Array

Maurizio Palesi
Salvatore Serrano

Tipi di dato stutturati

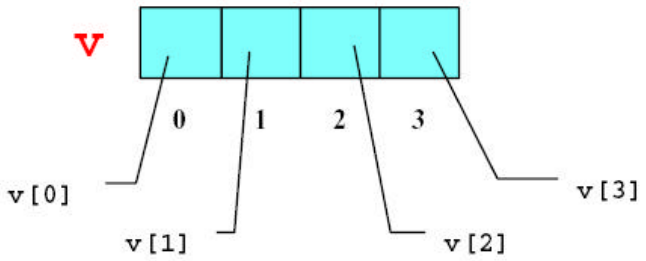
- In C si possono definire **tipi strutturati**
- Vi sono due **costruttori fondamentali**

- `[ ]` **array**
  - ➔collezione finita di N variabili dello stesso tipo

- `struct` **strutture**
  - ➔Collezione finita di variabili non necessariamente dello stesso tipo

Array (vettori)

- Un **array** di **N** elementi è una collezione finita di variabili dello stesso tipo.
- Ogni variabile è identificata da un **indice** compreso tra **0** e **N-1**



Definizione di un array

- **Sintassi:**

```
<tipo> <nomeArray>[<costante>];
```

- **Esempi:**

```
int v[4];
char nome[20];
```

- **ATTENZIONE: Sbagliato!!!**

```
int N;
char nome[N];
```

IL COMPILATORE NON SA COME DIMENSIONARE L'ARRAY

Esempio

- Leggere da tastiera gli elementi di un vettore di 10 elementi

```
#include <stdio.h>

main()
{
   int i, vett[10];

   for (i=0; i<10; 1++)
    {
       printf("Dammi elemento %d: ", i);
       scanf("%d", &vett[i]);
    }
}
```


Esempio

- Leggere da tastiera gli elementi di un vettore di 15 elementi

```
#include <stdio.h>

main()
{
   int i, vett[10];

   for (i=0; i<10; 1++)
    {
       printf("Dammi elemento %d: ", i);
       scanf("%d", &vett[i]);
    }
}
```

vett[15];

i<15

2 modiche!

Esempio

- Leggere da tastiera gli elementi di un vettore di 10 elementi

```
#include <stdio.h>
#define N 10
main()
{
   int i, vett[N];

   for (i=0; i<N; 1++)
    {
       printf("Dammi elemento %d: ", i);
       scanf("%d", &vett[i]);
    }
}
```


Esempio

- Leggere da tastiera gli elementi di un vettore di 15 elementi

```
#include <stdio.h>
#define N 15
main()
{
   int i, vett[N];

   for (i=0; i<N; 1++)
    {
       printf("Dammi elemento %d: ", i);
       scanf("%d", &vett[i]);
    }
}
```

1 sola modifica ben localizzata!

Esempio

- Inizializzare un vettore con il quadrato degli indici
  - ➔Es. {0, 1, 4, 9, 16, 25, 36, ...}

```
#include <stdio.h>
#define N 10
main()
{
   int i=0, vett[N];

   while (i<N) {
      vett[i] = i * i;
      i++;
   }
}
```


Esempio

- Cercare il massimo di un vettore

```
#include <stdio.h>
#define N 5
main()
{
   int vett[N] = {7, 25, 12, 41, 9};
   int i, max;

   max = vett[0];
   for (i=1; i<N; i++)
      if (vett[i] > max)
         max = vett[i];

   printf("Il massimo e': %d\n", max);
}
```

Esempio

- Cercare il massimo di un vettore il cui numero di elementi e gli elementi stessi sono immessi da tastiera

```
#include <stdio.h>
#define MAX_DIM 100
main() {
   int vett[MAX_DIM];
   int i, num, max;

   printf("Quanti elementi contiene il vettore? ");
   scanf("%d", &num);
   if (num > MAX_DIM)
      printf("Massimo %d elementi!\n", MAX_DIM);
   else {
      max = vett[0];
      for (i=1; i<num; i++)
         if (vett[i] > max)
            max = vett[i];

      printf("Il massimo e': %d\n", max);
   }
}
```


Esempio

- Convertire un numero decimale in binario

```
#include <stdio.h>
main()
{
   int n, i, j;
   int binario[32];

   printf("Inserisci il numero da convertire: ");
   scanf("%d", &n);

   printf("%d in binario vale: ", n);
   i = 0;
   while (n != 0) {
      binario[i] = n % 2;
      n = n / 2;
      i++;
   }
   for (j=i-1; j>=0; j--)
      printf("%d", binario[j]);
}
```

Esempio

- Scambiare gli elementi di un vettore (il primo con l'ultimo, il secondo con il penultimo…)

```
#include <stdio.h>
#define N 10
main()
{
   int i, j, temp;
   int vett[N];

   /* acquisizione del vettore (vedi slide 5) */

   i = 0;
   j = N-1;
   while (i != j) {
      temp = vett[i];
      vett[i] = vett[j];
      vett[j] = temp;
      i++;
      j--;
   }
}
```


