**Indice**

1. Il quesito 2

2. Richiami di Meccanica Analitica 3

3. Metodo dei moltiplicatori di Lagrange 4

4. Problema dinamico di indice differenziale tre 5

5. Partizionamento delle coordinate 6

6. Applicazione al pendolo triplo 7

7. Codice in Fortran 10

8. Codice della unità chiamante 12

9. Codice del modulo 17

**Pendolo triplo**

**1. Il quesito.** Si consideri un pendolo costituito da tre aste caratterizzate dai seguenti parametri geometrici e inerziali:

1.1)

Si assuma che le cerniere in siano senza alcun attrito e si ammetta altresì che l’unica forza attiva agente sia quella dovuta al campo gravitazionale.



Si intende inoltre che sono calcolati rispetto ai rispettivi assi baricentrali ortogonali al piano del pendolo. Allora si chiede

1.2) di impostare il problema dinamico attraverso il metodo dei moltiplicatori di Lagrange;

1.3) di integrarlo per via numerica.

Si assumano le seguenti condizioni iniziali:

1.4)

**2. Richiami di Meccanica Analitica.** Riporto qui quella parte della meccanica analitica a cui ci si può riferire con il nome di *metodologia dei moltiplicatori di Lagrange*. Lo scopo di questa trattazione è quello di rimaneggiare il sistema fondamentale della dinamica in modo da ottenere un sistema di equazioni differenziali del secondo ordine in cui non siano presenti le reazioni vincolari e che sia di agevole integrazione per via numerica.

Assegnato un meccanismo piano costituito da corpi rigidi si assuma per il generico corpo *j-mo* che

* sono le componenti della risultante della sollecitazione effettiva attiva;
* è il momento risultante della sollecitazione effettiva esterna, rispetto al polo ;
* è il momento risultante della sollecitazione vincolare, rispetto al polo ;
* sono le componenti della risultante della sollecitazione reattiva;
* è la massa;
* è il momento d’inerzia rispetto all’asse baricentrale ortogonale al piano del meccanismo;
* soni le coordinate del baricentro;
* è l’angolo che definisce la rotazione del corpo.

Allora scrivendo il sistema fondamentale della dinamica per ciascun corpo rigido si ha

2.1)

Se ora facciamo -per il corpo *j-mo*- le seguenti posizioni

possiamo riscrivere il sistema 2.1 nel seguente modo

2.2)

Facendo ancora le posizioni

2.3)

2.4)

2.5)

2.6)

il sistema 2.2 si scriverà

2.7)

**3. Metodo dei moltiplicatori di Lagrange.** Vediamo ora come il metodo dei moltiplicatori di Lagrange permette di eliminare le reazioni vincolari dalla 2.7. Il lavoro virtuale relativo alla sollecitazione vincolare per il corpo *j-mo* si scrive:

3.1)

Per il meccanismo complessivo si ha allora

3.2)

Consideriamo adesso le equazioni di vincolo e operiamone uno sviluppo in serie di Taylor del primo ordine di punto iniziale , essendo questa una configurazione compatibile con i vincoli:

3.3)

Si può osservare però che se è una configurazione compatibile con i vincoli, allora deve risultare ; non solo, se il sistema di spostamenti virtuali è compatibile con i vincoli deve essere anche vero che . Dunque la 2.10 si riscrive

3.4)

Al limite per si ha infine

3.5)

Operando il partizionamento delle coordinate indicando con quelle dipendenti e con quelle indipendenti, allora si ha

3.6)

D’altra parte -operando la stessa partizione nel vettore delle reazioni vincolari- la 3.2 può essere scritta come segue

3.7)

Le 3.6, 3.7 possono essere associate nel seguente sistema lineare

3.8)

Si consideri ora che se è non singolare allora la matrice a primo membro della 3.8 ha righe L.D.; in particolare la riga *p+1-ma* può essere scritta come combinazione lineare delle altre, ovvero sussiste la relazione

ovvero

3.9)

