LABORATORIO DI PROGRAMMAZIONE
Corso di laurea in matematica

9 – LE VARIABILI STRUTTUTRATE

Marco Lapegna
Dipartimento di Matematica e Applicazioni
Universita' degli Studi di Napoli Federico II

wpage.unina.it/lapegna

Dalla precedente lezione

Le strutture di controllo determinano l'ordine
con cui vengono eseguite le istruzioni

Struttura di selezione "if-then-else"

Esegue in alternativa due blocchi di istruzioni
a secondo della veridicita' di una condizione logica

Struttura di iterazione "for-endfor"

Esegue piu' volte un blocco di istruzioni
per tutti i valori di un indice compresi in un intervallo

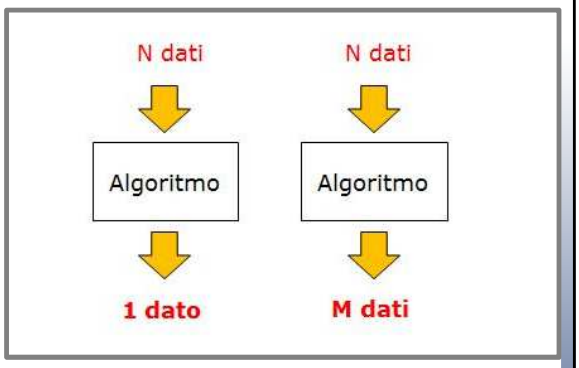
Le strutture di controllo possono essere innestate

Un problema

Algoritmi esaminati finora:

- Somma di N numeri
- Massimo di N numeri

Caratteristica principale:
N dati di input **1 solo dato di output**



Algoritmo N a 1 e Algoritmo N a M

Numerosi problemi
(ad es. quelli di algebra lineare)
hanno una struttura del tipo

N dati di input **M dati** di output

E' necessario **conservare tutti i dati**

Esempio

Dati due vettori geometrici *A* e *B* di lunghezza N

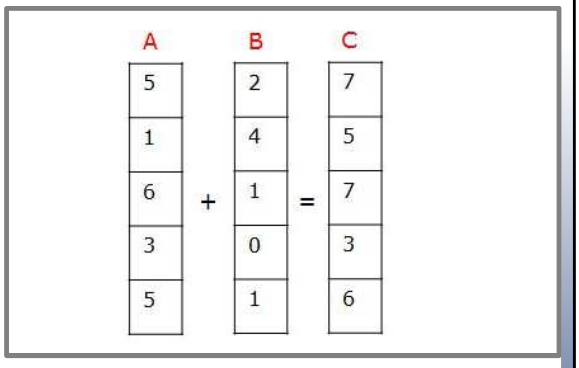
$A = (a_1, a_2, \ldots, a_N)$
$B = (b_1, b_2, \ldots, b_N)$

Calcolare il vettore *C*, somma dei precedenti due

$C = (c_1, c_2, \ldots, c_N)$

Dove $c_i = a_i + b_i$ pe ogni i=1,..,N

E' necessario conservare tutti i valori di *C*



Esempio di somma di due vettori di linghezza N=5

Le variabili strutturate (o strutture dati)

Individuano con **un singolo nome** non un singolo dato,
ma **un insieme di dati organizzati secondo un fissato criterio**

Ogni dato viene trattato come una singola variabile

VARIABILI STRUTTURATE

Le variabili strutturate definiscono il modo di
organizzare ed accedere in maniera efficiente gruppi di dati

Caratteristica comune: **omogeneita' degli elementi** che compongono la struttura

Varie strutture dati: alberi, liste, pile, code, heap, … **array**

Array

Gli array sono strutture dati che nascono con l'obiettivo di rappresentare in maniera efficiente i principali elementi del calcolo scientifico:

vettori e matrici

- **dimensione fissata** (non modificabile)
- tutti gli elementi sono dello **stesso tipo**
- Si accede agli elementi mediante un **indice che indica la posizione** del corrispondente elemento

Un esempio di array di 6 elementi di tipo carattere

Osservazione: a secondo dei linguaggi l'indice puo' partire da
- 1 (ad es. Fortran)
- 0 (ad es. C, C++, Java)

Dichiarazione di un array

Analogamente alle variabili, prima di utilizzare una struttura dati e' necessario dichiararla

