ISTITUTO DI ELABORAZIONE DELL'INFORMAZIONE DEL C.N.R. - PISA

UN SISTEMA PER ALGORITMI DI GENERAZIONE AUTOMATICA DI MATRICI DI RIGIDEZZA NEGLI ELEMENTI FINITI

M. Morandi Cecchi, C. Lami

Nota Interna B74-1

Gennaio 1974

Stampato in proprio

1 - INTRODUZIONE

Gli algoritmi qui presentati per la generazione formale automatica di matrici di rigidezza, necessari quando si voglia analizzare un problema mediante metodi agli elementi finiti, producono dette matrici quando fra i dati di ingresso sia specificato il tipo di elemento da adoperare ed il tipo di funzione mediante la quale si vuole approssimare la soluzione del problema. Alcuni algoritmi sono stati già realizzati ed hanno completamente eliminato le molte ore di calcolo manuale per costruire dette matrici di rigidezza. Sono state in particolare costruite le matrici di rigidezza per il problema degli sforzi piani applicato ad una lastra e per la flessione di una piastra. E' stato realizzato l'algoritmo per la costruzione della matrice necessaria alla soluzione del problema relativo all'equazione del calore nello spazio-tempo sia nel caso di una funzione di approssimazione di tipo lineare sia mediante una approssimazione di tipo quadratico. E' stato preparato un sistema di programmi costruito principalmente da routines scritte (per la maggior parte in linguaggio Fortran IV versione G e in Assembler 360, che è stato messo a punto sul calcolatore IBM 360/67 del C.N.U.C.E. di Pisa. Tale sistema è stato chiamato INTER ed è capace di riconoscere ed elaborare una serie di istruzioni (programma) che consentono di operare su insiemi di dati di natura formale.
L'idea fondamentale del lavoro è quella di produrre un sistema di programmi capaci di operare formalmente sopra le variabili e di produrre le matrici di rigidezza in maniera formale quale output del programma.

Integrazione e tecniche di semplificazione che operando formalmente sono in genere i problemi più difficili sono state in questi problemi risolte, tenendo conto dei dati del particolare problema. Sono cioè risolubili quando il tipo di elemento in esame sia specificato, la funzione approssimante sia specificata e le dimensioni della geometria dell'elemento siano prese come parametri.
In tal caso il sistema INTER fa ricorso a delle routines specifiche che sono indicate nel diagramma dinamico del sistema e le cui singole funzioni sono illustrate nel seguito. Tali routines prendono nome di istruzioni del sistema in quanto ne costituiscono in effetti le macroistruzioni.
In appendice è riportato un esempio di programma scritto per il sistema INTER nel caso della matrice relativo ad una discretizzazione triangolare per l'equazione del calore.