Sostituendo la 3.9 nella 3.7 abbiamo poi

3.10)

Sostituendo ora la 3.6 nella 3.10 abbiamo

Poiché questa relazione deve essere valida per ogni sistema di spostamenti virtuali congruenti, segue che

3.11)

Associando le 3.9, 3.11 si ha in fine

3.12)

**4. Problema dinamico di indice differenziale tre.** Sostituendo la 3.12 nella 2.7 abbiamo

4.1)

Pertanto, associando la 4.1 con le equazioni di vincolo, si ottiene il sistema che risolve il problema dinamico diretto:

4.2)

**5. Partizionamento delle coordinate.** Operiamo ora il già citato partizionamento delle coordinate (paragrafo 3) nel sistema 4.2. In particolare riordinando la matrice d’inerzia 2.3 secondo tale partizionamento si ottiene la matrice

5.1)

dove mentre contiene sulla diagonale principale i parametri inerziali relativi alle coordinate dipendenti e contiene quelli relativi alle coordinate indipendenti. Si riordina e partiziona anche il vettore 2.4, ponendo

5.2)

Con ciò la prima delle 4.2 si scrive

5.3)

Dalla prima si ricavano i moltiplicatori di Lagrange

5.4)

che sostituiti nella seconda porgono

5.5)

Adesso -per eliminare nella 5.5 le derivate seconde delle coordinate dipendenti- si ripercorra l’analisi cinematica del secondo ordine con il metodo delle equazioni di vincolo. Per vincoli indipendenti dal tempo si ha



5.6)

Sostituendo la prima delle 5.6 nella 5.5 si ha

5.7)

Abbiamo così ottenuto il sistema di equazioni differenziali del secondo ordine:

5.8)

dove si sono fatte le posizioni

5.9)

Per integrare poi numericamente il sistema 5.8 si seguirà l’algoritmo indicato in figura.

**6. Applicazione al pendolo triplo.** Vediamo ora di calcolare le grandezze 5.9 nel caso del pendolo triplo qui in esame. Assumiamo come coordinate lagrangiane sovrabbondanti le

6.1)

dove ho indicato anche la partizione scelta fra coordinate dipendenti e indipendenti. La relativa matrice di inerzia 5.1 si scrive allora

6.2)

mentre la partizione 6.3 del vettore delle forze si scrive

6.3)

Per le equazioni di vincolo si ricava dalla figura che

6.4)

Per gli jacobiani si ha allora

6.5)

6.6)

Procedo adesso al calcolo di (definito nella seconda delle 5.6). Tuttavia -essendo probabilmente più veloce- calcolo tale vettore nella sua espressione equivalente . Si ha

Quindi si conclude che

6.7)

Sostituendo le 6.2, 6.3, 6.5, 6.6, 6.7 nelle 5.9 e sostituendo a sua volta queste nel 5.8 si ottiene il sistema da integrare con le condizioni iniziali 1.4.

**7. Il codice in Fortran.** Per integrare il sistema 5.8 sono state scritte in Fortran le seguenti unità:

* *main\_ese\_14\_b* è l’unità chiamante la quale si occupa di integrare il sistema 5.8;
* *mod\_ese\_14\_b* è il modulo, il quale contiene i parametri inerziali e geometrici del problema, oltre alle seguenti subroutine:
* *inverse* la quale si occupa di invertire la matrice ;
* *pendolo* la quale disegna la configurazione del pendolo per ogni iterazione;
* *energie* la quale disegna gli andamenti della en. cinetica, potenziale e totale.

Il diagramma di flusso non si discosta -se non in qualche dettaglio- da quello generale indicato nel paragrafo 5. Si precisa inoltre che

* l’integrazione copre un intervallo di 5 secondi;
* si operano *n* iterazioni, ovvero si è applicato un incremento temporale di *5/n* secondi.

Per verificare la bontà dell’incremento temporale *h* utilizzato per integrare, il sistema 5.8 si può ricorrere all’espressione della energia totale del meccanismo, la quale deve conservarsi poiché ci troviamo in assenza di attrito.

