Gli array

Introduzione

L’array è un tipo di dato semplice che consente di memorizzare una collezione di dati. Cosa è una collezione?

È un insieme di dati simili (che quindi possono essere trattati allo stesso modo).

Vediamo un esempio, consideriamo dei casi reali come:

- le paghe degli impiegati di un’azienda
- l’elenco dei titoli dei libri di una biblioteca
- i voti degli alunni di una classe

Che caratteristiche hanno questi insiemi?

- esiste un tipo di dato da gestire: una paga, un titolo, un voto
- esiste un numero arbitrario di valori di questo tipo
- il programma deve poter memorizzare tutti i valori e poter accedere indifferentemente a ciascuno di esse.

ATTENZIONE: gli elementi del linguaggio visti sinora consentono di memorizzare un solo valore (non una collezione).

Esempio

Supponiamo di risolvere il seguente problema:

oggi vengono interrogati 5 alunni, a ciascuno di essi viene assegnato un voto (numero intero tra 1 e 10): interessa conoscere la media dei voti e il numero di alunni che contengono una valutazione superiore alla media.

Soluzione:

- leggo i numeri uno alla volta
- calcolo la somma
- alla fine dell'inserimento calcolo la media

Pensiamo alla soluzione:

Sino a qui tutto semplice, ma come risolvo il seguito?

Quando leggo il singolo numero non posso sapere se sia maggiore o minore della media, in quanto non posso calcolarla sino a quando non ho finito l'inserimento.

Una volta che calcolo il valore della media i primi valori inseriti li ho persi (sono stati sovrascritti durante i passaggi del ciclo iterativo). Cioè se devo leggere i valori 3, 5, 7, 4, 8 … Userò una variabile VOTO dentro un ciclo, al primo passaggio prenderà il primo voto (VOTO=3), poi al secondo passaggio prenderà il secondo voto (VOTO=5, e il vecchio valore '3' viene perso), al terzo passaggio VOTO=7 (e il '5' viene perso), e così via… Questo vuol dire che alla fine non avrò più ne 3, ne 5, ne 7 e ne 4 ma solo l'ultimo cioè 8.

Come posso fare?

Potrei chiedere nuovamente i valori all'utente e a questo punto confrontarli con il valore medio calcolato. Mah ….

Soluzione improbabile:

Altra soluzione:

usare 5 variabili (una per ciascun voto) ….

Anche questo non ha molto senso, infatti questo programma deve valere per classi con 5, 10, 30 o quanti alunni voglio.

Non ha senso, anche sapendo a priori il numero degli alunni, se ho 30 persone dovrei avere 30 variabili solo per i voti, e questo non esiste!

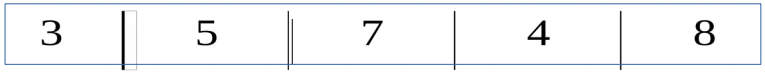
Inoltre devo adattare ad ogni classe il numero delle variabili (potrebbero esserci classi con 20 alunni, classi con 30, classi con 22…)

Soluzione con vettori

Decido allora di usare gli ARRAY o VETTORI.
Se una variabile VOTO la rappresento così:

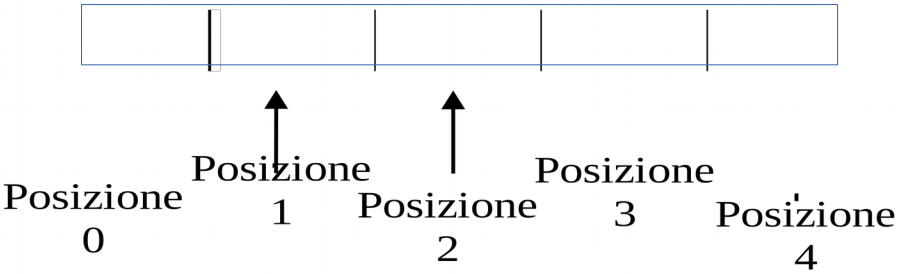
e al suo interno ci scrivo il valore attuale, ad esempio 6

Un array di voti lo rappresento così:



e dentro ci posso scrivere più di un valore.

