PUNTATORI

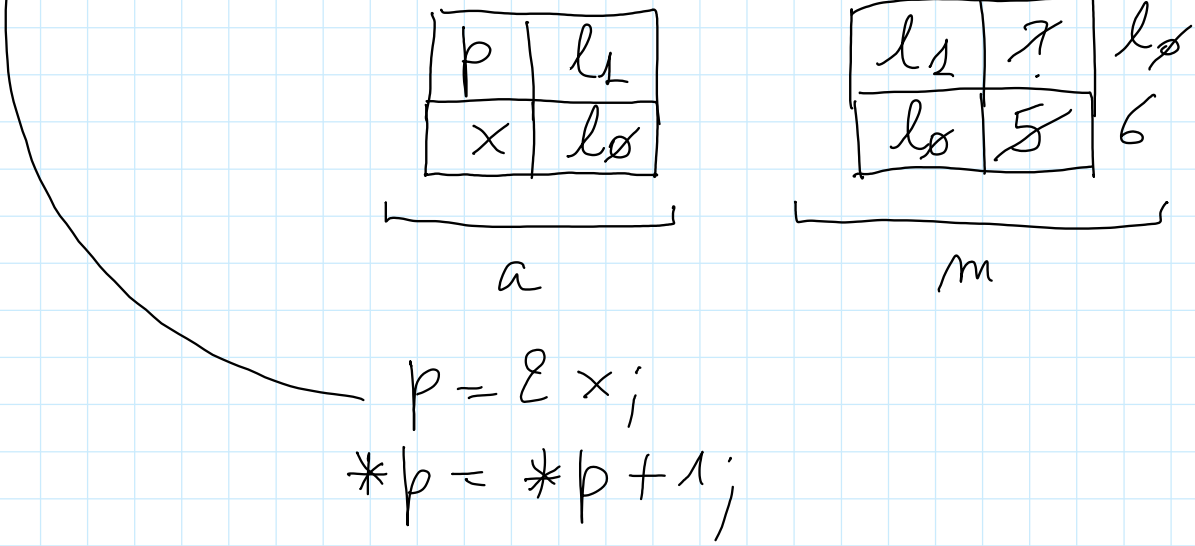
Variabili il cui valore in memoria è un indirizzo (interpretato come un indirizzo)

```
{ int x = 5;
  int * p;
  p = &x;
  *p = *p + 1;
  p = &x;
  *p = *p + 1;
```

`*p` → il valore puntato da p

`p` → il valore di p

`&p` → l'indirizzo di p




```
  p = &x;
  *p = *p + 1;
}
```

MODALITÀ DI PASSAGGIO DEI PARAMETRI IN C

È "PER VALORE"

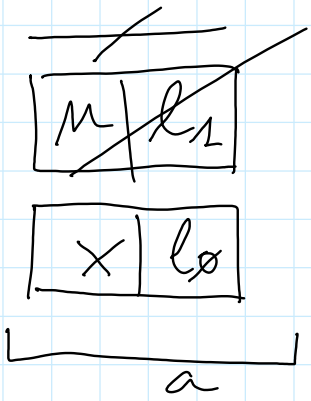
Quando una funzione o procedura viene chiamata si crea, sull'ambiente a nella memoria, un frame per contenere le variabili corrispondenti ai parametri (formali).

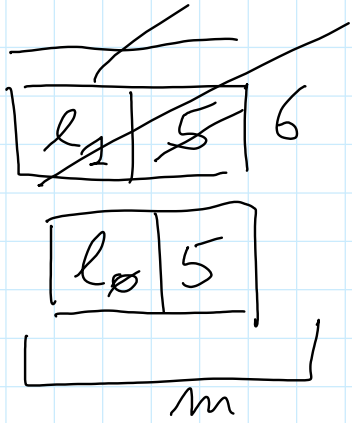
Alle variabili creati viene inizialmente assegnato il VALORE degli argomenti (parametri attuali)

```
void inca (int n)
{
  n = n+1;
}

main()
{ int x = 5;
  inca (x);
  :
```