L’energia potenziale del campo gravitazionale è data

L’energia cinetica è invece data da

Dunque l’energia totale del sistema è data da

8.1)

dove la costante additiva arbitraria *C* è stata posta pari a zero. Ebbene per un numero di iterazioni

8.2)



le energie cinetica, potenziale gravitazionale e totale hanno gli andamenti indicati in figura. Si vede quanto tenda ad aumentare -al progredire del numero di iterazione- l’energia totale. Sembra dunque opportuno aumentare il numero di iterazioni.



Attraverso l’analisi grafica qui proposta si scopre che l’energia totale comincia a stabilizzarsi per un numero di iterazioni

8.3)

in corrispondenza si ha l’andamento delle energie indicato in figura. Se si plotta un fotogramma ogni 10 iterazioni si ha un fotogramma ogni ; se poi si usa un tempo di riproduzione del filmato di si ha un’animazione rallentata di un fattore .

**8. Codice della unità chiamante.**

PROGRAM main\_ese\_14\_b

USE DISLIN !libreria grafica

USE mod\_ese\_14\_b !il modulo

!sezione dichiarativa

IMPLICIT NONE

!dichiaro gli array dgli angoli e delle loro derivate

REAL(KIND=8),DIMENSION(0:n-1):: theta2, theta2\_p, theta2\_p\_p

REAL(KIND=8),DIMENSION(0:n-1):: theta3, theta3\_p, theta3\_p\_p

REAL(KIND=8),DIMENSION(0:n-1):: theta4, theta4\_p, theta4\_p\_p

!dichiaro gli array delle coordinate dipendenti

REAL(KIND=8),DIMENSION(0:n-1):: x2, y2, x2\_p, y2\_p, x2\_p\_p, y2\_p\_p

REAL(KIND=8),DIMENSION(0:n-1):: x3, y3, x3\_p, y3\_p, x3\_p\_p, y3\_p\_p

REAL(KIND=8),DIMENSION(0:n-1):: x4, y4, x4\_p, y4\_p, x4\_p\_p, y4\_p\_p

!dichiaro gli array delle variabili dipendenti e indipendenti

REAL(KIND=8),DIMENSION(6,0:n-1):: u, u\_p, u\_p\_p

REAL(KIND=8),DIMENSION(3,0:n-1):: v, v\_p, v\_p\_p

!dichiaro i jacobiani

REAL(KIND=8), DIMENSION(6,6):: psi\_u, inv\_psi\_u

REAL(KIND=8), DIMENSION(6,3):: psi\_v

!dichiaro alcune matrici usate nel seguito

REAL(KIND=8), DIMENSION(6,6):: t\_inv\_psi\_u

REAL(KIND=8), DIMENSION(3,6):: t\_psi\_v

REAL(KIND=8), DIMENSION(3,6):: t\_molt

REAL(KIND=8), DIMENSION(6,6):: m\_molt

!dichiaro gli array delle forze

REAL(KIND=8),DIMENSION(6):: Fu

REAL(KIND=8),DIMENSION(3):: Fv

!dichiaro le matrici di inerzia

REAL(KIND=8),DIMENSION(6,6):: Muu

REAL(KIND=8),DIMENSION(3,3):: Mvv

!dichiaro i momenti di inerzia delle aste

REAL:: IG2, IG3, IG4 !i momenti d'inerzia in kg per m^2

!dichiaro gli array del sistema risolvente

REAL(KIND=8),DIMENSION(3,3):: A, inv\_A

REAL(KIND=8),DIMENSION(3):: b, b1, b2

REAL(KIND=8),DIMENSION(3):: x

REAL(KIND=8),DIMENSION(6):: csi

!dichiaro l'array del tempo

REAL(KIND=4), DIMENSION(0:n-1):: tempo

!dichiaro l'array della energia cinetica, potenziale e totale

REAL(KIND=4), DIMENSION(0:n-1):: T\_2, T\_3, T\_4, T, P, E

!dichiaro l'indice del ciclo

INTEGER:: i

!indico una stringa che uso per chiudere il programma

CHARACTER(len=10):: chiusura

!sezione esecutiva

!calcolo h

h = intervallo/REAL(n-1)