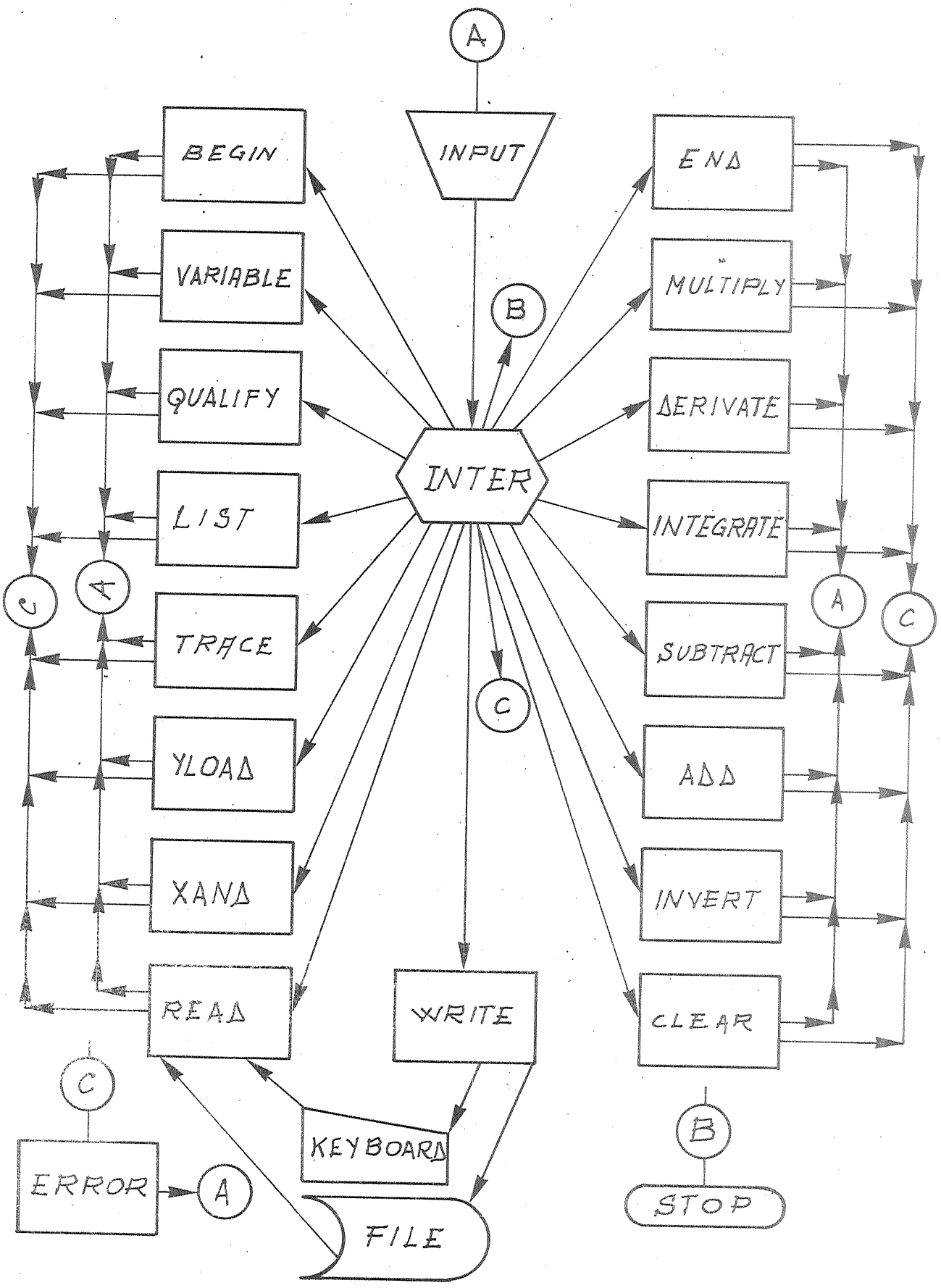
2 - ALCUNE OSSERVAZIONI SUL FUNZIONAMENTO DEL SISTEMA

Una volta appurata l'unità di input dalla quale leggere un programma, il sistema inizia l'esame delle istruzioni. Il primo riconoscimento, comune a tutte le istruzioni, viene effettuato sul nome. Si tratta, in sostanza, di verificare se il nome dell'istruzione è presente nel vocabolario del sistema. Il nome dell'istruzione può contenere nel suo interno una opzione; questa indica il modo in cui s'intende utilizzare l'istruzione. Da questo punto in poi INTER opera sugli argomenti, e tale modo di operare varia da istruzione a istruzione.
Ci limitiamo pertanto a riportare solo l'elenco degli argomenti possibili:

1) Nomi associati ad aree dati;
2) Nomi associati alle variabili;
3) Dimensioni aree dati;
4) Numero delle variabili;
5) Coordinate degli elementi da copiare;
6) Parametri opzionali.

Infine và ricordato che per ogni istruzione vengono valutati, prima e in forma ordinata, gli argomenti fissi, successivamente quelli opzionali che possono non rispondere a nessun criterio di ordinamento.

DIAGRAMMA DINAMICO DI INTER



3 - ISTRUZIONI

Le istruzioni, per la funzione che compiono durante l'elaborazione di un programma, si possono dividere in 6 tipi:

1) Istruzioni di controllo
2) " " definizione
3) " " I/O
4) " " operative
5) " " operative speciali
6) " " per le aree di memoria

3.1 - Istruzioni di controllo

Appartengono a questo tipo le istruzioni che permettono al sistema di individuare l'inizio e la fine di un programma e all'utente di conoscere lo stato attuale delle aree dati, delle variabili e di specificare commenti.

BEGIN:
Una volta riconosciuta questa istruzione, che significa inizio di un programma, si passerà a inizializzare le aree di memoria (puntatori, liste, vettori, indici, ecc.) necessarie per l'esecuzione del programma stesso.

Formato

BEGIN <VARIA = X X>

BEGIN è il nome dell'istruzione da eseguire.

<VARIA = X X> è un parametro opzionale il cui uso è a discrezione dell'utente. Infatti, questo parametro permette di specificare il numero delle variabili che s'intenderà usare durante una parte o per tutto il corso del programma (vedere per questo l'istruzione VARIABILE). Il numero massimo che può essere assegnato a X X è 12. Se l'opzione non è specificata, X X verrà considerato uguale a 7.

N.B. L'istruzione BEGIN deve essere sempre la prima di ogni programma; può, a discrezione dell'utente, essere riusata anche al suo interno. La conseguenza di ciò sarà quella di perdere tutto quanto era stato precedentemente memorizzato.

LIST:

Istruzione che consente di conoscere lo stato attuale delle aree usate per memorizzare dati o risultati e quello relativo alle variabili.

Formato

LIST qualunque cosa

LIST è il nome dell'istruzione da eseguire.
Provoca, a terminale, la stampa di tutte le informazioni relative alle aree ATTIVE, alle aree SLEEP e alle variabili usate.
Per le aree ATTIVE e SLEEP vedere "GESTIONE AREE DATI" e "AREE DATI".

N.B. L'analisi degli argomenti cessa dopo aver riconosciuto il nome dell'istruzione. E' possibile quindi, dopo il nome, scrivere qualsiasi cosa.
Questa istruzione può essere usata come opzione nella istruzione END.

TRACE:

Istruzione che permette la stampa a terminale della sequenza delle istruzioni fino a quel momento inviate al sistema.

Formato

TRACE <qualunque cosa>

TRACE nome dell'istruzione da eseguire.
Nella stampa a terminale della traccia saranno omessi i commenti e le istruzioni errate.
La lunghezza massima consentita della traccia è di 100 istruzioni.
Anche per questa istruzione vale la nota fatta per la LIST.

END:

Istruzione che indica la fine di un programma.

Formato

END <$> TRACE LIST

END è il nome dell'istruzione da eseguire.

<$> è un parametro opzionale il quale consente, se specificato, di lasciare il controllo al sistema per una nuova elaborazione di programmi; se non specificato, l'istruzione END sarà intesa come fine programma e come fine utilizzazione del sistema.

<TRACE>

<LIST>

Sia che il parametro opzionale <$> compaia nell'istruzione, sia che non compaia viene verificato se nell'istruzione sono stati specificati i parametri opzionali TRACE e LIST .
Per l'uso di questi due parametri vedere le relative istruzioni.

*:

Istruzione che permette di specificare commenti.

Formato

* <qualunque cosa>

Questa istruzione può essere inserita in qualsiasi punto del programma.

YLOAD:

Istruzione che permette di ricopiare i termini formali contenuti in un generico elemento, in uno o più elementi della stessa area. Da notare che su di un'area così generata non si potrà effettuare l'inversione.
(Istruzione GAUSS)

Formato

YLOAD CAMPO1 I J I1 J1 I2 J2 ... IN JN

YLOAD è il nome dell'istruzione da eseguire.

CAMPO1 è il nome associato all'area su cui si desidera operare.

I e J sono le coordinate dell'elemento da copiare.

I1 J1
I2 J2
IN JN
sono le coordinate degli elementi in cui dovrà essere ricopiato l'elemento (I, J).

Questa istruzione dà la possibilità di copiare uno stesso elemento in più altri elementi diversi. Il numero di questi ultimi non può superare i 20 per ogni istruzione.
Qualora si presentasse la necessità di dover eseguire questa operazione in più di 20 elementi occorre ripetere l'istruzione con gli elementi mancanti.

