Array

Aggragati di variabili omogenee ...

Cosa è un array ?

- È un insieme di variabili omogenee identificato da un indice, ad esempio
  - Se devo leggere 10 numeri dallo standard input e memorizzarli all'interno del mio programma è abbastanza scomodo usare 10 variabili diverse!

Array: esempio

```
int main (void){
  double a1,a2,a3,a4,a5,a6,a7,a8,a9,a10;
  double sum;
  printf("inserisci un numero: \n");
  scanf("%lf",&a1);
  printf("inserisci un numero: \n");
  scanf("%lf",&a2);
  ......
  printf("inserisci un numero: \n");
  scanf("%lf",&a10);
  /* es calcolo la somma ....*/
  sum = a1 + ... + a10;
  ......
  return 0;
}
```

Cosa è un array ?

- È un insieme di variabili omogenee identificato da un indice, ad esempio
  - Se devo leggere 10 numeri dallo standard input e memorizzarli all'interno del mio programma è abbastanza scomodo usare 10 variabili diverse!
  - Posso invece utilizzare un array di 10 variabili double e scorrerle utilizzando un indice intero .....

Array: esempio

Dichiarazione dell'array
Nome e numero di variabili
che servono (la *lunghezza* dell'array)

```
int main (void){
  double a[10];
  double sum = 0;
  int i;
  for (i=0; i<10; i++) {
     printf("inserisci un numero: \n");
     scanf("%lf",& a[i]);
}
/* es calcolo la somma e la stampo ....*/
  i=0
  while (i<10) {
   sum += a[i];
   i++;
  }
  printf("Somma: %f \n", sum);
  return 0;
}
```

Array: esempio

```
int main (void){
  double a[10];
  double sum = 0;
  for (i=0; i<10; i++) {
     printf("inserisci un numero: \n");
     scanf("%lf", &a[i] );
}
/* es calcolo la somma e la stampo ....*/
  i=0
  while (i<10) {
   sum += a[i];
   i++;
  }
  printf("Somma: %f \n", sum);
  return 0;
}
```

Accesso alla variabile di indice i.
In C le variabili di una array sono numerate da 0, in questo caso 0...9

Cosa è un array ?

- È un insieme di variabili omogenee identificato da un indice, ad esempio
  - Se devo leggere 10 numeri dallo standard input e memorizzarli all'interno del mio programma è abbastanza scomodo usare 10 variabili diverse!
  - Posso invece utilizzare un array di 10 variabili double e scorrerle utilizzando un indice intero .....
  - Servono anche a memorizzare vettori e matrici o qualsiasi altra informazione tabellare, lo vedremo fra un attimo

Array: dichiarazione e uso

- Per dichiarare un array è necessario definire il suo nome, il tipo e il numero delle variabili (la lunghezza) prima del suo uso,

```
double a[10];
```

tipo nome lunghezza

  - La lunghezza deve essere un valore costante o comunque noto a tempo di compilazione (standard ANSI, ne parleremo più avanti)
  - Alla dichiarazione si riserva memoria contigua per tutte le variabili nell'array. Se l'array è definito in un blocco la memoria viene riservata sullo stack.

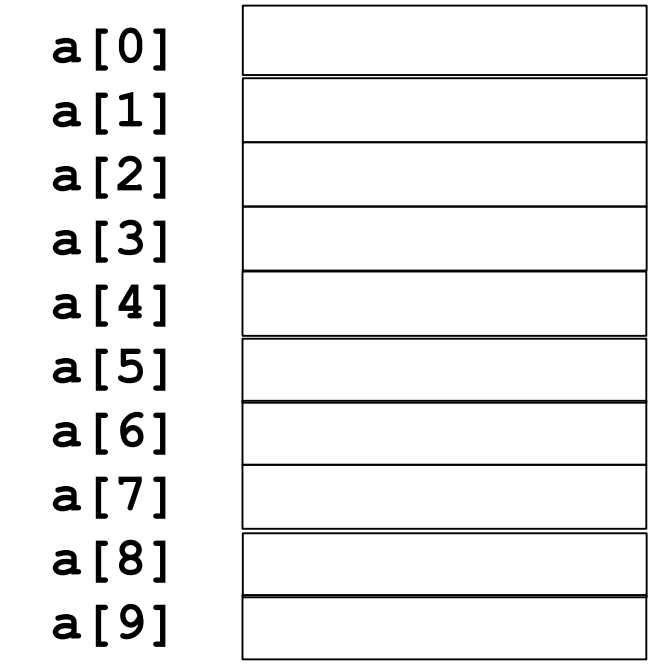
Array: dichiarazione e uso

```
double a[10];
```

