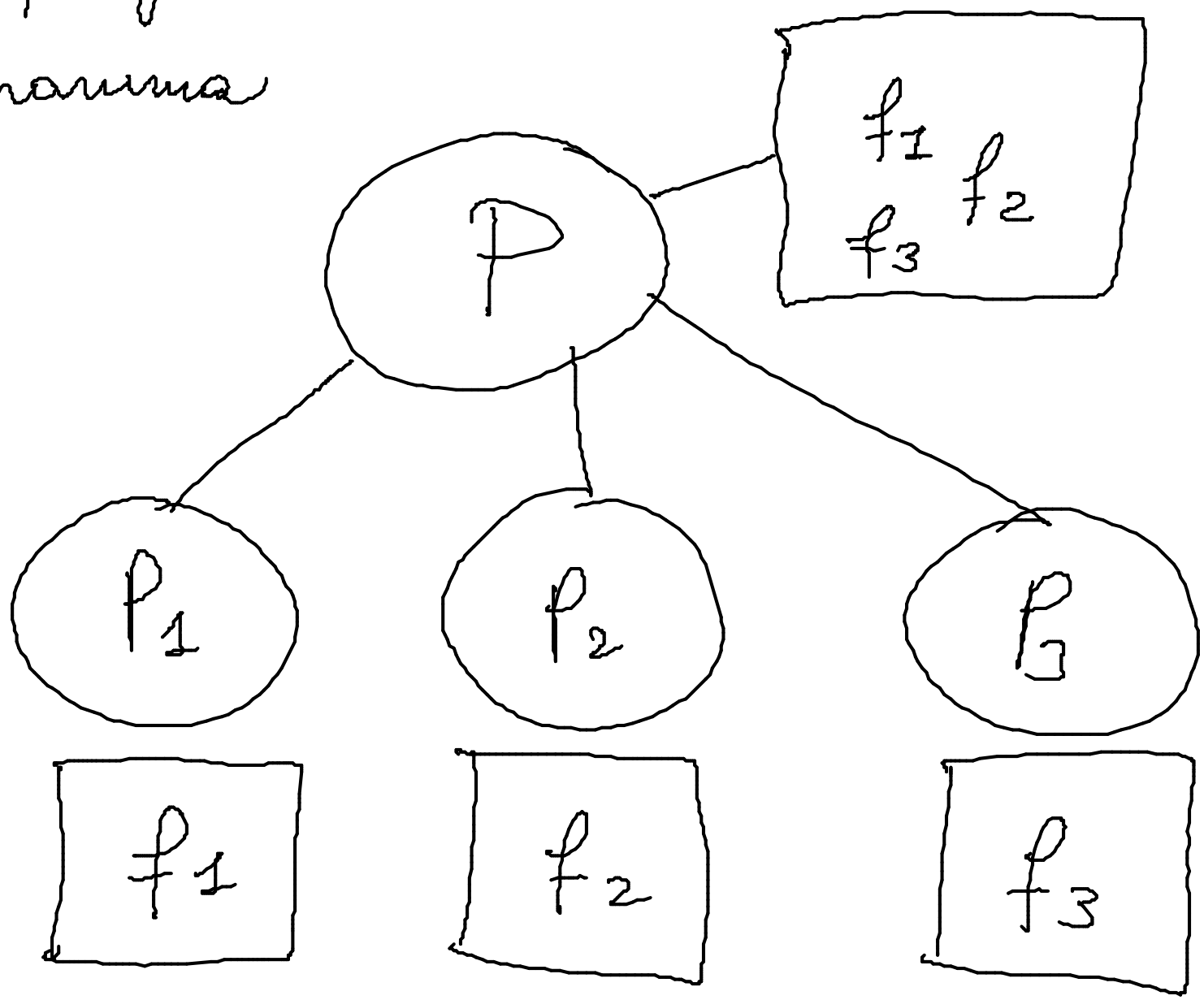
(PROCEDURE e) FUNZIONI

- Sottoprogrammi richiamabili all'interno di un programma



MINIMO COMUNE MULTIPLO

intestazione → int mcm (int x, int y)