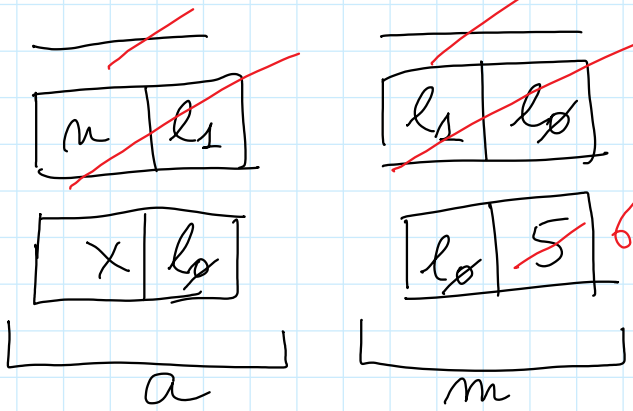
5





```
void inca (int * n)
{
  *n = *n+1;
}
```

```
main()
{ int x = 5;
  inca (&x);
  :
```



Supponiamo di voler scrivere una funzione o procedura che mi serve per modificare il valore di una variabile con la somma del suo valore più il valore di un'altra variabile

```
{ int x = 5;
  int y = 10;
  x = x + y;
```

La voglio fare con una funzione o una procedura.

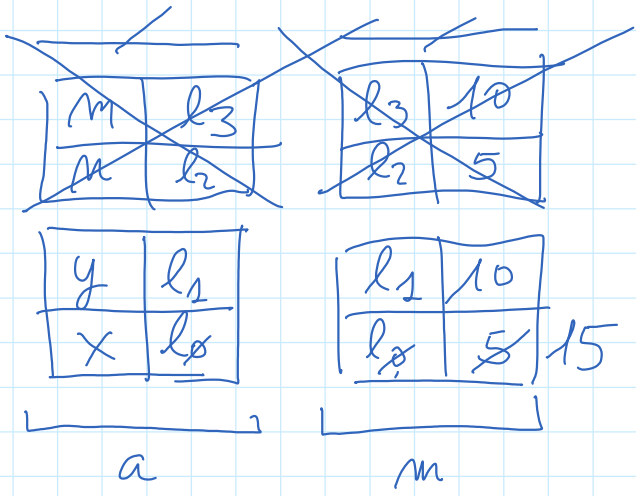
```
int sum (int n, int m)
{
   return n+m;
}

main()
{
   int x=5;
   int y=10;
   x = sum (x, y);
}
```

15

5 10

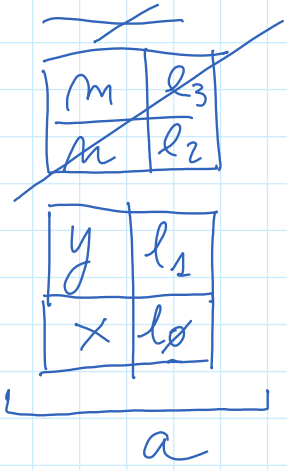
15



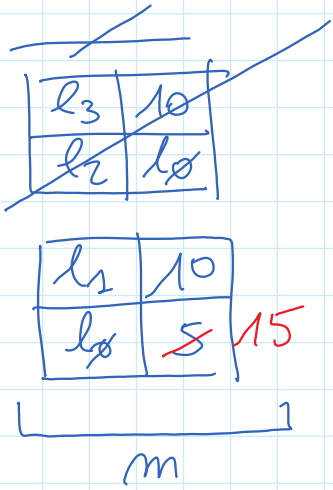
```
void sum (int *n, int m)
{  *n = *n + m;
}
main()
{  int x = 5;
   int y = 10;
   sum (&x, y);
```