WRITE(\*,\*) h

!calcolo i momenti di inerzia

IG2 = (1/12)\*m2\*l2\*\*2

IG3 = (1/12)\*m3\*l3\*\*2

IG4 = (1/12)\*m4\*l4\*\*2

!inizializzo il vettore delle forze

Fu(1) = 0.

Fu(2) = (-1.)\*m2\*g

Fu(3) = 0.

Fu(4) = (-1.)\*m3\*g

Fu(5) = 0.

Fu(6) = (-1.)\*m4\*g

Fv = 0.

!inizializzo le matrici d'inerzia

Muu = 0.

Muu(1,1) = m2

Muu(2,2) = m2

Muu(3,3) = m3

Muu(4,4) = m3

Muu(5,5) = m4

Muu(6,6) = m4

Mvv = 0.

Mvv(1,1) = IG2

Mvv(2,2) = IG3

Mvv(3,3) = IG4

!impongo le condizioni iniziali

theta2(0) = theta2\_zero

theta2\_p(0)= theta2\_p\_zero

theta3(0) = theta3\_zero

theta3\_p(0)= theta3\_p\_zero

theta4(0) = theta4\_zero

theta4\_p(0)= theta4\_p\_zero

x2(0) = (l2/2.)\*COS( theta2(0) )

y2(0) = (l2/2.)\*SIN( theta2(0) )

x3(0) = l2\*COS( theta2(0) ) + (l3/2.)\*COS( theta3(0) )

y3(0) = l2\*SIN( theta2(0) ) + (l3/2.)\*SIN( theta3(0) )

x4(0) = l2\*COS( theta2(0) ) + l3\*COS( theta3(0) ) + (l4/2.)\*COS( theta4(0) )

y4(0) = l2\*SIN( theta2(0) ) + l3\*SIN( theta3(0) ) + (l4/2.)\*SIN( theta4(0) )

x2\_p(0) = 0.

y2\_p(0) = 0.

x3\_p(0) = 0.

y3\_p(0) = 0.

x4\_p(0) = 0.

y4\_p(0) = 0.

tempo(0) = 0.

!inizializzo i vettori delle coordinate

u(1,0) = x2(0)

u(2,0) = y2(0)

u(3,0) = x3(0)

u(4,0) = y3(0)

u(5,0) = x4(0)

u(6,0) = y4(0)

v(1,0) = theta2(0)

v(2,0) = theta3(0)

v(3,0) = theta4(0)

!calcolo lo jacobiano delle u

psi\_u = 0.

psi\_u(1,1) = -1.

psi\_u(2,2) = -1.

psi\_u(3,3) = -1.

psi\_u(4,4) = -1.

psi\_u(5,5) = -1.

psi\_u(6,6) = -1.

inv\_psi\_u = psi\_u

ciclo: DO i= 0, n-1, 1

 !calcolo lo jacobiano di v

 psi\_v = 0.

 psi\_v(1,1) = (-1.)\*(l2/2.)\*SIN(theta2(i))

 psi\_v(2,1) = (l2/2.)\*COS(theta2(i))

 psi\_v(3,1) = (-1.)\*l2\*SIN(theta2(i))

 psi\_v(4,1) = l2\*COS(theta2(i))

 psi\_v(5,1) = (-1.)\*l2\*SIN(theta2(i))

 psi\_v(6,1) = l2\*COS(theta2(i))

 psi\_v(3,2) = (-1.)\*(l3/2.)\*SIN(theta3(i))

 psi\_v(4,2) = (l3/2.)\*COS(theta3(i))

 psi\_v(5,2) = (-1.)\*l3\*SIN(theta3(i))

 psi\_v(6,2) = l3\*COS(theta3(i))

 psi\_v(5,3) = (-1.)\*(l4/2.)\*SIN(theta4(i))

 psi\_v(6,3) = (l4/2.)\*COS(theta4(i))