blocco →
{
calcola il minimo comune multiplo tra x y
}

⟹

PROG. PRINCIPALE

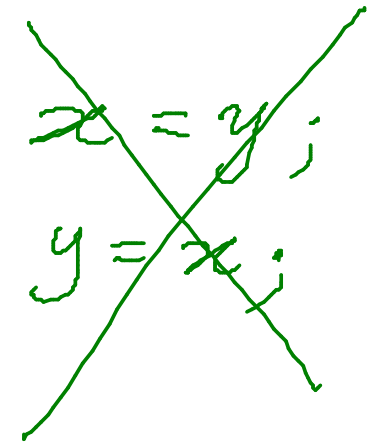
{
⋮
mcm (a, b)
⋮
mcm (c, d)

cfr. : let mcm x y = ... ;
⋮
let f z w = ... (mcm z w)

- Le funzioni permettono di definire nuovi OPERATORI parametrici
- Il loro uso ha senso all'interno di espressioni
- Le PROCEDURE sono astrazioni dei COMANDI permettono di definire COMANDI PARAMETRICI e il loro uso ha senso all'interno di blocchi; ovunque possono essere utilizzati comandi

- SCAMBIARE il valore di due variabili

```
void swap( . x.- -y )
{ int t;
  t = x;
  x = y;
  y = t;
}
```



No

FUNZIONI

```
int f (int x, int y)
{
   .... corpo ....
   return (espressione);
}
```

- `int` → tipo del risultato calcolato
- `f` → nome della funzione
- `(int x, int y)` → sequenza di parametri
- `.... corpo ....` → blocco che definisce come la funzione calcola il risultato

PARAMETRI FORMALI

```
int quadrato (int x)
{
    int r;
    r = x * x;
    return r;
}
```

```
int mcd (int x, int y)
{
    while (x != y)
        if (x>y) x= x-y;
            else y = y-x;
    return x;
}
```

Definizione o dichiarazione della funzione

```
{   PROGRAMMA PRINCIPALE
    int z, w, a, b;
    z = mcd (a, b);
```

PARAMETRI ATTUALI (actual)

Chiamata della funzione

```
{ int z; int q;
  ...
  z = 3;
  q = quadrato(z+1);
}
```

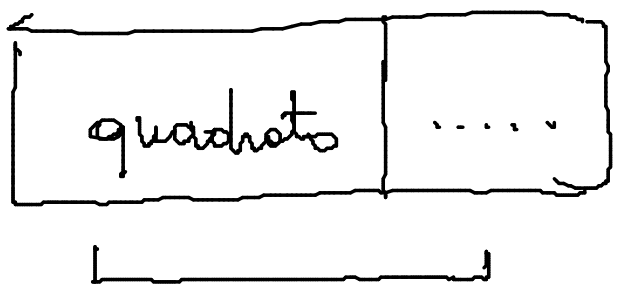
per. attuale

```
{ int x = z+1;
  int r;
  r = x * x;
  q = r;
}
```