Esempio

- Ricerca di un elemento in un vettore

```
#include <stdio.h>
#define N 10
main()
{
   int i, trovato, elemento;
   int vett[N];

   /* acquisizione del vettore (vedi slide 5) */
   printf("Inserisci l'elemento da cercare: ");
   scanf("%d", &elemento);

   trovato = 0;
   i = 0;
   while (i < N && !trovato) {
      if (vett[i] == elemento)
         trovato = 1;
      else
         i++;
   }

   if (trovato)
      printf("%d trovato in posizione %d\n", elemento, i);
   else
      printf("%d non contenuto nel vettore\n", elemento);
}
```

Esempio

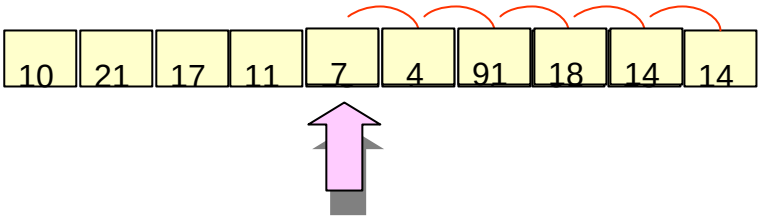
- Ricerca di un elemento in un vettore

```
#include <stdio.h>
#define N 10
main()
{
   int i, elemento;
   int vett[N];

   /* acquisizione del vettore (vedi slide 5) */
   printf("Inserisci l'elemento da cercare: ");
   scanf("%d", &elemento);

   i = 0;
   while (i < N) {
      if (vett[i] == elemento)
         break;
      else
         i++;
   }
   if (i < N)
      printf("%d trovato in posizione %d\n", elemento, i);
   else
      printf("%d non contenuto nel vettore\n", elemento);
}
```

```
for (i=0; i<N; i++)
   if (vett[i] == elemento)
      break;
```


Esempio

- Cancellazione di un elemento da un vettore



Esempio

- Cancellazione di un elemento da un vettore

```
#include <stdio.h>
#define N 10
main()
{
   int i, elemento;
   int vett[N];

   /* acquisizione del vettore (vedi slide 5) */

   printf("Inserisci l'elemento da eliminare: ");
   scanf("%d", &elemento);

   for (i=0; i<N && vett[i] != elemento; i++) ;

   if (i < N) { /* elemento trovato… cancellazione */
      for (j=i; j<N-1; j++)
         vett[j] = vett[j+1];
      /* i dati utili nel vettore sono ora N-1 */
      for (i=0; i<N-1; i++)
         printf("%d ", vett[i]);
   } else
      printf("elemento non trovato\n");
}
```

Non esegue nessuna istruzione (il corpo del ciclo è vuoto!)

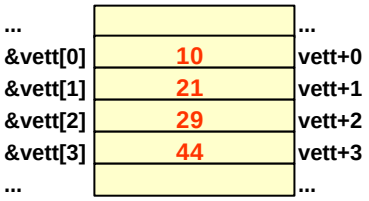
Esempio

- Inserimento ordinato in un vettore

```
#include <stdio.h>
#define N 10
main()
{
   int i, n, len;
   int vett[N];

   len = 0; /* lunghezza attuale del vettore */
   for (i=0; i<N; i++) {
      printf("Inserisci un valore: ");
      scanf("%d", &n);
      /* cerco la posizione del primo elemento del vettore maggiore di n */
      j = 0;
      while (j<len && n > vett[j])
         j++;
      /* creo spazio per inserire l'elemento in posizione j */
      for (k=len; k>j; k--)
         vett[k] = vett[k-1];

      /* ora posso inserire l'elemento */
      vett[j] = n;

      len++; /* il vettore contiene ora un elemento in più */
   }
}
```

Vettori e Puntatori

- Il nome di un vettore è un alias dell'indirizzo del primo elemento del vettore
  `int vett[10];`
  `vett` ° `&vett[0]`
- Aritmetica dei puntatori
  - ➔ Gli elementi di un vettore si trovano in locazioni contigue di memoria.
  - ➔ Es. **int vett[4] = {10, 21, 29, 44};**



Vettori e Puntatori

- `int vett[4];`
  `vett+1` non è l'indirizzo ottenuto sommando 1 all'indirizzo di partenza di `vett` bensì l'indirizzo ottenuto sommando `sizeof(int)` all'indirizzo di partenza di `vett`

  Restituisce la dimensione in byte del tipo passato come parametro.
- `Tipo vett[N];`
  `vett+i` è lindirizzo base di `vett` più `i*sizeof(Tipo)`
- **In generale:**

  `vett[i] = x;` ⟺ `*(vett+i) = x;`

Vettori e Puntatori

- Visto che il nome di un vettore rappresenta il puntatore al primo elemento, il vettore può essere visto come un puntatore.

- *Scrivere una funzione che inizializzi a 0 gli elementi di un vettore passato come parametro*

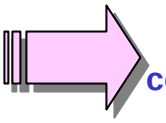
```
void Azzera(int vett[], int n) {
   int i;

   for (i=0; i<n; i++)
      vett[i] = 0;
}
```

- Il vettore deve essere modificato e può esserlo visto che **vett** rappresenta il suo indirizzo di partenza.
- Occorre passare la dimensione **n** del vettore (il numero di elementi che contiene).