15

l_0 10



a



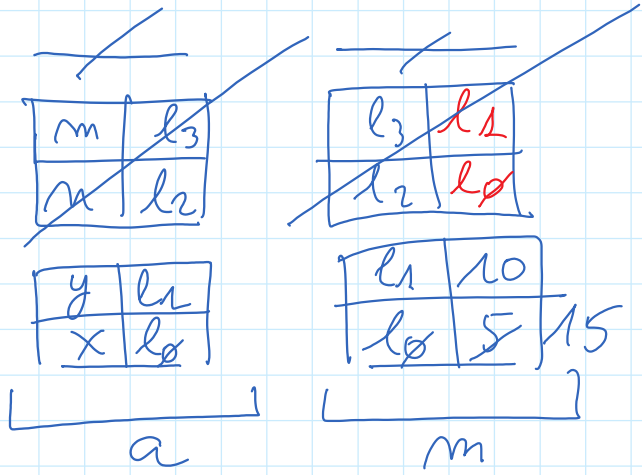
m

```
int Sum (int * n, int * m)
{ return *n + *m;
}.

main()
{ int x = 5;
  int y = 10;
  x = Sum (&x, &y);
}
```

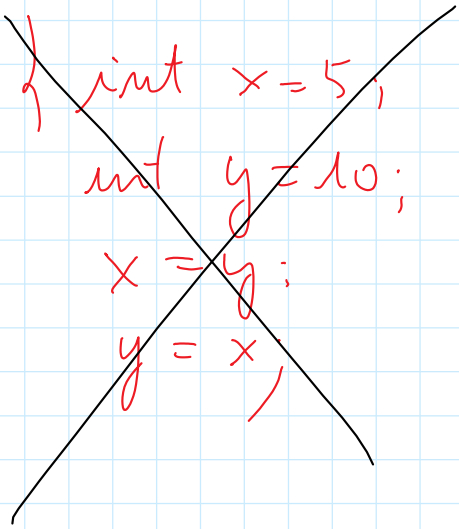
15

Funzione che non richiede programmare in questo modo



Scambiare il valore di due variabili

```
{ int x = 5;
  int y = 10;
  int temp = x;
  x = y;
  y = temp;
  :
```

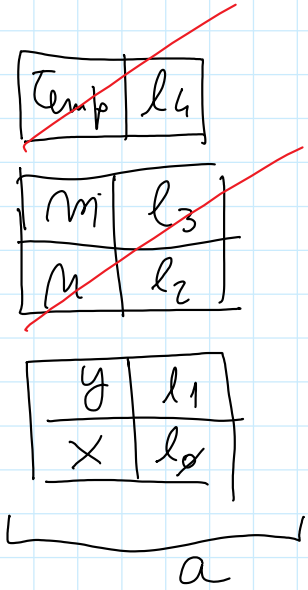


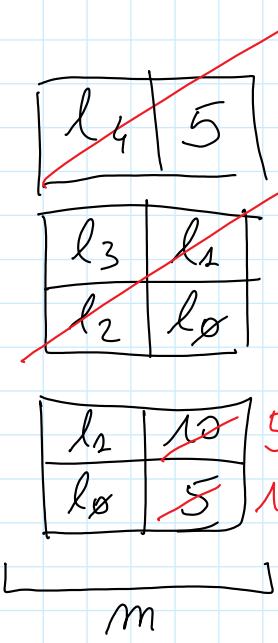

```
void swap (int * n, int * m)
{ int temp = *n;
  *n = *m;
  *m = temp;
}
main()
{ int x = 5;
  int y = 10;
  swap (&x, &y);
```