 !calcolo csi

 csi(1) = (l2/2.)\*(theta2\_p(i)\*\*2)\*COS(theta2(i))

 csi(2) = (l2/2.)\*(theta2\_p(i)\*\*2)\*SIN(theta2(i))

 csi(3) = l2\*(theta2\_p(i)\*\*2)\*COS(theta2(i)) + (l3/2.)\*(theta3\_p(i)\*\*2)\*COS(theta3(i))

 csi(4) = l2\*(theta2\_p(i)\*\*2)\*SIN(theta2(i)) + (l3/2.)\*(theta3\_p(i)\*\*2)\*SIN(theta3(i))

 csi(5) = l2\*(theta2\_p(i)\*\*2)\*COS(theta2(i)) + l3\*(theta3\_p(i)\*\*2)\*COS(theta3(i)) + (l4/2.)\*(theta4\_p(i)\*\*2)\*COS(theta4(i))

 csi(6) = l2\*(theta2\_p(i)\*\*2)\*SIN(theta2(i)) + l3\*(theta3\_p(i)\*\*2)\*SIN(theta3(i)) + (l4/2.)\*(theta4\_p(i)\*\*2)\*SIN(theta4(i))

 !calcolo b

 t\_psi\_v = TRANSPOSE(psi\_v)

 t\_inv\_psi\_u = TRANSPOSE(inv\_psi\_u)

 t\_molt = MATMUL(t\_psi\_v,t\_inv\_psi\_u)

 m\_molt = MATMUL(Muu,inv\_psi\_u)

 b1 = MATMUL(t\_molt,MATMUL(m\_molt,csi))

 b2 = MATMUL(t\_molt,Fu)

 b = b1 - b2 + Fv

 !calcolo A

 A = Mvv + MATMUL( t\_molt,MATMUL(m\_molt,psi\_v) )

 !calcolo l'inversa di A

 CALL inverse(A,inv\_A,3)

 !calcolo x

 x = MATMUL(inv\_A,b)

 !calcolo le derivate seconde delle v e delle u

 v\_p\_p(:,i) = x

 u\_p\_p(:,i) = MATMUL( (-1.)\*inv\_psi\_u,( MATMUL(psi\_v,v\_p\_p(:,i)) - csi ) )

 !incremento il tempo

 tempo(i+1) = tempo(i) + h

 !integro una volta

 u\_p(:,i+1) = u\_p(:,i) + u\_p\_p(:,i)\*h

 v\_p(:,i+1) = v\_p(:,i) + v\_p\_p(:,i)\*h

 !integro un'altra volta

 u(:,i+1) = u(:,i) + u\_p(:,i)\*h

 v(:,i+1) = v(:,i) + v\_p(:,i)\*h

 !WRITE(\*,\*) i, u(:,i), v(:,i)

 !assegno i valori alle coordinate lagrangiane

 x2(i+1) = u(1,i+1)

 y2(i+1) = u(2,i+1)

 x3(i+1) = u(3,i+1)

 y3(i+1) = u(4,i+1)

 x4(i+1) = u(5,i+1)

 y4(i+1) = u(6,i+1)

 x2\_p(i+1) = u\_p(1,i+1)

 y2\_p(i+1) = u\_p(2,i+1)

 x3\_p(i+1) = u\_p(3,i+1)

 y3\_p(i+1) = u\_p(4,i+1)

 x4\_p(i+1) = u\_p(5,i+1)

 y4\_p(i+1) = u\_p(6,i+1)

 theta2(i+1) = v(1,i+1)

 theta3(i+1) = v(2,i+1)

 theta4(i+1) = v(3,i+1)

 theta2\_p(i+1) = v\_p(1,i+1)

 theta3\_p(i+1) = v\_p(2,i+1)

 theta4\_p(i+1) = v\_p(3,i+1)