tipo

nome

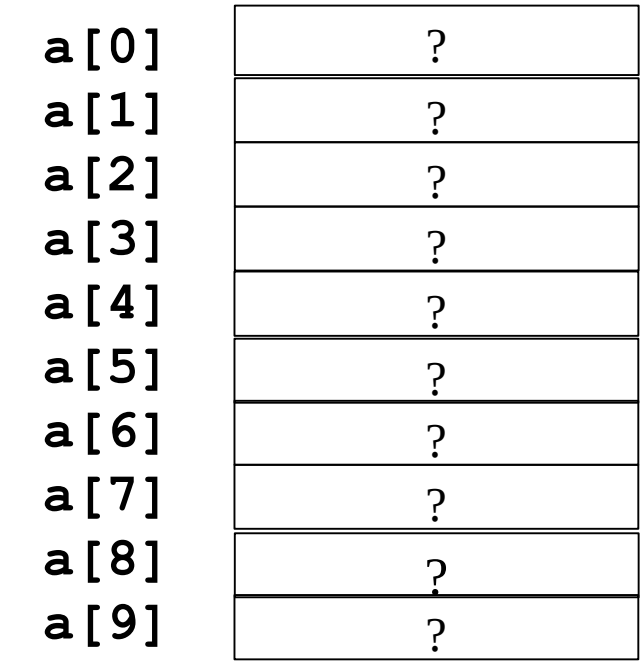
lunghezza



Array: dichiarazione e uso

```
double a[10];
```

tipo nome lunghezza

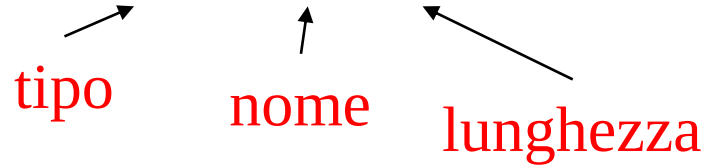


Come accade per le altre variabili l'array all'inizio contiene valori casuali che dipendono dal contenuto precedente della memoria allocata

..e deve essere *inizializzato* prima del suo uso

Array: dichiarazione e uso

```
double a[10];
```




```
a[5] = 15;
a[0] = a[5] + 2;
```

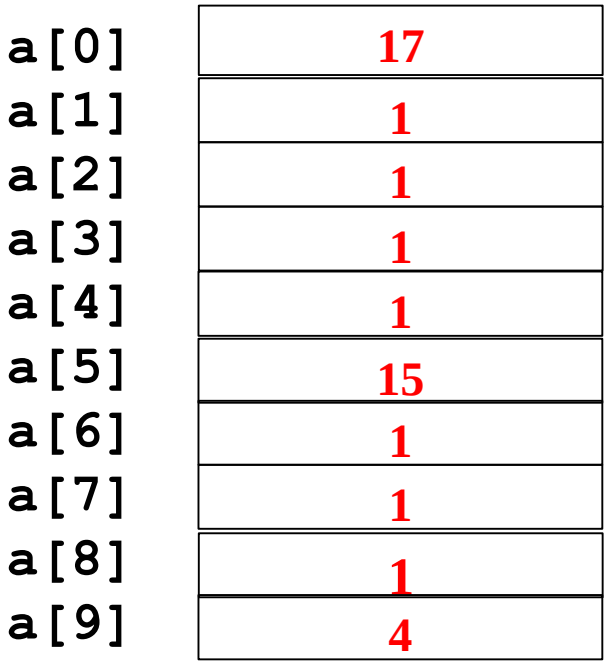
a[0]	17
a[1]	?
a[2]	?
a[3]	?
a[4]	?
a[5]	15
a[6]	?
a[7]	?
a[8]	?
a[9]	?