10

5

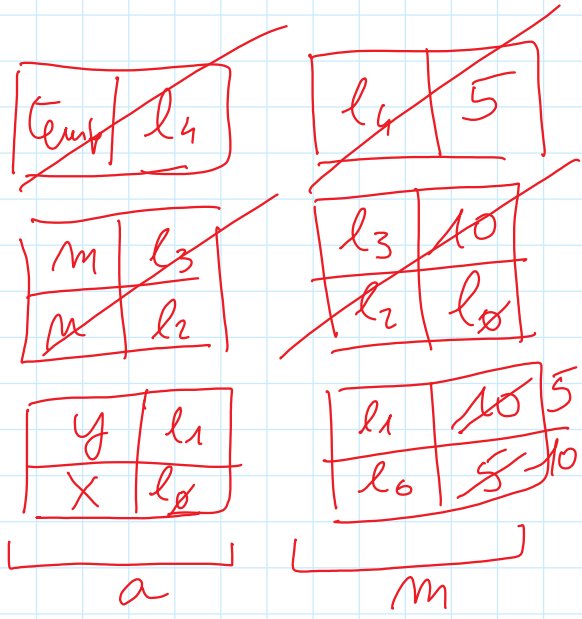
l_0 l_1





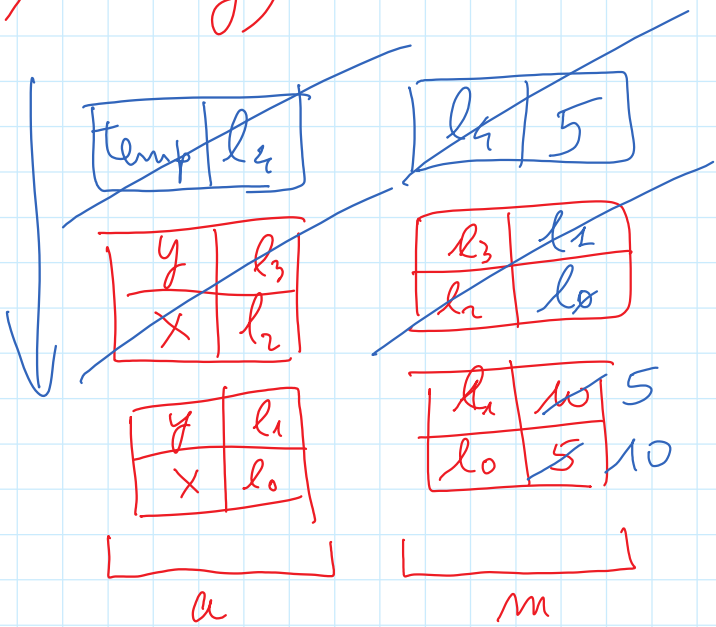
Funzione che modifica il valore di una variabile e restituisce come risultato il suo vecchio valore

```
int swap (int *n, int m)
{ int temp = *n;
  *n = m;
  return temp;
}
main()
{ int x = 5;
  int y = 10;
  y = swap(&x, y);
}
```



```
void swap (int *x, int *y)
{ int temp = *x;
  *x = *y;
  *y = temp; }

main()
{ int x = 5;
  int y = 10;
  swap(&x, &y);
```



l0 l1

```
void swap (int *x, int *y)
{ int x = *x;
```

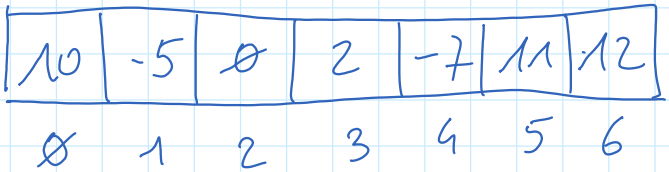
non va fatto!

ARRAY struttura di dati fornita come struttura di base della maggior parte dei linguaggi imperativi

(Collezione di dati con particolari operazioni)

Sequenza di dati (tutti dello stesso tipo) selezionabili attraverso un indice.

Graficamente



dichiarazione di un array in C

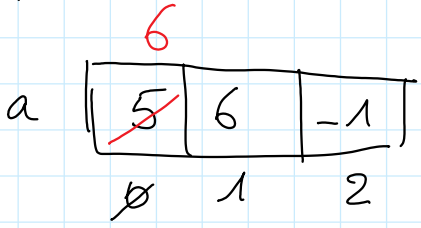
numero degli elementi

```
int a[3];
```

nome array

tipo degli elementi

Come si individuano gli elementi di a?