Ma andiamo con ordine. Supponiamo di avere a disposizione un array con 5 posti.
ATTENZIONE: la cosa più difficile è capire e distinguere la posizione dell'array



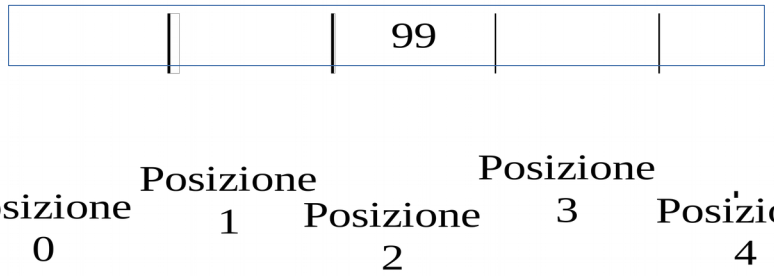
- Le posizioni vengono individuate da dei numeri interi
- A ciascun numero corrisponde un elemento, che potrà avere un valore
- In questo momento posso dire che alla posizione 2 non ho nessun valore, così come alla posizione 4 e così via

Il vettore è vuoto.

esempi

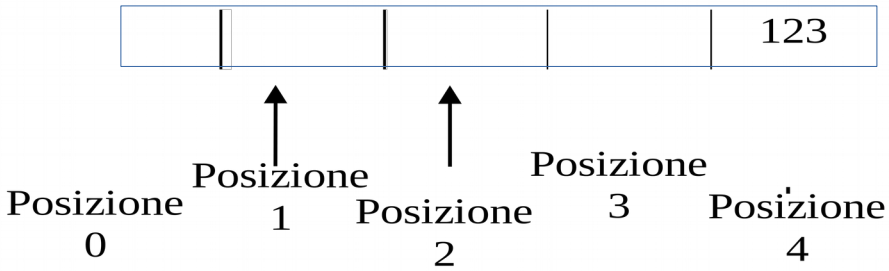
Vediamo degli esempi numerici, ipotizzando che il mio array conterrà al suo interno dei numeri interi.

ESEMPIO 1



Posso dire che il contenuto del vettore in posizione 2 è pari a 99

ESEMPIO 2

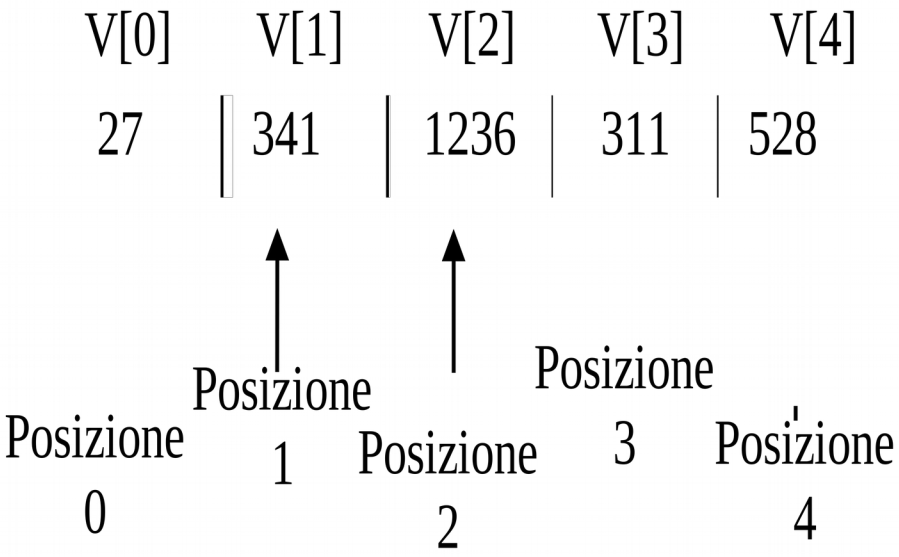


Posso dire che il contenuto del vettore in posizione 4 è pari a 123

Esempio 2

È abbastanza semplice, l'unica cosa da non confondere è il valore della cella con il valore della posizione.
Vediamo di arrivare alla rappresentazione usata nel codice

Supponiamo che il nostro vettore si chiami V, indicheremo



Esempi 3

Per riferirsi ad un elemento bisogna specificare sia il nome del vettore sia la sua posizione, detta indice, secondo la seguente sintassi:

V [3] → a cui corrisponde il valore 311

Nome del vettore

Indice dell' elemento

ATTENZIONE: L'operatore [ ] si chiama accesso all'elemento.

Vediamo degli esempi:

voglio assegnare 10 al primo elemento dell'array V

```
V[0] =10;
```

Voglio assegnare alla variabile ***a*** il contenuto del vettore V di posizione 4

```
a=V[4]; //a varrà 528
```

NOTA: l'indice dell'elemento deve essere di tipo INT, scrivere una istruzione di questo tipo

```
Double x=0;
V[x]=10;
```

è sbagliata, anche se la variabile x ha al suo interno un valore intero (senza la virgola).