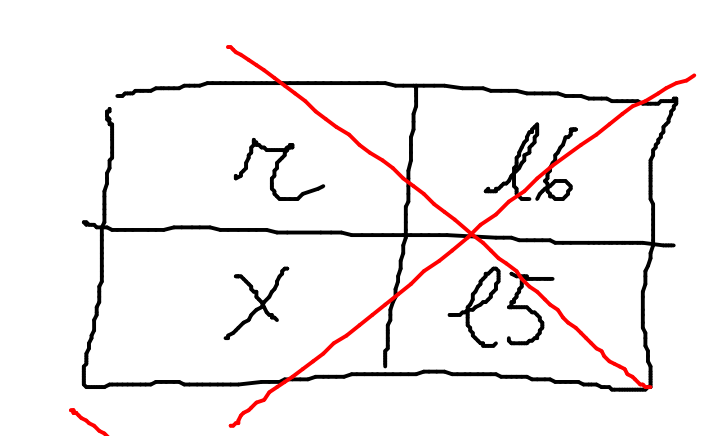
return r

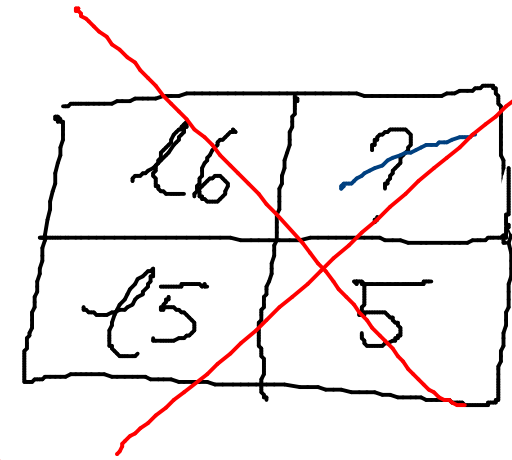
q	l2
z	l1

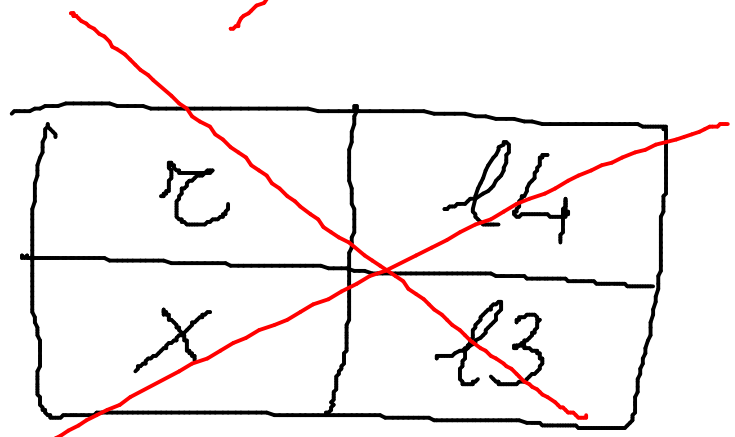
l2	?
l1	3

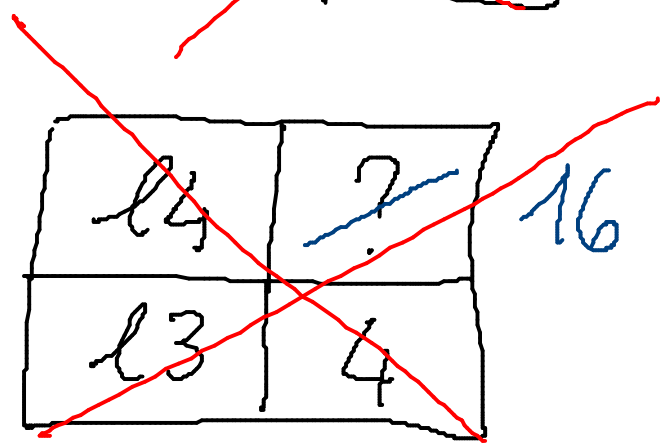


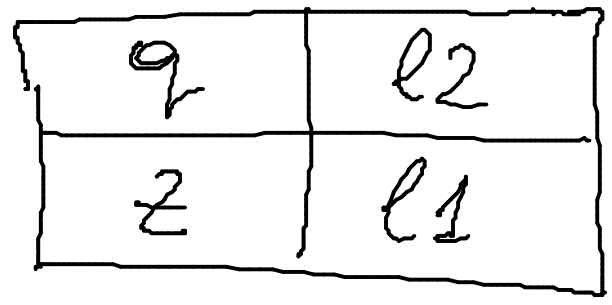
```
{ int z; int q;
  :
  z = 3;
  q = quadrato(z+1);
  :
  q = quadrato(5);
}
```