Es. YLOAD CAMPO1 IJ L1 K1 L2 K2 ... LN ... KN

WRITE:
Istruzione di output, permette l'uscita dei dati memorizzati in un'area il cui nome associato e l'apparecchiatura di output sono specificati nell'istruzione

Formato

WRITE CAMPO1 < OPRINT > < X >

WRITE è il nome dell'istruzione da eseguire.

CAMPO1 è il nome associato all'area dei dati che si vogliono stampare.

<X> è un parametro opzionale che permette di specificare l'unità di output (unità consentite X = 6 tastiera, X = 8 file). Se X non è specificato viene assunta come unità di output la tastiera.

<OPRINT> è anch'esso opzionale e specifica il metodo che l'utente desidera usare per l'uscita delle informazioni contenente nell'area specificata. L'unità di output usata con questo metodo è la 8 (per maggiori chiarimenti vedere la routine OPRINT CMS).
Questo tipo di parametro è specificato dall'utente che preveda di fare un uso massiccio dell'unità di output.

3.2. Istruzioni operative

Appartengono a questo tipo quelle istruzioni che consentono di eseguire operazioni di addizione, sottrazione e moltiplicazione tra insiemi di dati di natura formale.
A questo tipo appartiene anche l'istruzione X AND che permette l'eliminazione di una o più variabili da un insieme formale.

ADD e SUBSTRACT:
Queste due istruzioni eseguono rispettivamente (ADD) l'addizione e (SUBSTRACT) la sottrazione tra due aree dati i cui nomi associati sono specificati nell'istruzione. Nell'istruzione è anche specificato il nome associato all'area dati atta a contenere il risultato (preventivamente semplificato) dell'operazione.
Queste due istruzioni vengono trattate assieme perchè hanno una identica struttura.

Formato

ADD	CAMPO 1	CAMPO 2	CAMPO 3
SUBTRACT	CAMPO 1	CAMPO 2	CAMPO 3

ADD e SUBTRACT	sono i nomi delle istruzioni da eseguire.
CAMPO 1 e CAMPO 2	sono i nomi associati alle aree del 1° e del secondo operando.
CAMPO 3	è il nome associato all'area nella quale si desidera memorizzare il risultato dovuto all'esecuzione dell'istruzione.

Per la ADD; CAMPO 3 = CAMPO 1 + CAMPO 2
Per la SUBTRACT; CAMPO 3 = CAMPO 1 - CAMPO 2
Le caratteristiche (lunghezza area e dimensioni) che saranno assunte dall'area CAMPO 3 , verranno reperite da uno dei due operandi.
Per assegnare l'area CAMPO 3 saranno effettuati tutti i controlli descritti nella "GESTIONE AREE DATI" e verrà scelto il metodo più opportuno.
Viene inoltre controllato, prima dell'esecuzione dell'istruzione, se le dimensioni del 1° e del 2° operando sono uguali.
Se vi è diversità a terminale verrà stampata la dicitura di "ERRORE DIMENSIONI".

MULTIPLY:
Questa istruzione permette di eseguire diversi modi di moltiplicazione tra due aree dati formali.

Formato

MULTIPL <Y>	CAMPO 1	<T>	CAMPO 2<T>	CAMPO 3
MULTIPL <Y>	CAMPO 1			

MULTIPL è il nome dell'istruzione da eseguire.

<Y> se il nome dell'istruzione è completato con la "Y", oppure è incompleto ma non ha come ultimo carattere l'"1", significa che si vuole moltiplicare l'area associata al CAMPO 1 con l'area associata al CAMPO 2 e memorizzare nell'area associata al CAMPO 3 il risultato, preventivamente semplificato.

<1> se il nome dell'istruzione è completato con "1", oppure è incompleto ma ha come ultimo carattere l'"1", significa che si desidera moltiplicare ogni elemento della 1^{a} riga dell'area associata al CAMPO 1 , per se stesso e per tutti gli altri elementi della riga.
Si chiede inoltre che il risultato di tale moltiplicazione (matrice quadrata) sia posto nel CAMPO 1.

<T> è un parametro opzionale e interessa solo l'istruzione quando è completato con la "Y" o è incompleto ma non ha come ultimo carattere l'"1".
Questa opzione indica, se presente come ultima lettera del nome CAMPO 1 e/o CAMPO 2 , che l'insieme interessato deve essere considerato nella forma trasposta.

La MULTIPL, quando non ha la specifica "1" è in grado di eseguire tutti i tipi di moltiplicazione tra due insiemi.
Sono possibili vari tipi di moltiplicazione; fra matrice e matrice, fra vettore e vettore, e fra vettore e matrice.
Ogni volta che sarà richiesto di eseguire una di queste moltiplicazioni, si verificherà se la relazione tra le dimensioni delle due aree è soddisfatta.

Se non risulta soddisfatta si avrà a terminale la stampa della dicitura "ERRORE DI RELAZIONE".
Se nessuna delle opzioni <Y> o <1> è specificata l'INTER esegue il 1° tipo di istruzione (MULTIPL<Y>).

N.B. Anche un'area dati di tipo vettore può essere considerata nella forma trasposta.
Come per l'ADD e la SUBTRACT, anche per la MULTIPL<Y>, l'utente dà al sistema il reperimento della memoria, il calcolo delle dimensioni e l'assegnazione della nuova area da utilizzare per il risultato dell'istruzione.
Per quest'ultima vedere "AREE DATI" e "GESTIONE AREE DATI".

3.3 - Istruzioni operative speciali

Appartengono a questo tipo 3 istruzioni (INTEGRATE, DERIVATE e GAUSS) che eseguono calcoli del tutto particolari e limitati.

INTEGRATE:
Permette di integrare (secondo un determinato metodo di integrazione) su di un'area rettangolare o triangolare, elemento per elemento di una matrice formale. Al momento attuale il sistema è corredato di routines per il calcolo di due soli tipi di integrali.