 !calcolo l'energia cinetica

 T\_2(i) = (0.5)\*( m2\*( (x2\_p(i)\*\*2) + (y2\_p(i)\*\*2) ) + IG2\*theta2\_p(i)\*\*2 )

 T\_3(i) = (0.5)\*( m3\*( (x3\_p(i)\*\*2) + (y3\_p(i)\*\*2) ) + IG3\*theta3\_p(i)\*\*2 )

 T\_4(i) = (0.5)\*( m4\*( (x4\_p(i)\*\*2) + (y4\_p(i)\*\*2) ) + IG4\*theta4\_p(i)\*\*2 )

 T(i) = T\_2(i) + T\_3(i) + T\_4(i)

 !calcolo l'energia potenziale gravitazionale

 P(i) = g\*( m2\*y2(i) + m3\*y3(i) + m4\*y4(i) )

 !calcolo l'energia totale

 E(i) = T(i) + P(i)

 WRITE(\*,\*) E(i)

END DO ciclo

!disegno i fotogrammi del pendolo

fotogrammi: DO i = 0, n-1, 1

CALL pendolo (x2, y2, x3, y3, x4, y4, theta2, theta3, theta4,i)

END DO fotogrammi

!disegno l'andamento delle energie

CALL energie (T, P, E, tempo)

!comando di chiusura

WRITE (\*,\*)"Per chiudere il programma premi una lettera qualunque."

WRITE (\*,\*)"Tutti i dati andranno persi."

READ (\*,\*) chiusura

STOP

END PROGRAM main\_ese\_14\_b

**9. Codice del modulo.**

MODULE mod\_ese\_14\_b

!sezione dichiarativa

IMPLICIT NONE

!dichiaro i parametri inerziali e geometrici del pendolo

REAL(KIND=8), PARAMETER:: m2 = 10.0 !le masse in kg

REAL(KIND=8), PARAMETER:: m3 = 7.0

REAL(KIND=8), PARAMETER:: m4 = 10.0

REAL(KIND=8), PARAMETER:: l2 = 1.0 !le lunghezze

REAL(KIND=8), PARAMETER:: l3 = 0.7

REAL(KIND=8), PARAMETER:: l4 = 1.0

REAL(KIND=8), PARAMETER:: g = 9.807 !accelerazione gravitazionale

REAL(KIND=8), PARAMETER:: pi = 3.1415 !una approssimazione del pi greco

!dichiaro le condizioni iniziali

REAL(KIND=8), PARAMETER:: theta2\_zero = (-5)\*(pi/180)

REAL(KIND=8), PARAMETER:: theta3\_zero = (-15)\*(pi/180)

REAL(KIND=8), PARAMETER:: theta4\_zero = (-25)\*(pi/180)

REAL(KIND=8), PARAMETER:: theta2\_p\_zero = 0.

REAL(KIND=8), PARAMETER:: theta3\_p\_zero = 0.

REAL(KIND=8), PARAMETER:: theta4\_p\_zero = 0.

!dichiaro il numero di iterazioni, l'intervallo e l'incremento temporale

INTEGER, PARAMETER:: n = 8000

REAL(KIND=8), PARAMETER :: intervallo = 5.

REAL(KIND=8) :: h

!scrivo le subroutine

CONTAINS

!---------------------------------------------------------------

 subroutine inverse(a,c,n)

!============================================================

! Inverse matrix

! Method: Based on Doolittle LU factorization for Ax=b

! Alex G. December 2009

!-----------------------------------------------------------

! input ...

! a(n,n) - array of coefficients for matrix A

! n - dimension

! output ...

! c(n,n) - inverse matrix of A

! comments ...