Ciascuna casellina è una variabile

```
a[0] = 5;
a[1] = 6;
a[2] = -1;
a[0] = a[0] + 1;
```

6

a[ ]

nome array

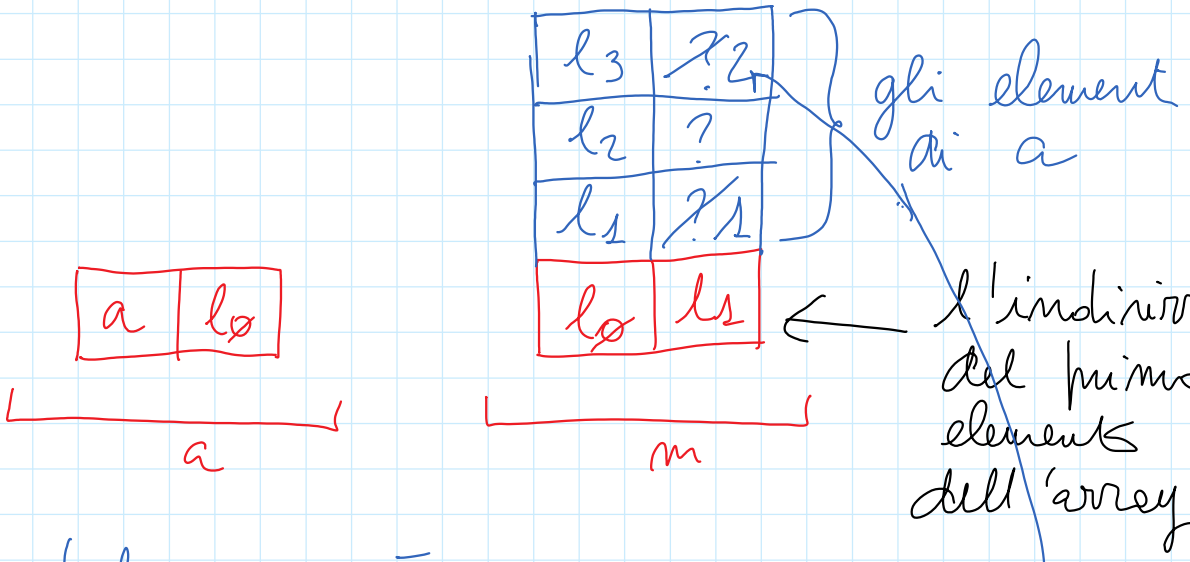
indice

a[4] non esiste!!

C non lo controlla.

Come viene memorizzato un array nello stack

```
int a[3];
```



La variabile a è un puntatore

```
a[0] = 1;      *(a + 0) = 1;
```

$l_1 + 0 = l_1$

$*l_1$

```
a[2] = a[0] + 1;
```

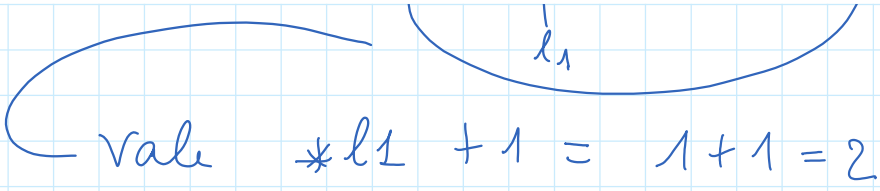
(Viene tradotto)

$l_1 + 2 = l_3$

$*(a+2) =$

$*(a+0) + 1$

l_1


l1
vale *l1 + 1 = 1 + 1 = 2

```
{ int a[3];
  int i=0;
  while (i<3)
    { a[i]=0;
      i=i+1;
    }
  ...
```

```
{ int a[3];
  int i=0;
  while (i<3)
    { scanf("%d", &a[i]);
      i=i+1;
    }
```

&(*(a+i)) — loc. dell'el. da modif.