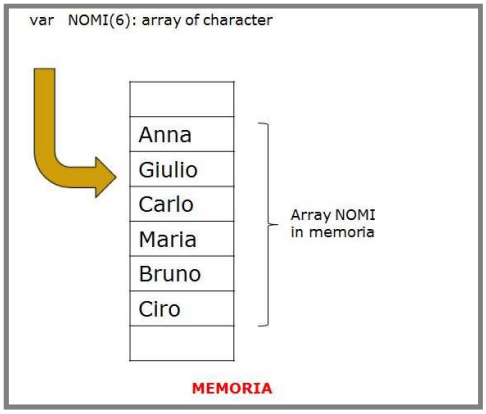
- Dichiarare il tipo della struttura
- Dichiarare il tipo di elementi

Es.: **var NOMI(6): array of character**

Con la dichiarazione viene determinata la dimensione massima della struttura

La dichiarazione permette al compilatore di riservare un adeguato spazio in memoria per tutti gli elementi dell'array

Gli elementi di un array sono allocati in memoria in locazioni consecutive



Dichiarazione di un array di 6 elementi di tipo carattere

I/O di un array

Ogni elemento di un array e' accessibile mediante un indice

NOMI(i) rappresenta l'i-mo elemento

Per le operazioni di I/O e' necessario accedere agli elementi di un array, uno alla volta, mediante una struttura di iterazione for-endfor

Osservazioni:

- l'indice i e' una variabile a tutti gli effetti e va dichiarata come variabile intera
- E' possibile utilizzare solo una parte dell'array

```
var  NOMI(6):  array of character
var   i , N : integer
...
...
N=5
for  i = 1 to N
          read   NOMI(i)
endfor
...
```

Leggere un array

```
var  NOMI(6):  array of character
var   i , N : integer
...
...
N=5
for  i = 1 to N
          print   NOMI(i)
endfor
...
```

Stampare un array

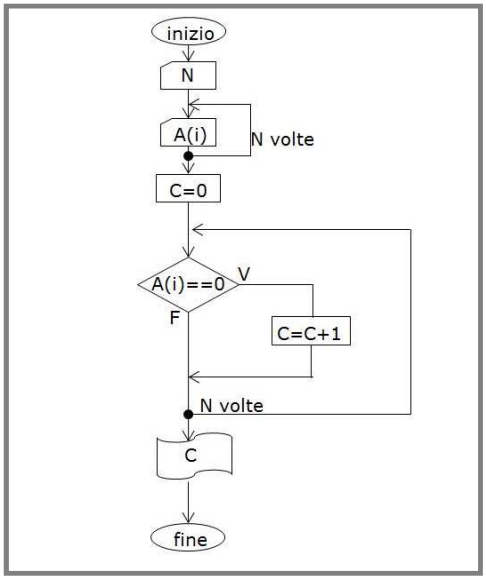
Esempio

Dato un vettore A di lunghezza N calcolare il numero di ricorrenze dell'elemento "0"

Dati di input: (esempio)
N = 10 (la lunghezza dell'array A)
A = (5; 0; 1; 8; 2; 0; 3; -1; 0; 6)

Risultato: Il numero di elementi nulli (C = 3)

Idea:
Esaminare uno alla volta gli N elementi dell'array e incrementare una variabile C quando si trova un elemento nullo



Flow chart per il calcolo delle ricorrenze

Algoritmo in pascal-like

Dati di input: N = 10
A = (5; 0; 1; 8; 2; 0; 3; -1; 0; 6)

passo I=1: A(I) == 0 (F)→ C = 0

passo I=2: A(I) == 0 (V) → C = 1

passo I=3: A(I) == 0 (F) → C = 1

passo I=4: A(I) == 0 (F) → C = 1

passo I=5: A(I) == 0 (F) → C = 1

passo I=6: A(I) == 0 (V) → C = 2

passo I=7: A(I) == 0 (F) → C = 2

passo I=8: A(I) == 0 (F) → C = 2

passo I=9: A(I) == 0 (V) → C = 3

passo I=10: A(I) == 0 (F) → C = 3

Risultato: 3 (ultimo valore di C)

```
begin ricorrenze
var  A(10):  array of   real
var  i , N   : integer
read   N
for i =1 to N
   read  A(i)
endfor
for i = 1 to N
   if  ( A(i)==0 ) then
       C = C+1
   endif
endfor
print  C
end ricorrenze
```

Algoritmo per il calcolo delle occorrenze di elementi nulli in un array in pascal like