1° Tipo - Integrazione su di un elemento rettangolare.

$$\int_0^a \int_0^b \int_0^c x^p y^q z^t \, dx\, dy\, dz = \frac{a^{p+1} b^{q+1} c^{t+1}}{(p+1)(q+1)(t+1)}$$

2° Tipo - Integrazione su di un elemento triangolare se L_1, L_2 ed L_3 sono le coordinate di area del triangolo

$$\int_{\Delta} L_1^a L_2^b L_3^c = \frac{a!b!c!}{(a+b+c+2)!} \; 2\Delta$$

ove con Δ si indica l'area del triangolo.

GAUSSS:

Consente di invertire, con il metodo di GAUSS, un insieme di dati in cui ogni elemento è composto di un termine numerico.
In casi del tutto particolari, quando esiste una relazione lineare fra la matrice delle costanti nodali ed i coefficienti della funzione approssimante, è stato possibile scrivere delle routines che consentono l'inversione formale di tale matrice.

Formato

GAUSS CAMPO 1

GAUSS è il nome dell'istruzione da eseguire.

CAMPO 1 è il nome associato all'area da invertire ed è anche il nome dell'area in cui sarà posto il risultato dell'inversione. E' compito del sistema creare un'area di lavoro per eseguire l'inversione; completata l'inversione l'area di lavoro viene distrutta.

Formato

INTEGRA	<T> <1>	CAMPO 1	CAMPO 2	V1	V2	V3
INTEGRA	<2>	CAMPO 1	CAMPO 2	V1	V2	V3

INTEGRA — è il nome dell'istruzione da eseguire.

<T> o <1> e <2> — sono parametri opzionali che identificano il metodo di integrazione da usare.

CAMPO 1 — è il nome associato alla matrice da integrare elemento per elemento.

CAMPO 2 — è il nome associato alla matrice che dovrà contenere il risultato dell'integrazione.

V1 V2 V3 / V1 V2 V3 V4 — sono i nomi associati alle variabili che vengono usate nei due metodi di integrazione.

Per calcolare un integrale con un diverso metodo da quelli consentiti, occorrerà scrivere un nuovo programma e inserirlo sotto il controllo del sistema.
Per il CAMPO 2 , cioè per il reperimento dell'area risultati vedere "GESTIONE AREE DATI".
Se nessuna delle opzioni <T>; <1>; <2> è specificata viene calcolato il primo tipo di integrale.

DERIVATE:
Questa istruzione abilita il sistema al calcolo della derivata parziale (I^a e II^a), elemento per elemento di una matrice, rispetto a una variabile.

Formato

DERIVAT <E> CAMPO 1 CAMPO 2 < V1 > < V2 >
<1>

DERIVAT <2> CAMPO 1 CAMPO 2 < V1 >

DERIVAT è il nome dell'istruzione da eseguire.

<E>; <1> sono parametri opzionali.
< 2 >

<E><o><1> La presenza di questo tipo di opzione permette di calcolare la derivata I^a rispetto alla variabile V1. Se è presente anche l'opzione V2 , il precedente risultato viene deviato anche rispetto alla variabile V2.

<2> Questa opzione consente di calcolare la derivata II^a rispetto alla variabile V1 specificata nell'istruzione.

CAMPO 1 è il nome associato alla matrice da derivare.

CAMPO 2 è il nome associato alla matrice risultato.

<V2><V1> sono i nomi associati alle variabili rispetto a le quali si deve derivare.

Se nessuna delle opzioni <E>; <1>; <2> è specificata, viene calcolata la derivata I^a rispetto alla variabile indicata.
Per il reperimento dell'area di memoria da associare a CAMPO 2 vedere "AREE DATI" e "GESTIONE AREE DATI".

3.4 - Istruzioni per le aree di memoria

Quando l'utente desidera riutilizzare una o più aree di memoria, le cui informazioni sono già state utilizzate, può, tramite l'istruzione CLEAR, rilasciare dette aree, lasciando al sistema il compito di gestirle. A tale scopo, le aree così rilasciate, saranno collocate nella struttura AREE SLEEP (per una possibile riutilizzazione) e ne sarà cancellato il nome che le era associato, dalla struttura AREE ATTIVE.

CLEAR:
Permette il rilascio di una o più AREE ATTIVE già utilizzate per la pronta riutilizzazione delle stesse con scopi diversi.

Formato

CLEAR CAMPO 1 CAMPO 2 ..... CAMPO N

CLEAR è il nome dell'istruzione da eseguire.

CAMPO 1 CAMPO 2 ..... CAMPO N
sono i nomi associati alle aree che si desidera rilasciare.

Non vi sono limitazioni per il numero dei nomi delle aree da rilasciare specificati nell'istruzione; và ricordato però che la lunghezza del record di ogni istruzione è di 80 caratteri.

3.5 Istruzioni di I/O

Queste istruzioni danno la possibilità di immettere dati in memoria o di riceverne.

READ:

Istruzione di input, permette di leggere dati e memorizzarli in una area di memoria. Il nome associato all'area, le dimensioni di tale area e l'apparecchiatura di input sono specificate nell'istruzione.

Formato

READ CAMPO 1 ≤ N≪M> <FORMAT <1>> < X >

READ — è il nome dell'istruzione da eseguire.

CAMPO 1 — è il nome che dovrà essere associato all'area nella quale saranno memorizzati i dati di input.

N ed M — sono le dimensioni dell'area di input (per specificare vettori sarà necessario assegnare a N o a M il valore 1).

<FORMAT<1>> e < X > — sono parametri opzionali; FORMAT, se specificato, segnala che l'utente desidera leggere i dati di input con un proprio formato (in questo caso verrà stampata a terminale la dicitura "FORMATO UTENTE", dopo di che l'utente invierà il proprio formato), se FORMAT non è specificato, la lettura sarà effettuata secondo le specifiche del formato standard dell'INTER.
(1x, 2I3, 1x, F12.5, 1x, 12I3).

<1> — Questa subopzione, presente solo se il parametro opzionale FORMAT è utilizzato, specifica che l'utente intende leggere i propri dati con il formato che precedentemente ha inviato.