- Gli elementi dell'array si usano

Esattamente come normali variabili

Array: dichiarazione e uso

```
#define N 10
double a[N];
int i;

for(i=0;i<N; i++)
    a[i]=1;
a[5] = 15;
a[0] = a[5] + 2;
a[N-1] = 4;
```

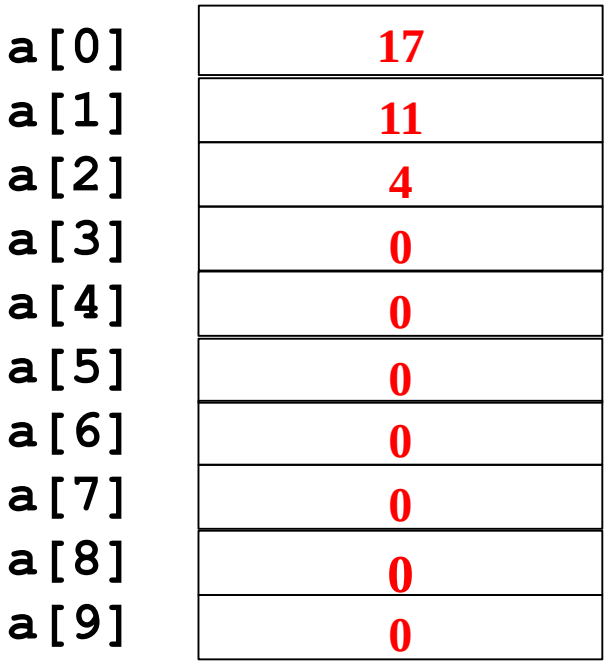


- È buona norma definire la lunghezza dell'array con una macro, come abbiamo già introdotto per le altre costanti

Array: inizializzazione alla dichiarazione

```
#define N 10
double a[N] = {17, 11, 4};
```

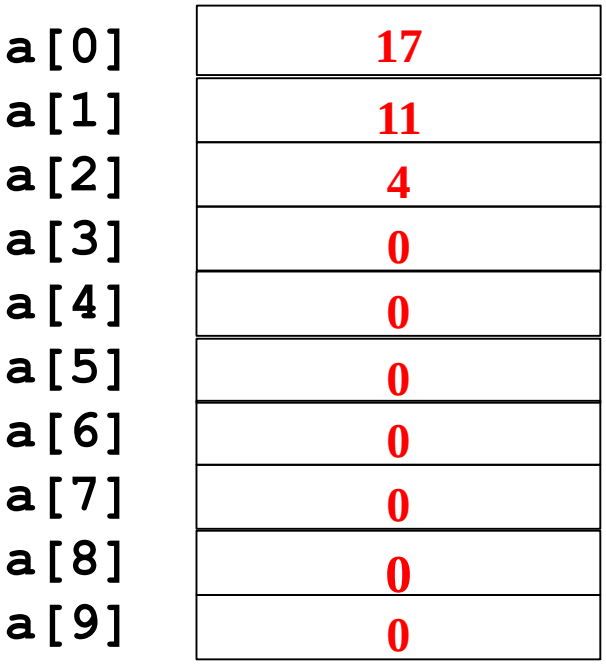
- Gli altri sono messi a 0
- Se sono di più degli elementi da errore in compilazione!