! the original matrix a(n,n) will be destroyed

! during the calculation

!===========================================================

implicit none

integer n

double precision a(n,n), c(n,n)

double precision L(n,n), U(n,n), b(n), d(n), x(n)

double precision coeff

integer i, j, k

! step 0: initialization for matrices L and U and b

! Fortran 90/95 aloows such operations on matrices

L=0.0

U=0.0

b=0.0

! step 1: forward elimination

do k=1, n-1

 do i=k+1,n

 coeff=a(i,k)/a(k,k)

 L(i,k) = coeff

 do j=k+1,n

 a(i,j) = a(i,j)-coeff\*a(k,j)

 end do

 end do

end do

! Step 2: prepare L and U matrices

! L matrix is a matrix of the elimination coefficient

! + the diagonal elements are 1.0

do i=1,n

 L(i,i) = 1.0

end do

! U matrix is the upper triangular part of A

do j=1,n

 do i=1,j

 U(i,j) = a(i,j)

 end do

end do

! Step 3: compute columns of the inverse matrix C

do k=1,n

 b(k)=1.0

 d(1) = b(1)

! Step 3a: Solve Ld=b using the forward substitution

 do i=2,n

 d(i)=b(i)

 do j=1,i-1

 d(i) = d(i) - L(i,j)\*d(j)

 end do

 end do

! Step 3b: Solve Ux=d using the back substitution

 x(n)=d(n)/U(n,n)

 do i = n-1,1,-1

 x(i) = d(i)

 do j=n,i+1,-1

 x(i)=x(i)-U(i,j)\*x(j)

 end do

 x(i) = x(i)/u(i,i)

 end do

! Step 3c: fill the solutions x(n) into column k of C

 do i=1,n

 c(i,k) = x(i)

 end do

 b(k)=0.0

end do

end subroutine inverse

!--------------------------------------------------------------------

SUBROUTINE pendolo (x2, y2, x3, y3, x4, y4, theta2, theta3, theta4, i)

!sezione dichiarativa

!dichiaro gli argomenti fittizi

!dichiaro gli array delle coordinate

REAL(KIND=8),INTENT(IN),DIMENSION(0:n-1):: x2, y2, x3, y3, x4, y4, theta2, theta3, theta4

!dichiaro il numero del fotogramma

INTEGER, INTENT(IN):: i

!dichiaro le variabili locali

!coordinate degli estremi delle aste

REAL(KIND=4):: xB, yB, xC, yC, xD, yD

!stringa usata per la legenda

CHARACTER(len=30)::stringa

!sezione esecutiva

!imposto il tipo di file

CALL METAFL ('bmp') !indico il formato dell'output

CALL BMPMOD (25000,'meter','resolution') !risoluzioneformato .bmp

!imposto la pagina

CALL SCRMOD ('revers') !scritta nera su fondo bianco

CALL DISINI !richaima alcune impostazioni di default

CALL PAGERA !traccio un bordo per il piano xy

CALL DUPLX !font a doppio spessore

!imposto gli assi

CALL AXSPOS (300,1800) !coordinate angolo basso sinistra

CALL AXSLEN (2000,2000) !lunghezza dei due assi in pixel

CALL NAME ('metri','x') !nome delle ascisse

CALL NAME ('metri','y') !nome delle ordinate

CALL GRAF (-2.7,2.7,-2.7,0.4,-2.7,2.7,-2.7,0.4) !imposto gli estremi degli assi e l'intervallo

CALL XAXGIT !traccio la retta x=0

CALL YAxGIT !traccio la retta y=0

CALL NAME ('metri','x') !nome delle ascisse

CALL NAME ('metri','y') !nome delle ordinate

!imposto il titolo

CALL TITLIN ("Pendolo triplo",1) !prima riga del titolo

CALL TITLE !stampa il titolo di cui sopra

CALL DASH !tratteggio per gli assi coordinati

CALL LINWID (5) !spessore delle linee

CALL MYLINE (1,1) !linee continue

!assegno le coordinate dei punti estremi

xB = l2\*COS( theta2(i) )

yB = l2\*SIN( theta2(i) )

xC = xB + l3\*COS( theta3(i) )

yC = yB + l3\*SIN( theta3(i) )

xD = xC + l4\*COS( theta4(i) )

yD = yC + l4\*SIN( theta4(i) )