<X> è il numero dell'unità di input da cui leggere i dati. Se <X> non è specificato verrà assegnata come apparecchiatura di input la tastiera.
Per l'assegnazione dell'area di nome CAMPO 1 vedere "AREE DATI" e "GESTIONE AREE DATI".

N.B. Sono accettate come apparecchiature di input le unità che vanno da 1 ÷ 5 e da 7 ÷ 9. Specificando l'unità 5 od omettendo l'unità di lettura si ottiene lo stesso risultato. Infatti la tastiera viene identificata con il numero 5.
Se il programma dell'utente non viene letto da tastiera è necessario scrivere immediatamente dopo l'istruzione READ (questo, nel caso in cui si volessero leggere i dati di input con un proprio formato), la specifica del formato dell'utente.

Es. (1x, 2I3, 1x, F4.0, 3I2)

Una specifica di formato dell'utente viene perduta sia nel caso che l'utente invii un nuovo formato, che utilizzi il formato standard del sistema.

3.5 Istruzioni di definizione

Queste istruzioni operano sulle variabili consentendone la definizione del numero e la qualificazione dei nomi.

VARIABILE:
Istruzione che dà la possibilità di inizializzare o modificare il numero delle variabili. Il suo uso può avvenire in qualsiasi parte del programma. Opera, nella maggior parte dei casi, in stretta connessione con l'istruzione QUALIFY.

Formato

VARIABLE <XX>

VARIABLE è il nome dell'istruzione da eseguire.

<XX> è il numero delle variabili con le quali si vuole operare ($0 \leqslant XX \leqslant 12$).

QUALIFY:

Questa istruzione dà la possibilità all'utente di qualificare il nome (max. 2 caratteri) delle variabili che saranno utilizzate durante il corso di un programma. Se all'utente si presentasse la necessità di modificare i nomi delle variabili durante una elaborazione, può utilizzare nuovamente la QUALIFY con i nuovi nomi da assegnare.
Se il numero dei nomi delle variabili è minore del numero delle variabili specificato tramite una BEGIN VAR = XX o una VARIABLE XX, il sistema assegnerà i nomi standard alle variabili in eccedenza.
Da tener presente che riveste particolare importanza l'ordine con cui si inviano i nomi delle variabili.

Formato

QUALIFY V1 V2 V3 ..... VN

QUALIFY è il nome dell'istruzione da eseguire.

V1 V2 V3 ... VN sono i nomi da associare alle variabili

N.B. Dato che si accettano un massimo di 12 variabili, 12 sono anche i nomi consentiti.
Se l'istruzione QUALIFY non viene utilizzata vengono assegnati, alle variabili usate, nomi standard (A1 B1 C1 ..... M1).

4 - DICITURE DI ERRORE

Un errore può essere denunciato in due momenti diversi:

a) durante l'esame degli argomenti di una istruzione;

b) durante l'esecuzione dell'istruzione.

Nel caso a) l'analisi cessa appena si è riconosciuto un errore e viene stampata a terminale la dicitura di errore appropriata. L'errore che avviene durante un esame (errore sintattico) non crea problemi per il proseguimento del programma.
Basta semplicemente riscrivere l'istruzione nella forma corretta.
L'esecuzione non si completa solo nel caso che si abbia a terminale la stampa della dicitura "MEMORY OVERFLOW". Questo avviene quando il sistema cerca di generare una nuova area e non ha a disposizione la memoria occorrente. L'utente per poter continuare può provare a liberare, se ne ha a disposizione, aree dati che non servono e quindi tentare di nuovo facendo rieseguire l'istruzione che ha dato errore.
Nel caso b) l'elaborazione cessa. La responsabilità è da attribuirsi sempre all'utente. Solo nel caso di "OVERFLOW ESPONENTE", è necessario tener conto che il sistema non permette di operare con esponenti il cui valore cada al di fuori dell'intervallo (- 15 + 15).

5 - AREE DATI

Le aree dati di un utente vengono gestite dal sistema INTER tramite tre strutture di memoria.

1) Struttura aree attive
2) Struttura aree sleep
3) Data list

5.1 - Struttura aree attive

La generazione di un'area attiva avviene quando il nome associato a un'area dati identifica un'area di input o un'area atta a contenere il risultato di una istruzione. La struttura aree attive ha il compito di contenere tutte le informazioni relative a un'area attiva o utilizzabile dall'utente.

5.2 - Struttura aree sleep

La generazione di un'area "sleep" avviene tramite l'uso appropriato di una istruzione (CLEAR). Questa area, contenente informazioni delle quali l'utente non intende fare più uso, viene posta in stato "sleep" o di attesa per essere utilizzata in futuro. La sua utilizzazione avverrà nel momento in cui sarà presentata al "sistema" la richiesta di un'area attiva compatibile con le caratteristiche (lunghezza area) di quella in stato "sleep".
La struttura aree sleep ha il compito di contenere le informazioni relative a un'area attiva utilizzata e resa libera dall'utente.

5.3 - Data list

La struttura "data list" ha il compito di contenere le informazioni relative a un generico termine formale (coefficienti ed esponenti delle variabili) di un generico elemento di un
Questa struttura, per una migliore utilizzazione della memoria, è gestita nella forma di FREE-LIST, con possibilità quindi di GARBAGE COLLECTIONS.