Array: non c'è l'assegnamento diretto

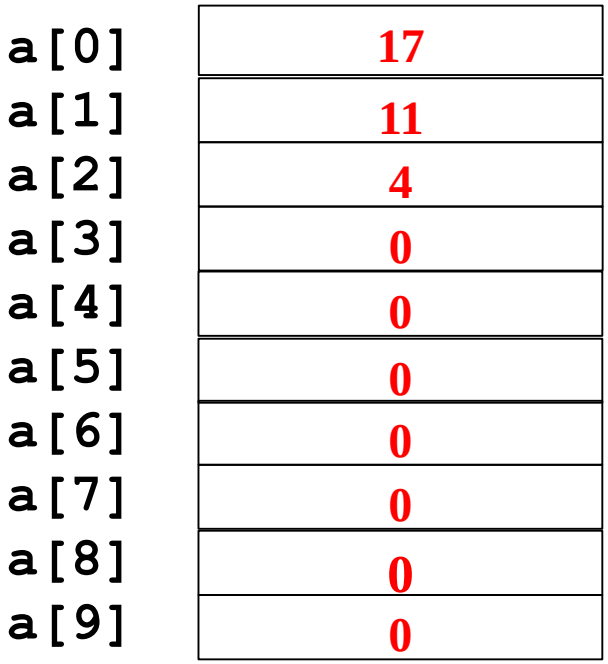
```
#define N 10
double a[N] = {17, 11, 4};
double b[N];

a = b;
```

- ???? Che effetto ha ????



Array: non c'è l'assegnamento diretto

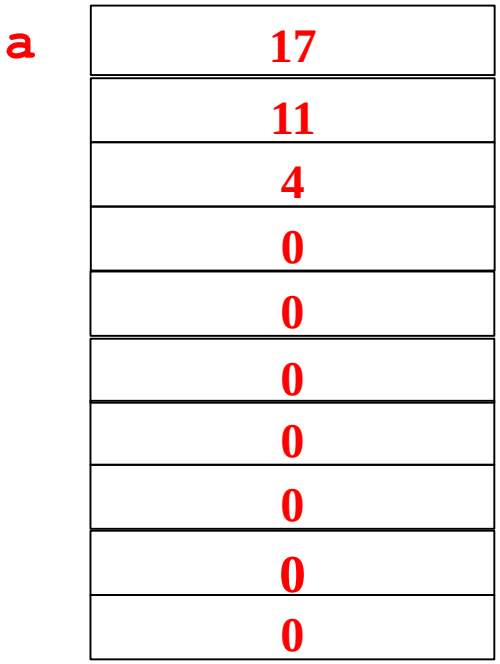
```
#define N 10
double a[N] = {17, 11, 4};
double b[N];


a = b;
```



- Da errore in compilazione
- L'unica possibilità è usare un ciclo per copiare gli elementi
- O usare la funzione di libreria **`memcpy()`** per copiare tutta la memoria (libreria **`string.h`**)

Array: non c'è l'assegnamento diretto

```
#define N 10
double a[N] = {17, 11, 4};
double b[N];
/* voglio copiare tutto b in
a */
a = b;  /* errato */
```



- L'errore da un messaggio un po' strano

Ma il motivo di fondo è che il nome di un array è un puntatore costante (non modificabile) che è uguale all'indirizzo di memoria dal quale è allocato l'array

- Ne riparleremo quando trattiamo i puntatori

Esempio: filtro i valori maggiori

```
#define N 10
int main (void){
  double a[N], filtro;
  int i;
  for (i=0; i<N; i++) ){
    printf("inserisci un numero: \n");
    scanf("%lf",&a[i]);
  }
  printf("inserisci il filtro: \n");
  scanf("%lf",&filtro);
  for (i=0; i<N; i++)
    if ( a[i] > filtro ) printf("%f\n",a[i]);
  return 0;
}
```

Esempio: istogramma ...

- Leggere da standard input una sequenza di caratteri (terminata da '\n')
- Stampare le frequenze delle cifre da 0 a 9

Caratteri

- Interi rappresentati su un singolo byte
  - Rappresentano caratteri alfanumerici attraverso il codice ASCII (American Standard Code for Information Interchange)
- Esempi di valori:

carattere	'0'	...	'9'	':'	';'	'<'
Val. decimale	48		57	58	59	60
carattere	'a'	...	'z'	'{'	'|'	'}'
Val. decimale	97	...	122	123	124	125
carattere	'A'	...	'Z'			
Val. decimale	65	...	90			

Esempio: istogramma ...