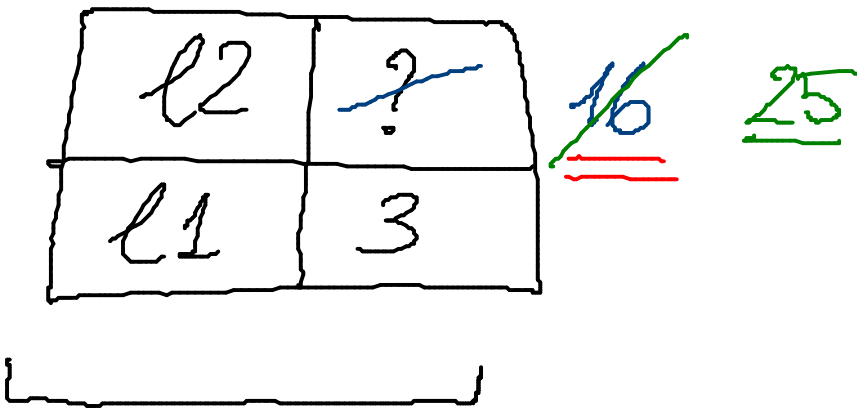












Al momento della CHIAMATA DI FUNZIONE

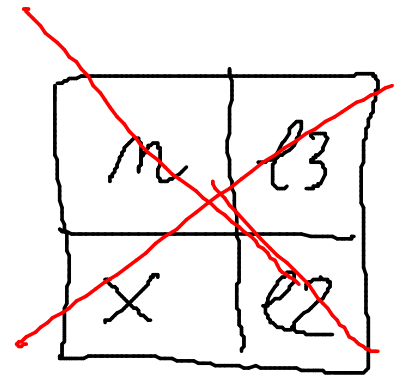
1) i parametri formali vengono "trattati" come variabili locali alla funzione che vengono inizializzate al VALORE dei corrispondenti parametri attuali

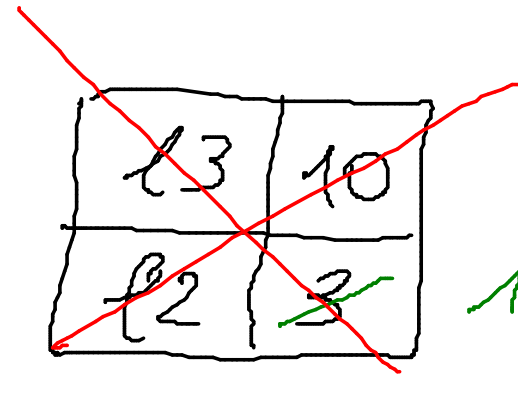
2) la chiamata della funzione "sta per" il valore dell'espressione calcolata dopo l'esecuzione del blocco (corpo) della funzione in corrispondenza del comando "return"

PROCEDURE

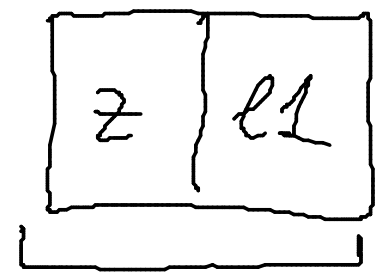
```
void  incr ( int x , int n)
      {  x = x + n;
      }

{  int z;
   z = 3;
   incr ( z , 10);
}
```

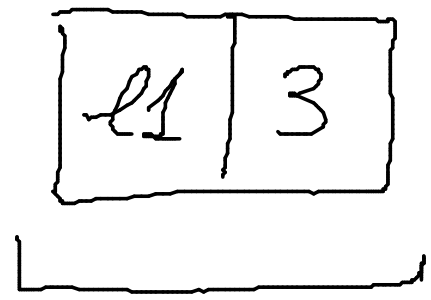
```
{ { int x = z;
    int n = 10;
    x = x + n;
  }
}
```