NOTA 2: altra cosa da non fare è :

V=1000; il nome del vettore da solo non fa riferimento a nessun elemento, quindi non posso assegnargli nessun valore, è sbagliato.

caricamento

Un’operazione semplice e indispensabile è il caricamento di elementi nel vettore.

Il procedimento è elementare, devo avere accesso a tutti gli elementi dell’array singolarmente, uno per volta, e per ciascuno di questi devo assegnare un valore.

Come si accede ad un elemento??

Abbiamo visto, attraverso un indice, pertanto si avrà

- V[0] per il primo elemento, V[1] per il secondo elemento, V[2] per il terzo elemento, V[3] per il quarto elemento, V[4] per il quinto elemento.

Se i valori da inserire sono nell’ordine 5, 10, 22, 44, 17 per assegnarli al vettore dovrò eseguire 5 istruzioni di questo tipo:

- V[0] =5; V[1] =10 V[2] =22; V[3] =44; V[4] =17;

In java

...da informatici ci accorgiamo che operazioni uguali possono essere indicate come una singola operazione che si ripete (ciclo iterativo):

quanti passi deve fare il ciclo? 5 (una per ciascun elemento)

quale istruzione devo fare ad ogni passaggio? Leggere un valore e assegnarlo ad un elemento dell'array. Come faccio ad accedere in ogni passaggio ad un elemento distinto dell'array? Usando la variabile contatore del ciclo come indice dell'array (è un intero che si incrementa di 1 ad ogni ciclo)

come si indica? Se la variabile si chiama i si avrà V[i]='*valore letto';*

se la variabile fosse stata CONTA avremmo avuto V[CONTA]='*valore letto'.*

Scriviamo le righe di codice corrispondenti a questi semplici passaggi:

```
for (int i=0;i<5;i++){
System.out.print("inserisci un elemento: "); //chiedo un elemento
   v[i]=in.nextInt(); //leggo un elemento
}
```

In java

Vediamo a questo punto qualcosa che riguarda la dichiarazione di un vettore in java

int[] V = new int[5];

Spieghiamo gli elementi di questa riga:

int --> definisce il tipo di elementi che il mio vettore conterrà all'interno

[ ] --> la presenza di questo operatore indica che stiamo dichiarando un array

V--> è il nome che vogliamo dare alla nuova variabile ARRAY

New --> serve per creare in memoria uno spazio riservato alla nostra variabile

Int --> stiamo dicendo che in memoria gli elementi sono di tipo int

[5] --> dichiara quanti elementi avrà il nostro vettore

Questa è la spiegazione della riga che dobbiamo digitare nel linguaggio java per dichiarare un vettore di 5 interi.

In java

ATTENZIONE potrei anche inizializzare il vettore, se conoscessi già a priori i valori che conterrà.

Così come per un contatore abbiamo spesso scritto una istruzione del tipo

```
int conta=0;
```

dove abbiamo dichiarato conta come intero e gli abbiamo assegnato inizialmente il valore 0. Allo stesso modo lo possiamo fare per un vettore, quando in fase di dichiarazione scriviamo:

```
int[] vett = { 23, 44, 55, 34, 2 };
```

Riepilogo

- Dichiarazione vettore
  - tipo [] nome_vettore; (Es. **int [] Vett**)
- Istanziazione
  - nome_vettore=new tipo [N] (dove N rappresenta la dimensione del vettore, es. **Vett=new int[10]**)
- La dichiarazione e l'istanziazione può essere fatta anche in un'unica riga nel seguente modo:
  - tipo[] nome_vettore=new tipo [N] (Es. **int [] Vett=new int [10]**)
- Nel caso in cui non avessi la dimensione del vettore, ma questa dipende da una scelta espressa dall'utente mediante input allora devo necessariamente dichiarare e istanziare separatamente…

Riepilogo 2

- caricamento vettore: per caricare un vettore devo usare un ciclo (consiglio ciclo for)

  for (int i=0;i<v.length;i++) //N rappresenta il numero di celle del vettore

  v[i]=in.nextint(); //leggo elemento

- stampa di un vettore: anche per stampare un vettore devo usare un ciclo (consiglio ciclo for)

  for (int i=0;i<N;i++) //N rappresenta il numero di celle del vettore

  System.out.println(Vett[i]); //stampa l'elemento i-esimo del vettore

Esercizio 4

Esercizio 4a

Il ciclo for-each negli array

Il ciclo **for-each** “non richiede” gli indici degli elementi

```
double []a=new double[10];
for (double elemento:a)
        System.out.print(elemento+” “); //stampa elemento array
```

la sintassi generale è la seguente:

for(tipo_base variabile: nome_array)
*   istruzioni*

dove tipo_base rappresenta il tipo della variabile dichiarata dentro il for dello stesso tipo degli elementi dell’array.