- Leggere da standard input una sequenza di caratteri (terminata da '\n')
- Stampare le frequenze delle cifre da 0 a 9
- Possiamo utilizzare un array di 10 posizioni nel quale memorizziamo le frequenze dei caratteri da '0' a '9'

0	1	2	3	4	5	6	7	8	9

Esempio: istogramma

```
int main (void){
  int h[N], i;
  char c;
  for (i=0; i<N; i++) h[i]=0;
  c = getchar();
  while (c != '\n') {
    switch (c) {
      case '0': h[0]++; break;
      case '1': h[1]++; break;
      case '2': h[2]++; break;
      case '3': h[3]++; break;
      case '4': h[4]++; break;
      case '5': h[5]++; break;
      case '6': h[6]++; break;
      case '7': h[7]++; break;
      case '8': h[8]++; break;
      case '9': h[9]++; break;
    }
   c = getchar();
  } /* segue ... */
```

Esempio: istogramma

```
int main (void){
  int h[N], i; char;
  ....


  for (i=0; i<10; i++)
    printf("Frequenze di %d = %d\n",i, a[i]);
  return 0;
}
```

Caratteri

- Interi rappresentati su un singolo byte
  - Rappresentano caratteri alfanumerici attraverso il codice ASCII (American Standard Code for Information Interchange)
- Esempi di valori:

carattere	'0'	...	'9'	':'	';'	'<'
Val. decimale	48		57	58	59	60
carattere	'a'	...	'z'	'{'	'|'	'}'
Val. decimale	97	...	122	123	124	125
carattere	'A'	...	'Z'			
Val. decimale	65	...	90			

Esempio: istogramma

```
#define N 10
int main (void){
  int h[N], i;
  char c ;
  for (i=0; i<N; i++) h[i]=0;

  c = getchar();
  while (c != '\n') {
    if ( c >= '0' && c <= '9' ) {
          i = c - '0';
          h[i]++;
    }
   c = getchar();
  }
  for (i=0; i<N; i++)
    printf("Frequenze di %d = %d\n",i, a[i]);
  return 0;
}
```

E gli array con 2 o più dimesioni ?

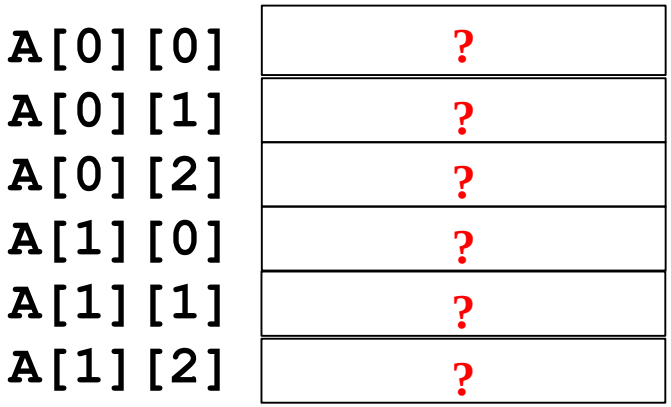
- E possibile definire in C array con due o più dimensioni

```
double a[10][5];
/* dichiara una matrice con 10 righe e 5
colonne di double */
double c[3][3][3];
/* dichiara una cubo 3x3x3 */
```

- Vengono memorizzate in celle di memoria successive per riga
  - o comunque a partire dalla dimensione minore

Esempio: due dimensioni ...

```
#define N 2
#define M 3

double A[N][M] ;
```



Esempio: somma di matrici

```
#define N 10
int main (void){
  int a[N][N], b[N][N], c[N][N];
  int i, j;
  /* lettura delle due matrici a e b dallo
     standard input lo svilupperemo in laboratorio */
   .....
 /* calcolo della somma */
 for (i=0; i < N; i ++)
   for (j=0; j < N; j ++)
     c[i][j]=a[i][j]+b[i][j]


  /* eventuale stampa del risultato ... */
}
```

Esempio: tre dimensioni ...

```
#define M 2

double A[M][M][M] ;
```