6 - GESTIONE AREE DATI

Per una migliore utilizzazione della memoria disponibile è necessaria la collaborazione tra l'utente e il sistema. Infatti se il nome che l'utente intende associare a un'area non è presente nella struttura aree attive, è compito del sistema verificare se esistono aree attive rese libere dall'utente (aree SLEEP) che hanno le caratteristiche (lunghezza area) compatibili con l'area richiesta.
Se esistono aree SLEEP saranno utilizzate (qualora non se ne trovino di uguali dimensioni) quelle con dimensioni maggiori che più si avvicinano per lunghezza; inoltre a quest'ultima sarà assegnato il nome specificato dall'utente e detto nome sarà inserito nella struttura aree attive.
Se non esistono aree "sleep" o aree "sleep" compatibili, si utilizzerà una parte (di uguali dimensioni) di aree di memoria associandovi il nome specificato.
Detto nome, associato all'area appena generata, sarà inserito nella struttura aree attive.
L'ultimo caso (quello in cui si sviluppa la collaborazione tra l'utente e il sistema) è quello di specificare, da parte dell'utente, e utilizzare un'area presente nella struttura aree attive.
Di fronte a questa richiesta il sistema lascierà libera l'area in questione per la pronta riutilizzazione della stessa con scopi diversi.
L'utente dovrà stare attento che la nuova dimensione, che sarà assegnata all'area, sia di lunghezza non superiore alla vecchia.

7 - STRUTTURA DATI DI INPUT

I dati di input che siano letti tramite le specifiche di formato dell'utente o secondo quelle standard del sistema dovranno essere così strutturati:

I J COEFF V1 V2 V3 ..... VN

dove I e J sono le coordinate di un generico elemento.

COEFF e V1 V2 V3 ..... VN
sono rispettivamente il coefficiente e i valori degli esponenti delle variabili di un termine dell'elemento letto.

N.B. Se un generico elemento è composto di più termini, si può omettere, dopo il primo, la specifica del valore delle coordinate I e J nei campi relativi. Per le variabili non presenti nel termine letto, il campo della relativa V dovrà essere posto uguale a 0 oppure uguale a blank. Da ricordare inoltre che particolare significato investe l'ordine con cui si inviano i valori degli esponenti delle variabili. Detto ordine sarà mantenuto per tutto il corso del programma.
Infine per indicare che i dati relativi a una certa area sono terminati, basta inviare un record di input particolare dove tutte le informazioni, meno la coordinata I sono poste uguale a blank. La I, ossia il campo relativo alla coordinata I, dovrà contenere un valore 0.

7.1 - Esempi di input

1) Lettura di un termine dell'elemnto (1,1) di un'area dati.

Struttura formale $0.5\ a\ b^{2}\ c^{-1}$

Struttura record di input	coordinate		coeffic.	a	b	c
	1	1	0.5	1	2	-1

2) Lettura dell'elemento (4,4) e del record di fine data

Struttura formale $0.5\ a^{-1}\ b\ c^{2} + 0.3\ c^{-1} - 0.015\ b^{-1}\ c^{-1}$

Struttura record di input	coordinate		coeffic.	a	b	c	
	4	4	0.5	-1	1	2	
	'b'	'b'	0.3	0	0	-1	
	'b'	'b'	-0.015	0	-1	-1	b = blank

Record fine dati -1

8 - SEDUTA A TERMINALE

Qui di seguito viene riportato, come esempio, un programma molto semplice che permette di moltiplicare due insiemi formali (ALFA e BETA). Il programma, che dovrà contenere le istruzioni relative alla lettura dei dati delle due aree, al calcolo della moltiplicazione e alla stampa dei risultati, potrà essere inviato da una unità di input diversa dalla tastiera (tastiera = 5) (unità 1 4 e 7 9) (nell'esempio riportato il programma viene inviato da tastiera). I dati delle due aree sono memorizzati sull'unità 3 e saranno letti secondo le specifiche di formato inviato dall'utente. L'uscita dei risultati sarà ottenuta sull'unità 8 (metodo OPRINT).

Infine và ricordato che, ogni qualvolta il sistema è in grado di eseguire un'istruzione, invia all'utente il segno "___" (solo nel caso di programma inviato da tastiera).

Nell'esempio riportato si fà uso di 3 variabili.

(in ambiente CMS) Le scritte in maiuscolo si riferiscono all'utente; quelle minuscole al sistema.

```
INTERPRE
EXECUTION BEGINS ..... (CMS)
source program file ?  5

              . Inter . 1973

-   * INIZIALIZZAZIONE
-   BEGIN
-   * DEFINIZIONE NUMERO VARIABILI
-   VARIABLE 3
-   * ASSEGNAZIONE NOMI ALLE VARIABILI
-   QUALIFY X Y T
-   * LETTURA AREA ALFA
-   READ ALFA 2  3 FORMAT 3

    formato utente
    (2I3, 1x, F4.0, 3I2)

-   * STAMPA AREA ALFA
-   WRITE ALFA OPRINT
-   * LETTURA AREA BETA
-   READ BETA 2  3 FORMAT1 3
-   * STAMPA AREA BETA
-   WRITE BETA OPRINT
-   * MOLTIPLICAZIONE DI ALFA-TRASP. * BETA
-   * GAMMA SARA' L'AREA RISULTATO
-   MULTIPLY ALFAT BETA GAMMA
-   * STAMPA AREA GAMMA
-   WRITE GAMMA OPRINT
-   * FINE PROGRAMMA
-   END

              . Inter . 1973

R; ../.../...
```

9 - LIMITI E RESTRIZIONI

9.1 - Programma

L'INTER accetta programmi non più lunghi di 100 istruzioni, la prima delle quali dovrà essere sempre una BEGIN e l'ultima una END.

9.2 - Istruzioni

La lunghezza del record di ogni istruzione è di 80 caratteri. Gli argomenti dell'istruzione, rispettando l'ordine stabilito, non sono vincolati a campi fissi del record; occorre però che siano sempre separati da uno o più caratteri di "blank".
Un'istruzione non può continuare nel record successivo; l'assenza di un argomento fisso provoca a terminale la stampa della dicitura "FINE BUFFER".

9.3 - Nomi di istruzioni

Ogni istruzione è riconosciuta tramite il primo carattere del nome. Al primo carattere ne possono seguire una serie che ne completano il nome e che in totale (compreso il primo) non devono superare gli 8.
Per le istruzioni di DERIVATE, INTEGRATE e MULTIPLY (END) la lunghezza del nome dell'istruzione è sempre di 8 caratteri; l'ultimo però, e non necessariamente l'ottavo (per la END il quarto), ha un compito particolare come visto in dettaglio nella descrizione delle istruzioni.