13

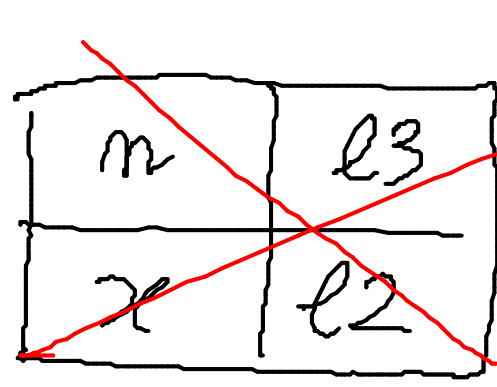


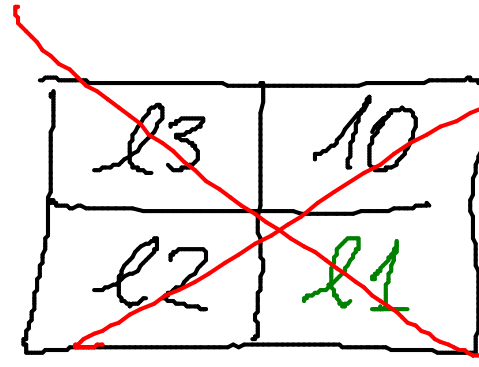


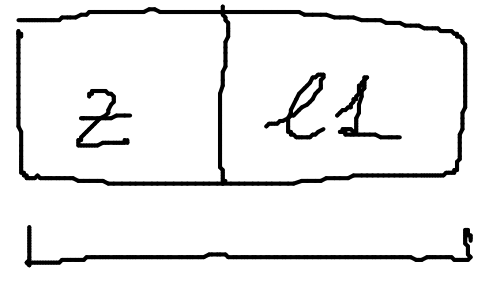
PASSAGGIO DEI PARAMETRI

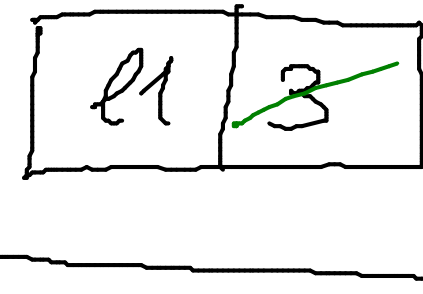
- Il C utilizza il passaggio dei parametri PER VALORE: la variabile corrispondente al parametro formale viene inizializzata al VALORE del parametro attuale