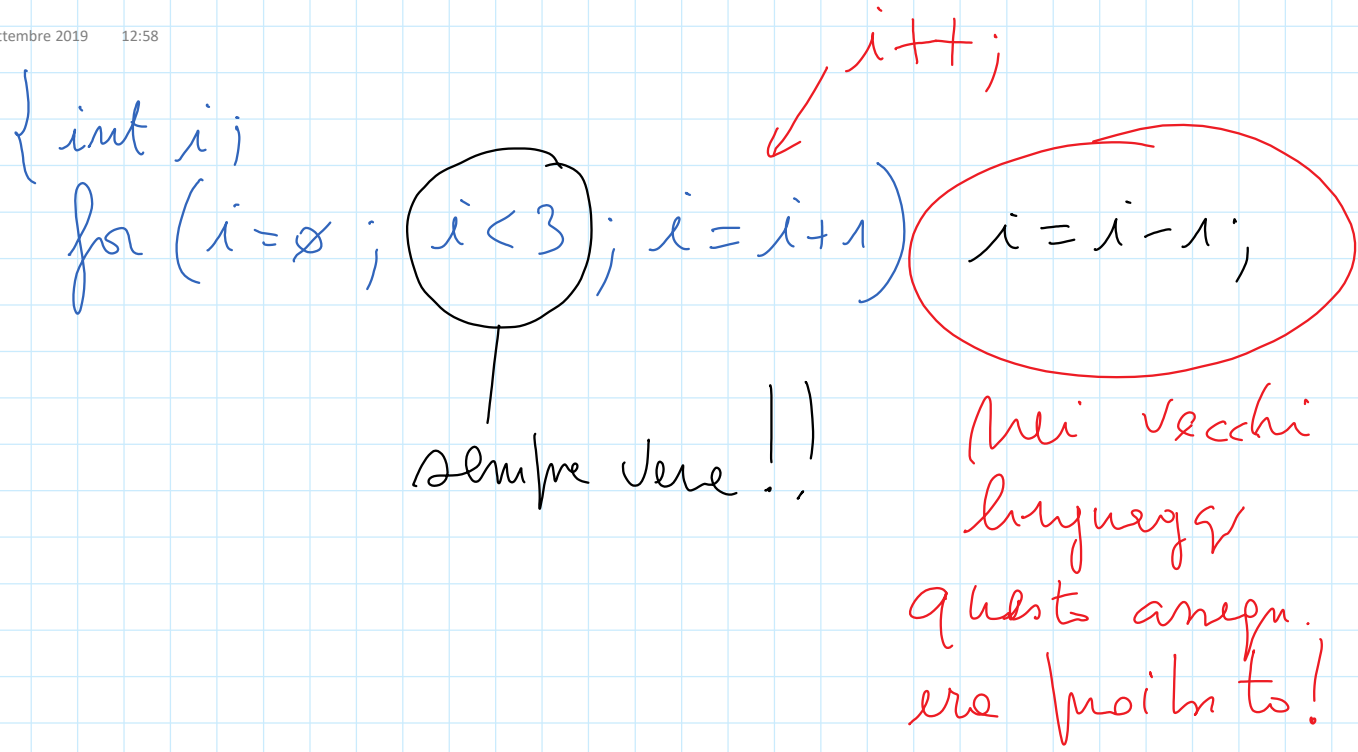
Comando for

Tradizionalmente il for era l'iterazione determinata (il numero di iterazioni era finito)

```
int i;
for (i=0; i<3; i=i+1) Comando
```

- `i=0` : inizializzazione delle variabile di controllo
- `i<3` : condizione di iterazione
- `i=i+1` : incremento della variabile di controllo
- `Comando` : corpo del for

- inizializza la variabile di controllo
- finché è vera la condizione di iterazione si esegue il "Comando" (corpo del for) e si incrementa la variabile di controllo


i++;
int i;
for (i=0; i<3; i=i+1)
i=i-1;
sempre vera!!
nei vecchi linguaggi questo assegn. era proibito!