1)
  NEXT K
```

```
  PRINT #ITER,
  PRINT #ITER, "DIVIDENTE MOMENTO FLETTENTE"
  PRINT #ITER,
  FOR K = 1 TO T + 1
    PRINT #ITER, TAB(2); K; TAB(15); M(K)
  NEXT K
  PRINT #ITER,
  PRINT #ITER, "TEMPO DI ESECUZIONE = "; TIME2; " SEC."
  PRINT #ITER, "RUN DEL "; DATE$; " ORE "; TIME$
NEXT ITER
END SUB
```