Titolo nota 24/09/2015

Passaggio dei parametri per valore

(possibile modificare il valore delle variabili da parte di una procedura passando "PER VALORE GLI INDIRIZZI"

```
void incr (int * n)
{ *n = *n + 1;}
   ⋮
incr(&x);
```

scanf

comando per le letture di valori dell'esterno (da tastiera, da file, ...) e l'assegnamento di tali valori a variabili.

```
scanf("%d", &x)
```

stringa di caratteri che indica il tipo di valore da leggere.

variabile da modificare

procedure predefinite in una LIBRERIA (collezione di procedure già scritte)

la definizione di scanf si trova in una libreria di operazioni STANDARD di INPUT/OUTPUT

stdio

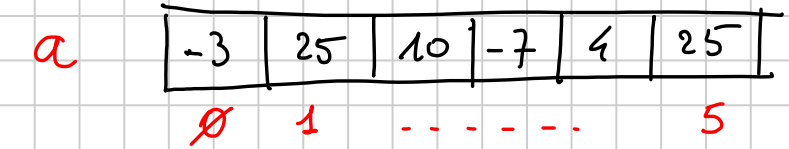
Se dobbiamo leggere un valore intero dobbiamo indicare la stringa "%d"

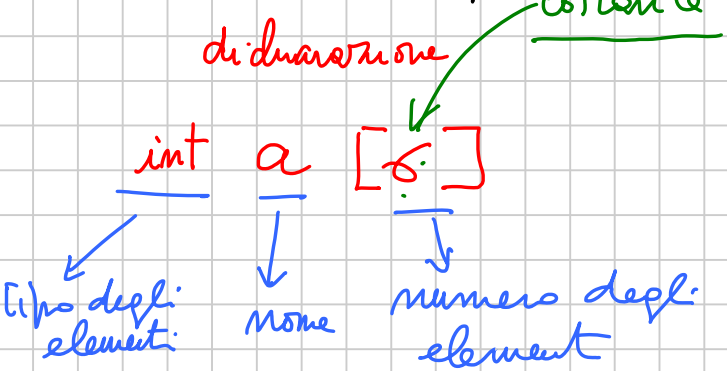
bisogna leggere un valore intero

ARRAY

Titolo nota 24/09/2015

Strutture dati : collezione di elementi tutti dello stesso tipo





→ dato del problema con una struttura (cioè è un dato non atomico)

primo elemento dell'array a → a[0]

```
x = a[0] + 1;
a[0] = a[0] + 1;
```

```
{ int x;
  scanf("%d", &x);
  { int a[x];   No
```

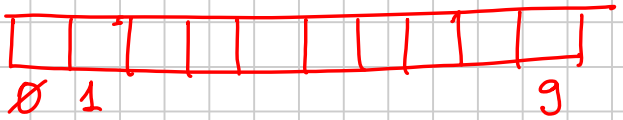
costante

programma che azzera tutti gli elementi di un array di interi

Titolo nota 24/09/2015

```
# include ...

int a[10];
```




```
main()
{ int i = 0;
  while (i < 10)
      { a[i] = 0;
        i = i+1;
      }
}
```

particolare perché fin dall'inizio sappiamo quante volte sarà eseguito il corpo del while!

I terazione DETERMINATA

for permette di scrivere facilmente una iterazione determinata!

la variabile di controllo del for (usiamo la variabile i)

Titolo nota 24/09/2015

```
for (i=0; i<10; i=i+1)  a[i]=0;
```

inizializzazione della variabile di controllo

condizione di iterazione (fino a quando il for deve essere eseguito?)

aggiornamento della variabile di controllo

```
main ()
{ int i;
  for (i=0; i<10; i = i+1)
      a[i] =0;
}
```

i++

Titolo nota 24/09/2015

modulo ....

```
int a[10];
int b[15];
void azzera (int v[])
{ int i;
  for (i=0; i<10; i++)
     v[i]=0;
}

main ()
{   azzera (a);
    azzera (b);
}
```

l'argomento potrà essere un array di interi DI QUALSIASI LUNGHEZZA

non c'è bisogno di *

Se passato come argomento un array, il C in realtà passa alla procedura l'indirizzo di a!

Titolo nota 24/09/2015

indovole ....

```
int a[10];
int b[15];
void azzera (int v[], int dim)
{ int i;
  for (i=0; i<dim; i++)
      v[i]=0;
}

main ()
{   azzera (a, 10);
    azzera (b, 15);
}
```

a[i] errore i=10

a[9] 10

i < 9

Titolo nota 24/09/2015

```
{ int a [10];
  int x = 5;

  a[1] = 3;
```

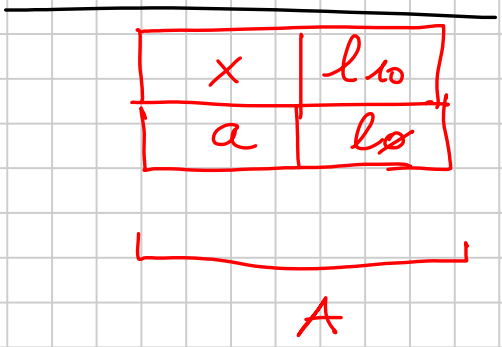
$l_0 + 1 = l_1$

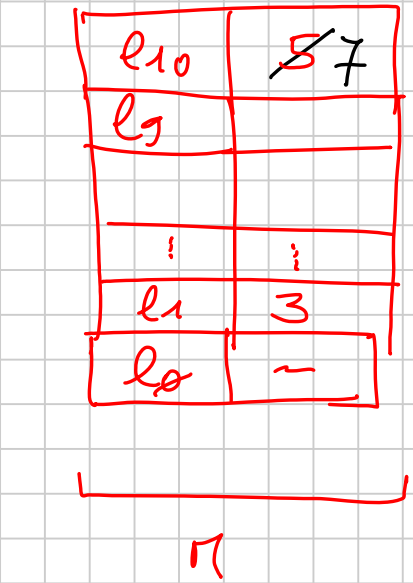
```
  n = 11;
  a[n] = 5
```

$l_0 + 11 = l_{11}$ errore

```
  n = 10
  a[n] = 7;
```

$l_0 + 10 = l_{10}$





Leggere tutti gli elementi di un array

Titolo nota 24/09/2015

```
#include stdio
int a [10];
int b [15];
void leggi (int v[], int dim)
  { int i;
    for (i=0; i<dim; i++)
        scanf ("%d", &v[i]);
  }
main()
{   leggi (a, 10);
    leggi (b, 15);
```

```
{ int x; int enste;
  scanf("%d", &x);
enste = member(x, a, 10);
```

0 falso

1 vero

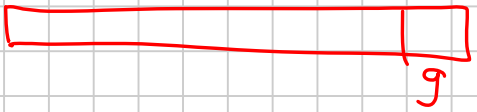
```
int member (int el, int v[],
            int dim)
{ int trovato = 0;                    NO
  int i;
  for (i=0; i<dim; i++)
    if (v[i] == el) trovato = 1;
} return trovato;
```

Titolo nota 24/09/2015

```
int member (int el, int v[], int dim)
{
    int i = 0;
    while (i < dim && v[i] != el)
        i = i + 1;
    return v[i] == el;
}
```

and in C

$(i == dim \;\&\&\; v[dim] != el)$



$i = 9 < 10$

nel caso in cui $(i == dim)$

quanto vale la condizione?

La condizione è short-circuited