9.4 - Nomi di variabili

I nomi associati alle variabili (la cui presenza è indicata dal valore del proprio esponente) possono avere una lunghezza massima di 2 caratteri e non devono essere in numero superiore a 12.

9.5 - Nomi di campi

Anche i nomi dei campi (nomi associati alle aree dati) possono essere lunghi 8 caratteri. Nel caso dell'istruzione MULTIPLY l'ultimo carattere dei primi due campi di cui necessita, ha una particolare funzione, infatti stabilisce il modo in cui dovrà essere considerata l'area. E' opportuno che l'utente eviti di associare nomi od aree da moltiplicare la cui ultima lettera sia una "T".

9.6 - Lunghezza di aree

Attualmente sono a disposizione dell'utente, globalmente, 4.000 elementi, i quali, mediamente, possono essere formati di 2,5 termini formali ciascuno.

9.7 - Parametri formali

Per quanto riguarda l'opzione VARIA = XX, specificata nella BEGIN, questa può raggiungere la lunghezza massima di 6 caratteri compreso il segno di =. Al segno di "=" deve immediatamente seguire il numero delle variabili che l'utente intende definire per il proprio programma.

Per tutti gli altri parametri opzionali, come del resto per il VARIA = , il riconoscimento avviene sulla prima lettera. Nel caso del parametro opzionale FORMAT 1 si considera anche l'ultimo carattere. Infine tutti i parametri opzionali possono essere assunti nell'istruzione perchè ciò è previsto in un modo di operare alternativo.

9.8 - Valori degli esponenti delle variabili

I valori degli esponenti possono essere compresi nell'intervallo (-15⟷+15). Qualora il sistema dovesse compiere una somma algebrica tra due valori degli esponenti non compresi nell'intervallo (-7⟷+7), informerebbe l'utenete, con una stampa a terminale, che è avvenuto un overflow.

9.9 - Numero delle aree

Il numero massimo delle aree attive e delle aree "sleep" è di 10 per ciascuna.

9.10 - Programma e dati di input

Vi sono due modi per inviare un programma:

a) da file

b) da tastiera

a) da file

L'utente dovrà memorizzare il proprio programma su di un file di una certa unità e alla richiesta del sistema, formulata con la dicitura SOURCE PROGRAM FILE?, rispondere con il valore dell'unità utilizzata.

b) da tastiera

Si tratta di rispondere alla domanda del sistema con 5. Questa risposta specifica che l'input del programma avverrà da tastiera.

9.11 - Dati di input

Come il programma anche i dati possono essere inviati da tastiera o da un'altra unità. Per indicarlo basta assegnare, nella istruzione READ, all'argomento che qualifica l'apparecchiatura di input, il numero dell'unità desiderata.
Se l'input dei dati avviene tramite tastiera l'argomento qualificante l'apparecchiatura di input può essere omesso.