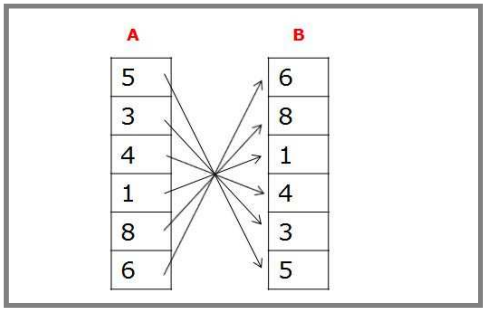
Esempio

- leggere un array A di N elementi
- costruire un array B contenente gli elementi di A in ordine inverso
- stampare gli elementi di B

Esempio: N=6

- I=1 ➔ B(1) = A(6)
- I=2 ➔ B(2) = A(5)
- I=3 ➔ B(3) = A(4)
- I=4 ➔ B(4) = A(3)
- I=5 ➔ B(5) = A(2)
- I=6 ➔ B(6) = A(1)

L'i-mo elemento di B (cioe' B(i)) e' uguale all'elemento di A con indice N-i+1 (cioe' A(N-i+1))



scambio degli elementi di due array

```
begin scambio
var  A(10), B(10) : array of   real
var   i , N , j : integer
read   N
for i =1 to N
   read  A(i)
endfor
j = 0
for i = N to 1 step -1
    j = j +1
   B(i) = A(j)
endfor
for i =1 to N
   print  B(i)
endfor
end scambio
```

algoritmo in pascal like

Osservazioni

- il valore da assegnare alle componenti di un array deve essere del tipo indicato al momento della dichiarazione

```
var   A(5) : array of character
...
for  i  =1 to 5
    A(i) = 2*i
endfor
...
```

Errore !!

Esempio di errore

- non e' possibile fare riferimento ad una componente con un indice maggiore della dimensione dell'array

```
var   A(5) : array of integer
...
for  i  =1 to 10
    A(i) = 2*i
endfor
...
```

Errore !!

Esempio di errore

Array 2-dimensionali

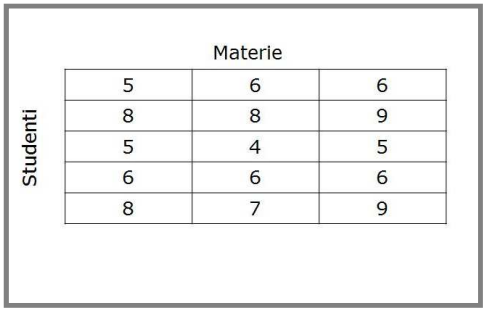
Mediante gli array e' possibile trattare anche tabelle

(array 2-dimensionali)

Esempio

Memorizzare i voti in varie materie di un gruppo di studenti

Necessaria una tabella dove
- Le righe rappresentano gli studenti
- Le colonne rappresentano i voti



Una tabella di voti di un gruppo di studenti

Dichiarazione di array 2-dimensionali

Al momento della dichiarazione e' necessario definire il numero di righe e il numero di colonne che compongono l'array

Esempio:

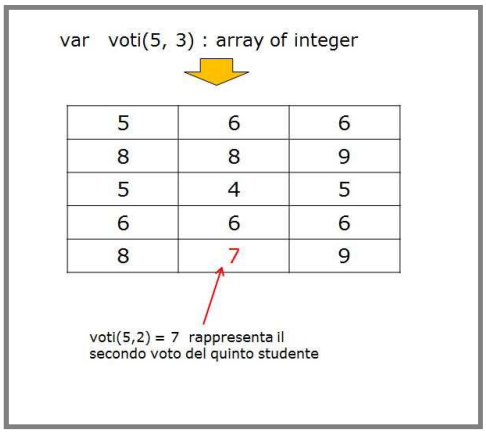
var voti(5, 3) : array of integer

dichiara una array di interi
di 5 righe e 3 colonne

Sono necessari due indici:

- Il primo indice identifica la riga
- Il secondo indice identifica la colonna

voti(i, j) si riferisce al
j-mo voto dell'i-mo studente



Dichiarazione di un array 2-dimensionale

I/O per gli array 2-dimensionali

Per un array 2-dimensionale di

N righe e M colonne

e' necessario accedere a tutti gli elementi, uno alla volta, scorrendo entrambi gli indici

A(i,j) i=1,..,N e j=1,..,M

2 strutture di iterazione for-endfor innestate

Negli esempi riportati
l'indice j varia piu' velocemente
rispetto all'indice i

```
var   A(10,10):  array of integer
var    i , j, N, M: integer
...
N=5
M=8
for  i = 1 to N
      for j=1 to M
           read   A(i,j)
      endfor
endfor
...
```