E’ preferibile quando possibile usare questo tipo di ciclo negli array!

Ordinamento array

Supponiamo di voler riordinare un array in senso crescente o decrescente... senza scendere nei dettagli degli algoritmi di ordinamento, possiamo dire che la Java Class Library fornisce gli algoritmi di ordinamento. La class Arrays del package ***java.util*** fornisce il metodo **sort.**

Arrays.sort(vett); //ordina tutto l'array vett

Arrays.sort(vett, inizio, fine); //ordina il vettore vett dall'indice iniziale (inizio) all'indice finale (fine)

Array multidimensionali

Matrici dichiarazione

La sintassi delle matrici segue quella degli array.

```
// dichiaro alcune matrici
tipoCella [][]nomeMatrice; //a due dimensioni
tipoCella []..[]nomeMatrice; // a + dimens.
```

esempi

```
//dichiaro alcune matrici
int [][]mat; //dichiaro mat a 2 dimensioni
double [][]qMat; //qMat matrice di reali a 2 dimens
String frasi3d[][][]; // matrice 3 dimens
boolean logic3d[][][]; // matrice 3 dimens
char myCaratteri[][]; // matrice 2 dimens
```

Matrici istanziazione

La creazione di matrici (allocazione di memoria) è analoga a quella degli array

```
NomeMatrice = new TipoMatrice[<Elenco-Dimensioni>]; //creo matrice
```

Esempi

```
//creo le matrici già dichiarate
m = new int[13][17]; // 13 righe e 17 colonne; ogni cella è un intero
q = new double [11][21]; // 11 righe e 21 colonne; ogni cella è un reale
wow = new float [3][5][7]; // 3 righe, 5 colonne, 7 livelli di profondità
```

Oppure si può creare e istanziare in un'unica riga:

```
Tipo []nomeMatrice = new tipo[dimensione];
```

Esempi

```
int [][]m = new int[13][17]; // 13 righe e 17 colonne; ogni cella è un intero
double [][]q = new double [11][21]; // 11 x 21; ogni cella è un reale
char wow[][][] = new char [3][5][7];//3 righe, 5 colonne, 7 di profondità
```

Matrici inizializzazione

La inizializzazione di matrici nella dichiarazione è analoga a quella degli array mono-dimensionali. Occorre particolare attenzione ai blocchi.

La inizializzazione estesa è indipendente dalla dichiarazione. La sua sintassi è:

```
nomeMatrice = new tipoMatrice [].. {blocco1, blocco2 ... bloccoK};
```

dove tra le graffe (blocco principale) si elencano altri blocchi fino ai valori che si intende inserire nelle celle.

Esempi:

```
//inizializzo le matrici che devono essere già dichiarate
m = new int [][] {{7,5,6} , {4,8,3}};
//nel blocco principale ho due blocchi: uno per la 1a riga e l'altro per la 2a
Q = new double [][] {{7.1,5.2,6.3} , {8.1,7.2,6.4} , {1.1,2.2,3.3}} ;//nel blocco principale ho tre blocchi: uno per ogni riga
```

riepilogo matrici

- Esempio di stampa matrice

```
int [][]m2 = new int[8][8]; // matrice 8righe 8colonne
for (int r = 0; r < 8; r++) {
    for (int c = 0; c < 8; c++)
        System.out.print(m2[r][c] + " \t");
    System.out.println();
    }
```

Esercizio5

- Esempio di caricamento matrice

```
for (int r = 0; r < 8; r++)  //cicli che scandiscono righe e colonne
    for (int c = 0; c < 8; c++)
        m2[r][c]=in.nextint(); //leggo elemento
```

esercizi

1. dichiarare tre matrici quadrate di ordine 4 di interi
    1. inizializzare la prima con dei cicli **for** ponendo in ogni cella un valore dispari
    2. inizializzare la seconda con dei cicli **for** ponendo in ogni cella un valore multiplo di 3
    3. scrivere nella terza la somma delle prime due
    4. mostrare a video la matrice somma (andando a capo correttamente)
2. dichiarare una matrice rettangolare con 2 righe e 6 colonne
    1. inserire dei valori da tastiera (maggiori di zero)
    2. trovare il massimo e stamparlo a video
    3. Trovare la media per riga e stamparla a video
    4. mostrare a video la matrice (andando correttamente a capo)
    5. trovare il massimo e sostituirlo con uno zero
    6. mostrare a video la matrice (andando correttamente a capo)