APPENDICE

```
*
*          **********************************
*          *                                *
*          *     COSTRUZIONE  MATRICI       *
*          *                  DI            *
*          *              RIGIDEZZA         *
*          *                                *
*          **********************************
*
*    PROVA CASO TRIANGOLARE CALORE (6 X 6)
*
*
*          INIZIALIZZAZIONE SISTEMA. DEFINIZIONE E
*     QUALIFICAZIONE DELLE VARIABILI.
*
*          LETTURA DEL POLINOMIO INIZIALE 'POLI' (1 X 6)
*
BEGIN
VARIABLE 9
QUALIFY L1 L2 L3 X1 X2 X3 Y1 Y2 TT
READ POLI 6 6 FORMAT 2
WRITE POLI OPRINT
*
*          SI CALCOLA LA MATRICE POLI
*    MOLTIPLICANDO OGNI TERMINE DEL POLINOMIO
*    PER SE STESSO E PER GLI ALTRI.
*
MULTIPL1 POLI
WRITE POLI OPRINT
*
*          SI CALCOLA WK.
*    WK E' DATO DALL'INTEGRAZIONE DI POLI.
*
INTEGR2 POLI WK L1 L2 L3 TT
WRITE WK OPRINT
*
*          DOPO AVER CALCOLATO L'INTEGRALE:
*
*             A    B    C     (A! * B! * C!) * 2
*    //     L1   L2   L3     ------------------|
*     |                       (A + B + C + 2)!
*
*     SI QUALIFICANO ALCUNE VARIABILI.
*
QUALIFY X0 X1 X2 X3 Y0 Y1 Y2 Y3
*
*          LETTURA DI GX.
*
READ GX 6 6 FORMAT1 2
```

```
_ *
_ *          SI RICOPIANO GLI ELEMENTI UGUALI.
_ *
_ YLOAD GX 1 1       2 1    3 1     4 4     5 4     6 4
_ YLOAD GX 1 2       2 2    3 2     4 5     5 5     6 5
_ YLOAD GX 1 3       2 3    3 3     4 6     5 6     6 6
_ WRITE GX OPRINT
_ *
_ *          SI CALCOLA K1.
_ *     K1 = GXTRASPOSTA * WK * G
_ *
_ *          LA G DEVE ESSERE INTESA COME
_ *     LA MATRICE GX CON L'ELIMINAZIONE DELLA
_ *     VARIABILE X0 DA TUTTI I TERMINI DELL'INSIEME.
_ *
_ *     SI RIUTILLIZZA L'AREA POLI CHE ADESSO NON SERVE.
_ *
_ CLEAR POLI
_ MULTIPLY GXT WK WORK
_ XAND GX X0
_ MULTIPLY WORK GX K1
_ WRITE K1 OPRINT
_ *
_ *     SI RIUTILIZZANO LE AREE GX E WORK .
_ *
_ CLEAR GX WORK
_ *
_ *          SI CALCOLA K2.
_ *     K2 = GYTRASPOSTA * WK * G
_ *
_ *        LETTURA DI GY
_ *
_ *     ANCHE QUI LA G DEVE ESSERE INTESA COME
_ *     L'INSIEME GY SENZA LA VARIABILE Y0.
_ *
_ READ GY 6 6 FORMAT1 2
_ *
_ *          SI RICOPIANO GLI ELEMENTI UGUALI.
_ *
_ YLOAD GY 1 1    2 1    3 1    4 4    5 4    6 4
_ YLOAD GY 1 2    2 2    3 2    4 5    5 5    6 5
_ YLOAD GY 1 3    2 3    3 3    4 6    5 6    6 6
_ WRITE GY OPRINT
_ MULTIPLY GYT WK WORK
_ XAND GY Y0
_ MULTIPLY WORK GY K2
_ WRITE K2 OPRINT
_ *
_ *     SI RIUTILIZZANO LE AREE WORK E GY
_ *
_ CLEAR WORK GY
_ *
_ *
_ *        LETTURA DI DELTA (AREA).
_ *
_ READ DELTA 1 1  2
_ WRITE DELTA OPRINT
_ *
```

```
*       MOLTIPLICAZIONE DI K1 AND K2 PER DELTA (AREA).
*
MULTIPL2 K1 DELTA KK1
WRITE KK1 OPRINT
CLEAR K1
MULTIPL2 K2 DELTA KK2
WRITE KK2 OPRINT
CLEAR K2
*          CALCOLO K3.
*    K3 = KK1 - KK2
*
SUBTRACT KK1 KK2 K3
WRITE K3 OPRINT
*
*    LETTURA DI GI.
*
READ GI 6 6 FORMAT 2
WRITE GI OPRINT
*
*    CALCOLO WKGI.
*    WKGI = WK * GI
*
MULTIPLY WK GI WKGI
WRITE WKGI OPRINT
*
*
*       MOLTIPLICAZIONE DI WKGI PER DELTA.
*
MULTIPL2 WKGI DELTA WKKGI
WRITE WKKGI OPRINT
CLEAR WKGI
*
*    CALCOLO K4.
*    K4 = K3 - WKKGI
*
SUBTRACT K3 WKKGI K4
WRITE K4 OPRINT
*
*     SI RIUTILIZZANO LE AREE K3,WK,KK1,KK2,GI,WKKGI.
*
CLEAR K3 WK KK1 KK2 GI WKKGI
*
*    LETTURA DELLA N.
*
READ N 6 6 FORMAT1 2
WRITE N OPRINT
*
*    SI CALCOLA R1.
*    R1 = NTRASPOSTA * K4 * N
*
MULTIPLY NT K4 WORK
CLEAR K4
MULTIPLY WORK N R1
WRITE R1 OPRINT
END TRACE LIST
```

```
T R A C E

 1) BEGIN
 2) VARIABLE 9
 3) QUALIFY L1 L2 L3 X1 X2 X3 Y1 Y2 TT
 4) READ POLI 6 6 FORMAT 2
 5) WRITE POLI OPRINT
 6) MULTIPL1 POLI
 7) WRITE POLI OPRINT
 8) INTEGR2 POLI WK L1 L2 L3 TT
 9) WRITE WK OPRINT
10) QUALIFY X0 X1 X2 X3 Y0 Y1 Y2 Y3
11) READ GX 6 6 FORMAT1 2
12) YLOAD GX 1 1       2 1    3 1     4 4     5 4     6 4
13) YLOAD GX 1 2       2 2    3 2     4 5     5 5     6 5
14) YLOAD GX 1 3       2 3    3 3     4 6     5 6     6 6
15) WRITE GX OPRINT
16) CLEAR POLI
17) MULTIPLY GXT WK WORK
18) XAND GX X0
19) MULTIPLY WORK GX K1
20) WRITE K1 OPRINT
21) CLEAR GX WORK
22) READ GY 6 6 FORMAT1 2
23) YLOAD GY 1 1    2 1    3 1    4 4    5 4    6 4
24) YLOAD GY 1 2    2 2    3 2    4 5    5 5    6 5
25) YLOAD GY 1 3    2 3    3 3    4 6    5 6    6 6
26) WRITE GY OPRINT
27) MULTIPLY GYT WK WORK
28) XAND GY Y0
29) MULTIPLY WORK GY K2
30) WRITE K2 OPRINT
31) CLEAR WORK GY
32) READ DELTA 1 1  2
33) WRITE DELTA OPRINT
34) MULTIPL2 K1 DELTA KK1
35) WRITE KK1 OPRINT
36) CLEAR K1
37) MULTIPL2 K2 DELTA KK2
38) WRITE KK2 OPRINT
39) CLEAR K2
40) SUBTRACT KK1 KK2 K3
41) WRITE K3 OPRINT
42) READ GI 6 6 FORMAT 2
43) WRITE GI OPRINT
44) MULTIPLY WK GI WKGI
45) WRITE WKGI OPRINT
46) MULTIPL2 WKGI DELTA WKKGI
47) WRITE WKKGI OPRINT
48) CLEAR WKGI
49) SUBTRACT K3 WKKGI K4
50) WRITE K4 OPRINT
51) CLEAR K3 WK KK1 KK2 GI WKKGI
52) READ N 6 6 FORMAT1 2
53) WRITE N OPRINT
54) MULTIPLY NT K4 WORK
55) CLEAR K4
56) MULTIPLY WORK N R1
57) WRITE R1 OPRINT
58) END TRACE LIST
```

```
L I S T


A R E E  A T T I V E

      NOME       INIZIO    DIMENSIONE
  1) DELTA            1     1     1
  2) WORK            37     6     6
  3) N              145     6     6
  4) R1             217     6     6
NUMERO AREE ATTIVE =   4


A R E E  S L E E P

      INDICEAA  INIZIOAA  LUNGHEZZA
  1)         3        73         36
  2)         4       109         36
  3)         6       181         36
  4)         8       253         36
NUMERO AREE SLEEP =   4
NUMERO VARIABILI =  9
NOMI VARIABILI X0 X1 X2 X3 Y0 Y1 Y2 Y3 TT
INIZIO AREE LIBERE LIB =  2693 LA =  289


                                    .F O R M I N T E. 1 9 7 3


R; T=86.02/111.48 12.17.49
```