Lettura di un array 2-dimensionale

```
var   A(10,10): array of integer
var    i , j, N, M: integer
...
N=5
M=8
for  i = 1 to N
      for j=1 to M
           print   A(i,j)
      endfor
endfor
...
```

Stampa di un array 2-dimensionale

Esempio: somma di due matrici

Date due matrici quadrate A e B di ordine N
calcolare la matrice C ottenuta come somma di A e B

Esempio: N=3

$$\begin{pmatrix} 1 & 2 & 3 \\ 1 & 0 & -1 \\ 3 & 2 & 1 \end{pmatrix} + \begin{pmatrix} 5 & 6 & 7 \\ 0 & 1 & 0 \\ 7 & 6 & 5 \end{pmatrix} = \begin{pmatrix} 6 & 8 & 10 \\ 1 & 1 & -1 \\ 10 & 8 & 6 \end{pmatrix}$$

Esempio: somma di due matrici

$$C = A + B$$

Per ogni elemento e' necessario calcolare

$$C(i,j) = A(i,j)+B(i,j)$$
$$i=1,..,N \quad j=1,..,N$$

```
begin sommamatrici
var   A(10,10), B(10,10) : array of integer
var   C(10,10):  array of integer
var    i , j, N: integer
read  N
for  i = 1 to N
       for j=1 to N
            read   A(i,j), B(i,j)
       endfor
endfor
for  i = 1 to N
       for j=1 to N
            C(i,j)=A(i,j)+B(i,j)
       endfor
endfor
for  i = 1 to N
       for j=1 to N
            print  C(i,j)
       endfor
endfor
end sommamatrici
```

Algoritmo per la somma di matrici

Esempio: scambio righe

Data una matrice A di N righe e M colonne
e due interi P e Q
scambiare la P-ma con la Q-ma riga

Esempio: N=5, M=5, P=2 , Q=5

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 0 & -1 & 0 & 1 \\ 5 & 4 & 3 & 2 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 6 & 7 & 8 & 9 & 10 \end{bmatrix} \qquad A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 5 & 4 & 3 & 2 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 1 \end{bmatrix}$$

N.B. Lo scambio deve avvenire sulla stessa matrice

Algoritmo per lo scambio di righe

Scambio delle righe P e Q

Scambio di
A(P,j) con A(Q,j)
j=1,..M

```
begin scambiorighe
var   A(10,10): array of integer
var    i , j, N, M, S: integer
read  N,M
read P, Q
for  i = 1 to N
      for j=1 to M
            read   A(i,j)
      endfor
endfor
for  j=1 to M
            S=A(P,j)
            A(P,j) = A(Q,j)
            A(Q,j) = S
endfor
for  i = 1 to N
      for j=1 to M
            print  A(i,j)
      endfor
endfor
end scambiorighe
```

Algoritmo in pascal like per lo scambio di due righe di una matrice

Riassumendo

Gli array sono variabili strutturate caratterizzate da

- **dimensione fissata** (non modificabile)
- elementi dello **stesso tipo**
- accesso agli elementi mediante **un indice che indica la posizione** del corrispondente elemento

- Versioni **1-dimensionale** (un solo indice) e **2-dimensionale** (2 indici)

- Rappresentano in maniera naturale i principali oggetti matematici dell'algebra lineare (**vettori e matrici**)

Sono elementi universali degli algoritmi
presenti in tutti i linguaggi di programmazione

Un po' di storia (9)

Ken Thompson(1943) e Dennis Ritchie (1941-2011)

- Nel 1969 ai Bell Lab. sviluppano il sistema operativo Unix, una versione piu' efficiente di Multics, capace di funzionare anche su piccoli calcolatori.
- Succesivamente sviluppano il linguaggio di programmazione C, mediante il quale riscrivono Unix, rendendolo cosi' portabile su tutti i calcolatori dove il linguaggio e' disponibile
- Vincono numerosi premi tra cui il Turing Award nel 1983
- A causa delle leggi antitrust, negli anni '70 i Bell Lab. sono costretti a distribuire gratuitamente Unix, facendolo diventare il s.o. piu' diffuso in ambito scientifico
- Ad oltre 40 anni di distanza Unix e C sono ancora il modello di riferimento per i sistemi operativi (Linux, Android) e per i linguaggi di programmazione (C++, Java)

Thompson e Ritchie nel 1997
(courtesy of Computer History Museum)

Thompson e Ritchie ai Bell Lab. Nel 1970
(courtesy of Computer History Museum)