!disegno il pendolo e la reazione vincolare istante per sitante

!traccio le cerniere

CALL RLCIRC (0.,0.,0.06)

CALL RLCIRC (xB,yB,0.06)

CALL RLCIRC (xC,yC,0.06)

!traccio le barre

CALL RLINE (0., 0., xB, yB)

CALL RLINE (xB, yB, xC, yC)

CALL RLINE (xC, yC, xD, yD)

CALL DISFIN

END SUBROUTINE pendolo

!------------------------------------------------------------------

SUBROUTINE energie (T, P, E, tempo)

!sezione dichiarativa

!dichiaro gli argomenti fittizi

!dichiaro l'array della funzione e quello del tempo

REAL,INTENT(IN),DIMENSION(0:n-1):: T, P, E, tempo

!dichiaro le variabili locali

!dichiaro la stringa usata per la leggenda

CHARACTER(len=13):: stringa

!dichiaro i valori massimi e minimi delle funzioni rx, ry, r

REAL:: max\_T, max\_P, max\_E

REAL:: min\_T, min\_P, min\_E

!dichiaro il valore massimo dei massimi e minimo dei minimi

REAL:: max\_tot

REAL:: min\_tot

!sezione esecutiva

!calcolo il valore massimo e minimo delle funzioni

max\_T = MAXVAL (T(0:n-1))

min\_T = MINVAL (T(0:n-1))

max\_P = MAXVAL (P(0:n-1))

min\_P = MINVAL (P(0:n-1))

max\_E = MAXVAL (E(0:n-1))

min\_E = MINVAL (E(0:n-1))

!calcolo il valore massimo dei massimi e il minimo dei minimi

max\_tot = MAX(max\_T, max\_P, max\_E)

min\_tot = MIN(min\_T, min\_P, min\_E)

!imposto il formato del file

CALL METAFL ('bmp') !indico il formato dell'output

CALL BMPMOD (25000,'meter','resolution') !risoluzione del formato .bmp

!imposto la pagina

CALL SCRMOD ('revers') !scritta nera su fondo bianco

CALL DISINI !richiama alcune impostazioni di default

CALL PAGERA !traccio un bordo per il piano xy

CALL DUPLX !font a doppio spessore

!imposto gli assi x,y

CALL AXSPOS (900,2700) !coordinate angolo basso sinistra

CALL AXSLEN (1700,1300) !lunghezza dei due assi in pixel

CALL NAME ('secondi','x') !nome delle ascisse

CALL NAME ('joule','y') !nome delle ordinate

CALL LABDIG (3,'x') !chiedo 3 cifre decimali per l'asse x

!inizio, fine, intervallo assi x, y

CALL GRAF (0.0,5.,0.0,5./10.,min\_tot,max\_tot,min\_tot,max\_tot/10.)

!traccio le curve

CALL CHNCRV ('line') !usa una linea diversa per ogni curva

CALL CURVE (tempo, T, n) !traccio rx in funzione del tempo

CALL CURVE (tempo, P, n) !traccio ry in funzione del tempo

CALL CURVE (tempo, E, n) !traccio r in funzione del tempo

!imposto il titolo

CALL TITLIN ('Energie',1) !prima riga del titolo

CALL TITLE !stampa il titolo di cui sopra

!imposto la legenda

CALL LEGINI (stringa,3,13) !variabile di carattere, righe, lunghezza

CALL LEGLIN (stringa,'e. cinetica',1) !prima riga della legenda

CALL LEGLIN (stringa,'e. potenziale',2) !seconda riga della legenda

CALL LEGLIN (stringa,'e. totale',3) !terza riga della legenda

CALL LEGTIT ('Legenda') !titolo legenda

CALL LEGEND (stringa,3) !posizine in alto a destra

CALL DISFIN

END SUBROUTINE energie

!--------------------------------------------------------------------

END MODULE mod\_ese\_14\_b