```
void incr (int *x, int n)
  {
    *x = *x + n;
  }

{
  int z;
  z = 3;
  incr (&z, 10);
```



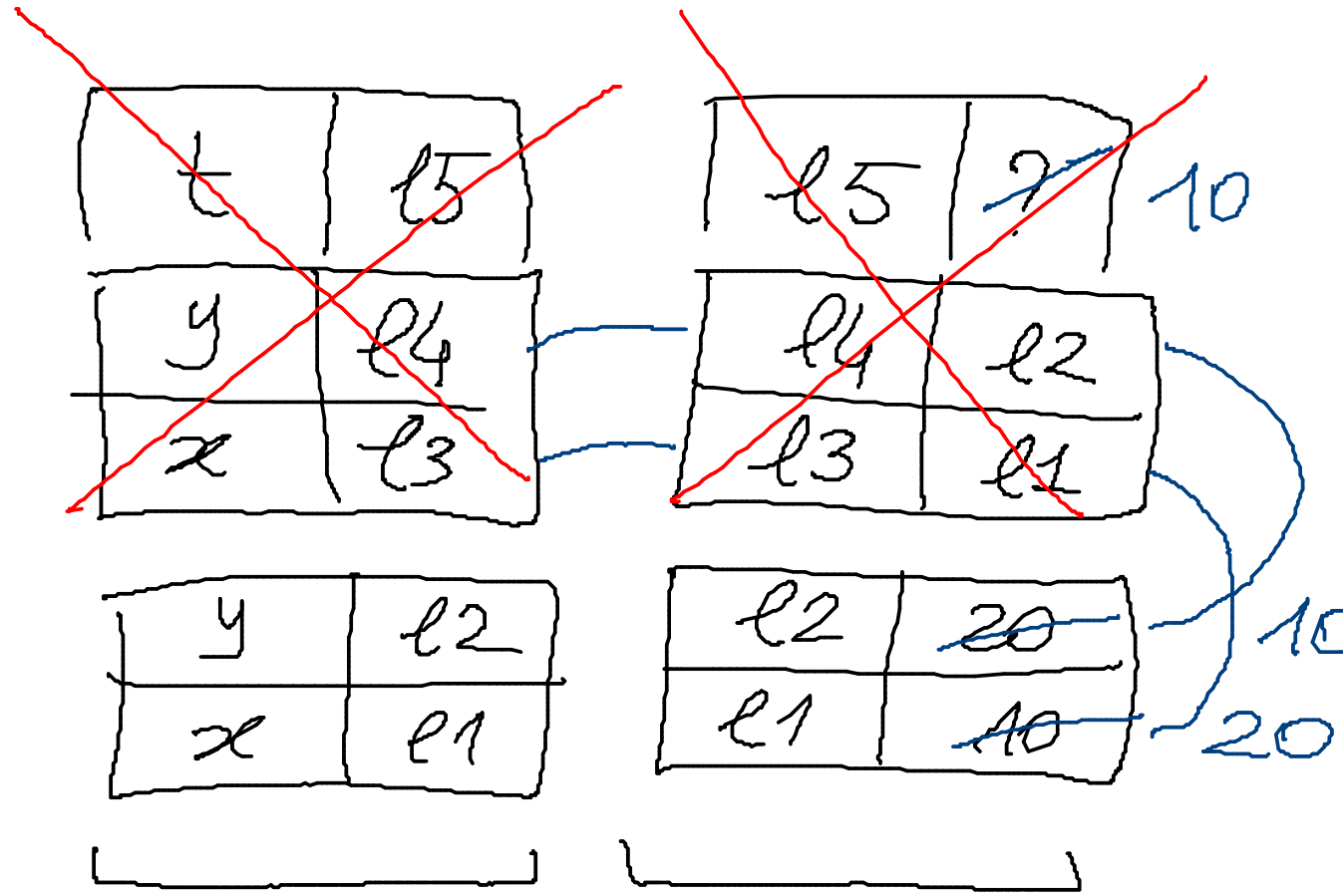






```
void swap (int *x, int *y)
    { int t;
      t = *x;
    *x = *y;
    *y =  t;
    }
```

```
{ int (x) = 10, int y = 20;
  swap (&x, &y);
```