Esercizi

- Implementare le funzioni del seguente programma

```
#define MAX_DIM 10
void main(void) {
   int   vett[MAX_DIM];
   int   dim, e, pos;
   float media;

   LeggiNumeri(vett, &dim);
   media = CalcolaMedia(vett, dim);
   printf("La media e': %f\n", media);
   printf("Inserisci l'elemento da cercare: ");
   scanf("%d", &e);
   pos = Ricerca(vett, dim, e);
   if (pos == -1)
      printf("%d non si trova nel vettore.\n", e);
   else
      printf("%d si trova in posizione %d.\n", e, pos);
   printf("Inserisci l'elemento da eliminare: ");
   scanf("%d", &e);
   if (Elimina(vett, &dim, e))
      printf("Elemento eliminato.\n");
   else
      printf("%d non presente nel vettore.\n", e);
}
```



Esercizi (...continuo)

- Prototipi (dichiarazione) delle funzioni

```
void LeggiNumeri(int vett[], int *dim);
/* Legge gli elementi del vettore vett. Termina l'inserimento
quando l'utente inserisce -1. Restituisce in dim il numero di
elementi inseriti */

float CalcolaMedia(int vett[], int dim);
/* Calcola la media degli elementi del vettore vett */

int Ricerca(int vett[], int dim, int e);
/* Ricerca e all'interno del vettore vett. Restituisce la
posizione dell'elemento se trovato altrimenti restituisce -1
*/

int Elimina(int vett[], int *dim, int e);
/* Elimina l'elemento e dal vettore vett ed aggiorna la
dimensione dim. Restituisce 1 se l'elemento è stato eliminato
altrimenti restituisce 0 */
```


void LeggiNumeri(int vett[], int *dim)

```
void LeggiNumeri(int vett[], int *dim)
{
   int i, e;

   i = 0;
   do {
      printf("Inserisci un elemento (-1=fine): ");
      scanf("%d", &e);
      if (e != -1) {
         vett[i] = e;
         i++;
      }
   } while (e != -1);

   *d = i;
}
```

1° modo

void LeggiNumeri(int vett[], int *dim)

```
void LeggiNumeri(int vett[], int *dim)
{
   int i, e;

   e = i = 0;
   while (e != -1) {
      printf("Inserisci un elemento (-1=fine): ");
      scanf("%d", &e);
      if (e != -1) {
         vett[i] = e;
         i++;
      }
   }

   *d = i;
}
```

2° modo

void LeggiNumeri(int vett[], int *dim)

```
void LeggiNumeri(int vett[], int *dim)
{
   int i, e;

   i = 0;
   while (1) { /* ciclo infinito */
      printf("Inserisci un elemento (-1=fine): ");
      scanf("%d", &e);
      if (e == -1)
        break;
      vett[i] = e;
      i++;
   }

   *d = i;
}
```

3° modo

float CalcolaMedia(int vett[], int dim)

```
float CalcolaMedia(int vett[], int dim)
{
   int i, somma;

   somma = 0;
   for (i=0; i<dim; i++)
      somma = somma + vett[i];

   return (float)somma/(float)dim;
}
```


int Ricerca(int vett[], int dim, int e)

```
int Ricerca(int vett[], int dim, int e)
{
   int i;

   i = 0;
   while (i < dim) {
      if (vett[i] == e)
         return i;
      else
         i++;
   }

   return -1;
}
```

1° modo

int Ricerca(int vett[], int dim, int e)

```
int Ricerca(int vett[], int dim, int e)
{
   int i;

   for (i=0; i<dim; i++)
      if (vett[i] == e)
         return i;

   return -1;
}
```

2° modo

int Ricerca(int vett[], int dim, int e)

```
int Ricerca(int vett[], int dim, int e)
{
   int i, trovato;

   i = trovato = 0;
   while (i < dim && trovato == 0) {
      if (vett[i] == e)
         trovato = 1;
      else
         i++;
   }

   if (trovato)
      return i;
   else
      return -1;
}
```

3° modo

int Ricerca(int vett[], int dim, int e)

```
int Ricerca(int vett[], int dim, int e)
{
   int i;

   i = 0;
   while (i < dim && vett[i] != e)
      i++;

   if (i < dim)
      return i;
   else
      return -1;
}
```

4° modo

int Elimina(int vett[], int *dim, int e)

```
int Elimina(int vett[], int *dim, int e)
{
   int i, p;

   /* cerco 'e' in 'vett' */
   p = Ricerca(vett, *dim, e);
   if (p == -1)
      return 0;

   /* compatto il vettore in modo da fare
      scomparire l'elemento di posizione 'p' */
   for (i=p; i<*dim - 1; i++)
      vett[i] = vett[i+1];

   /* ora il vettore contiene un elemento
      in meno… aggiorno la dimensione */
   *dim = *dim - 1;
}
```