SEMANTICA DI PROCEDURE CON 1 SOLO PARAMETRO FORMALE

```
void p (T x) { .... }
```

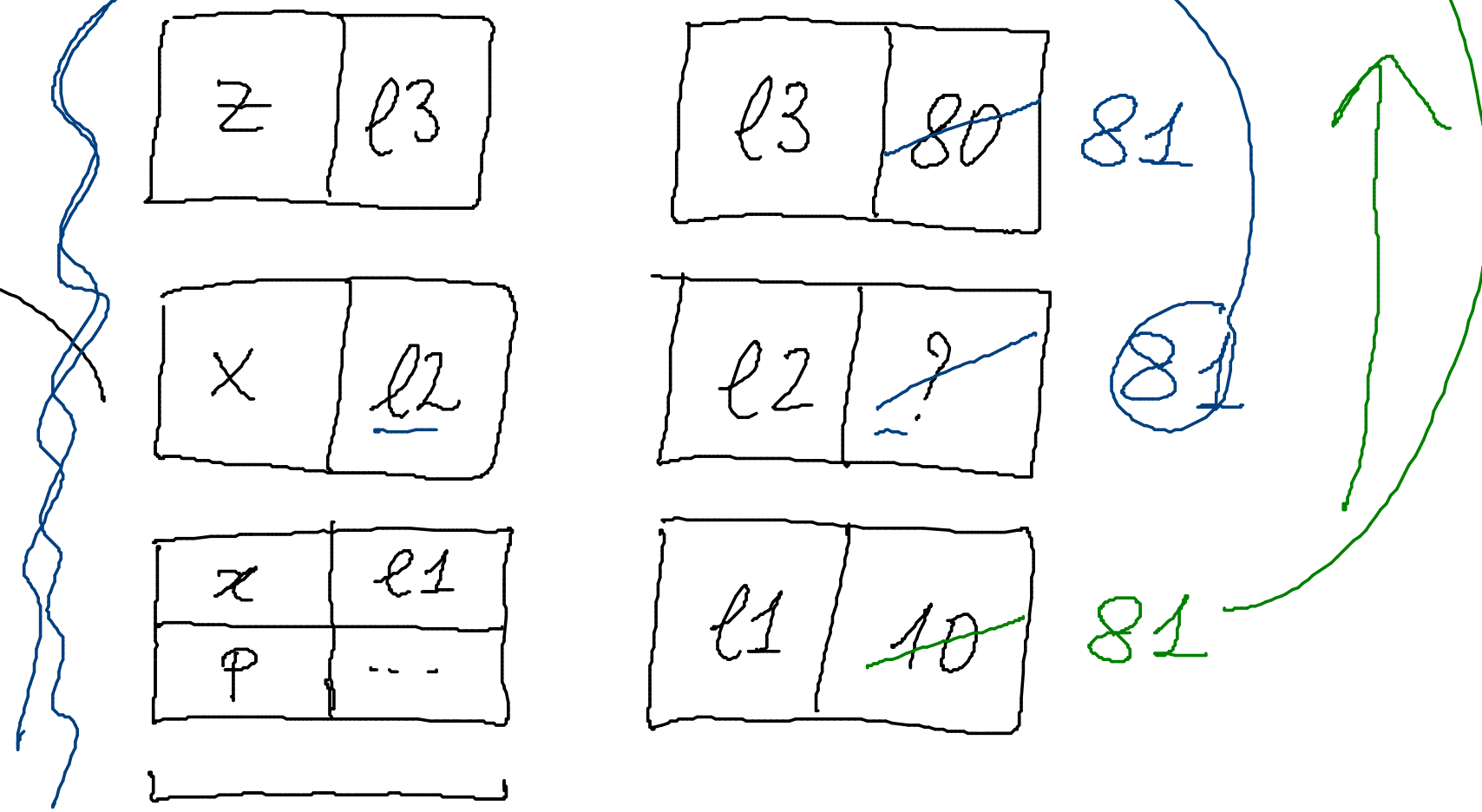
- C'è un "potenziale" problema

SCOPING STATICO

```
{ int x = 10;
  void p (int z)
  { z = z + 1; x = z; }
  ...
  { int x;
    p (80);
    ...
  }
  ...
}
```

variabile GLOBALE

SCOPING DINAMICO



SCOPING STATICO: le variabili globali utilizzate nel corpo della procedura sono quelle presenti nello stato al momento della dichiarazione della procedura

SCOPING DINAMICO: " " "

presenti nello stato al momento della chiamata

Dec ::= void Ide (Type Ide) Blck

Com ::= Ide (Exp);

- al nome della procedura sia associato un "oggetto semantico" che ci consente, al momento della chiamata di

1) associare al parametro formale il parametro attuale
2) operare nell'ambiente presente al momento della dichiarazione (SCOPING STATICO)
3) eseguire il blocco della procedura

DOMINIO SEMANTICO **PROC**

$$Proc = \{ f \mid f : Val \to \cancel{E} \to M \to M \}$$

NO

parametro attuale

ambiente (SCOPING STATICO)

memoria al momento della chiamata

$$Sem_c\ p\ (E);\ e\ \mu = \mu'$$

$$(e\ p) = psem \in PROC \qquad (psem : Val \to M \to M)$$

$$psem\ v\ \mu = \mu'$$

$$V = Sem_{exp}\ E\ e\ \mu$$

come si definisce psem al momento della dichiarazione di p ??

$$\rho : \mathrm{Ide} \longrightarrow \mathrm{Loc} \cup \mathrm{Proc}$$

$$\mathrm{Sem}_d\,[\,\mathrm{void}\ p\,(\mathrm{T}\,x)\ B;\,]\ \rho\ \mu = \rho\left[\begin{smallmatrix} psem \\ p \end{smallmatrix}\right]^{odd},\ \mu$$

argomenti di psem

dove $psem\ \boxed{a\ \mu'} = \mu''$

$$\mathrm{Sem}_c\ B\ \ \omega\left[\begin{smallmatrix} l \\ x \end{smallmatrix}\right]^{odd} \cdot \rho\ \ \omega\left[\begin{smallmatrix} a \\ l \end{smallmatrix}\right]^{odd} \cdot \mu' = \nu \cdot \mu''$$

$$l = \mathrm{succloc}\ \mu'$$