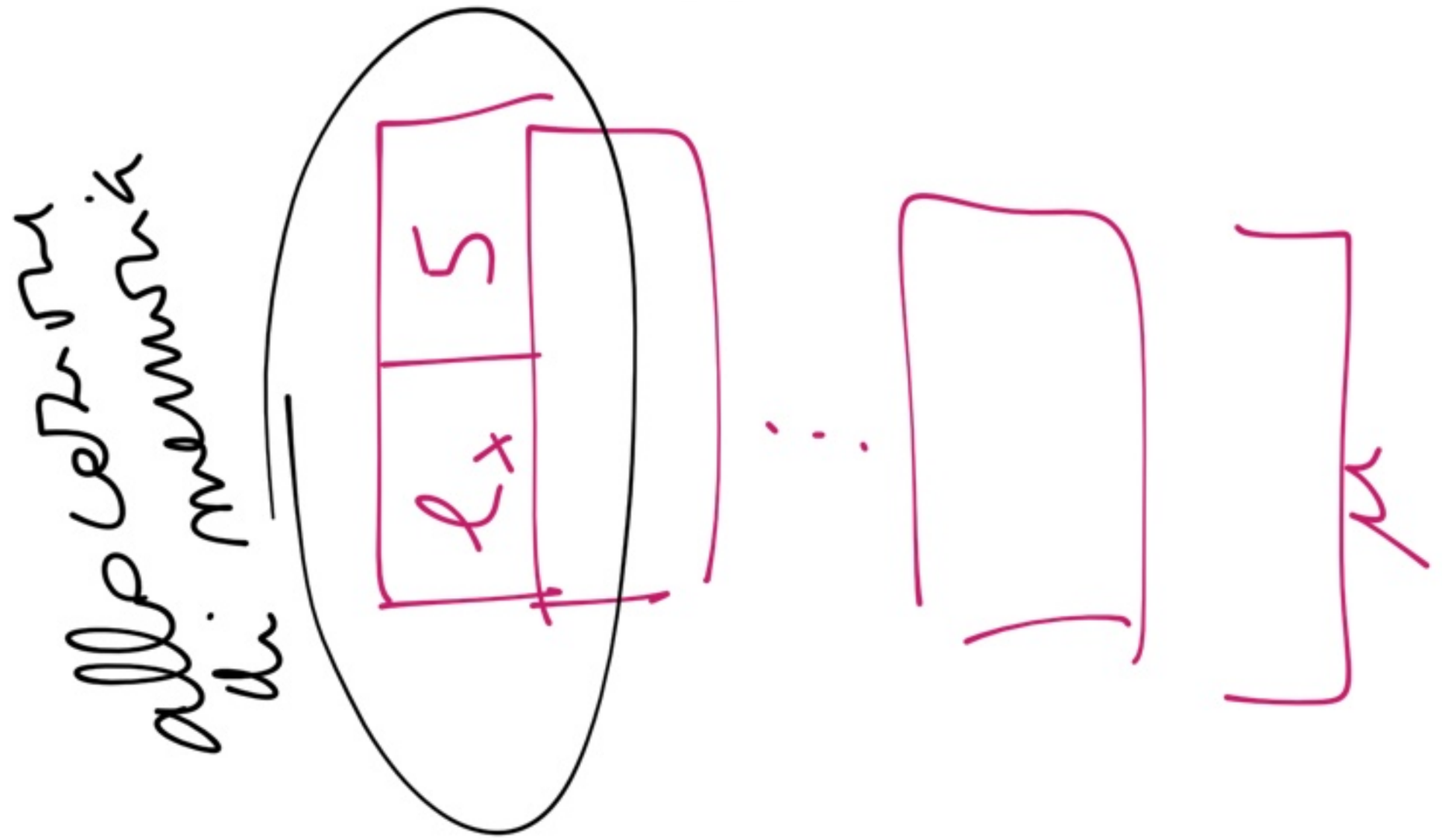
memoria dinamica
(memoria heap)

---

la allocazione (riservare parole di memoria) viene fatta durante la dichiarazione

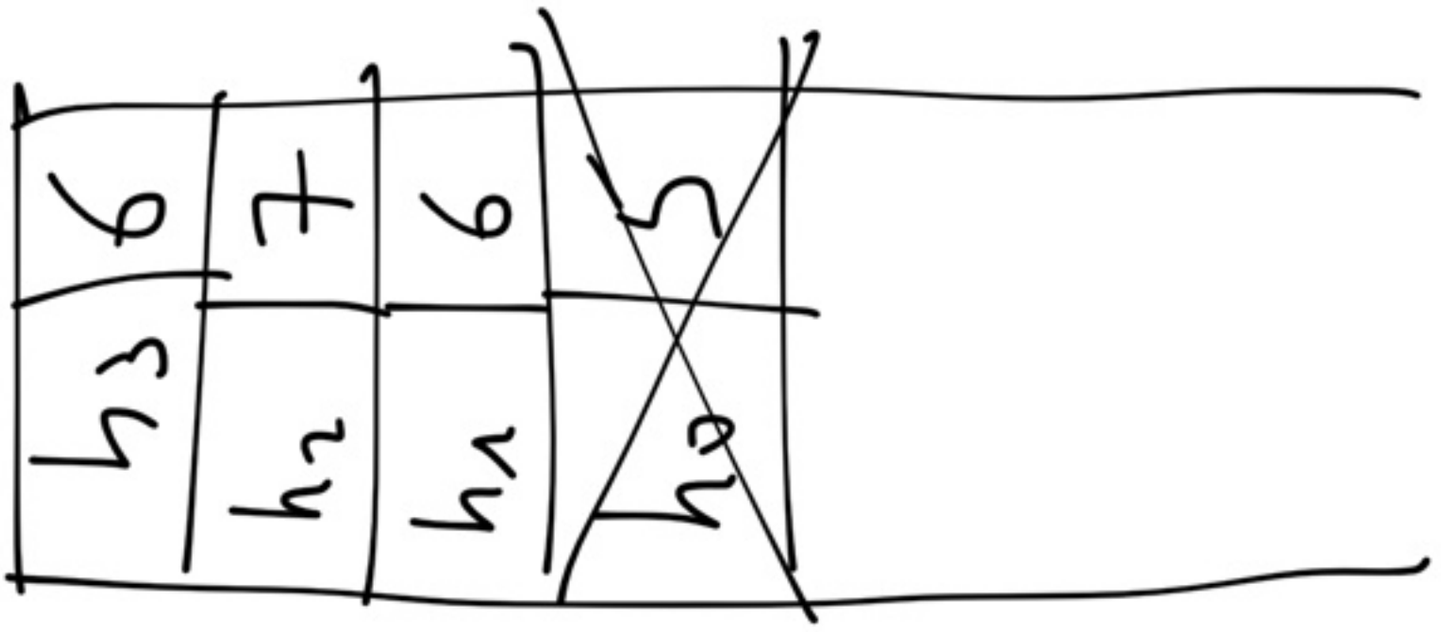
allo stesso modo

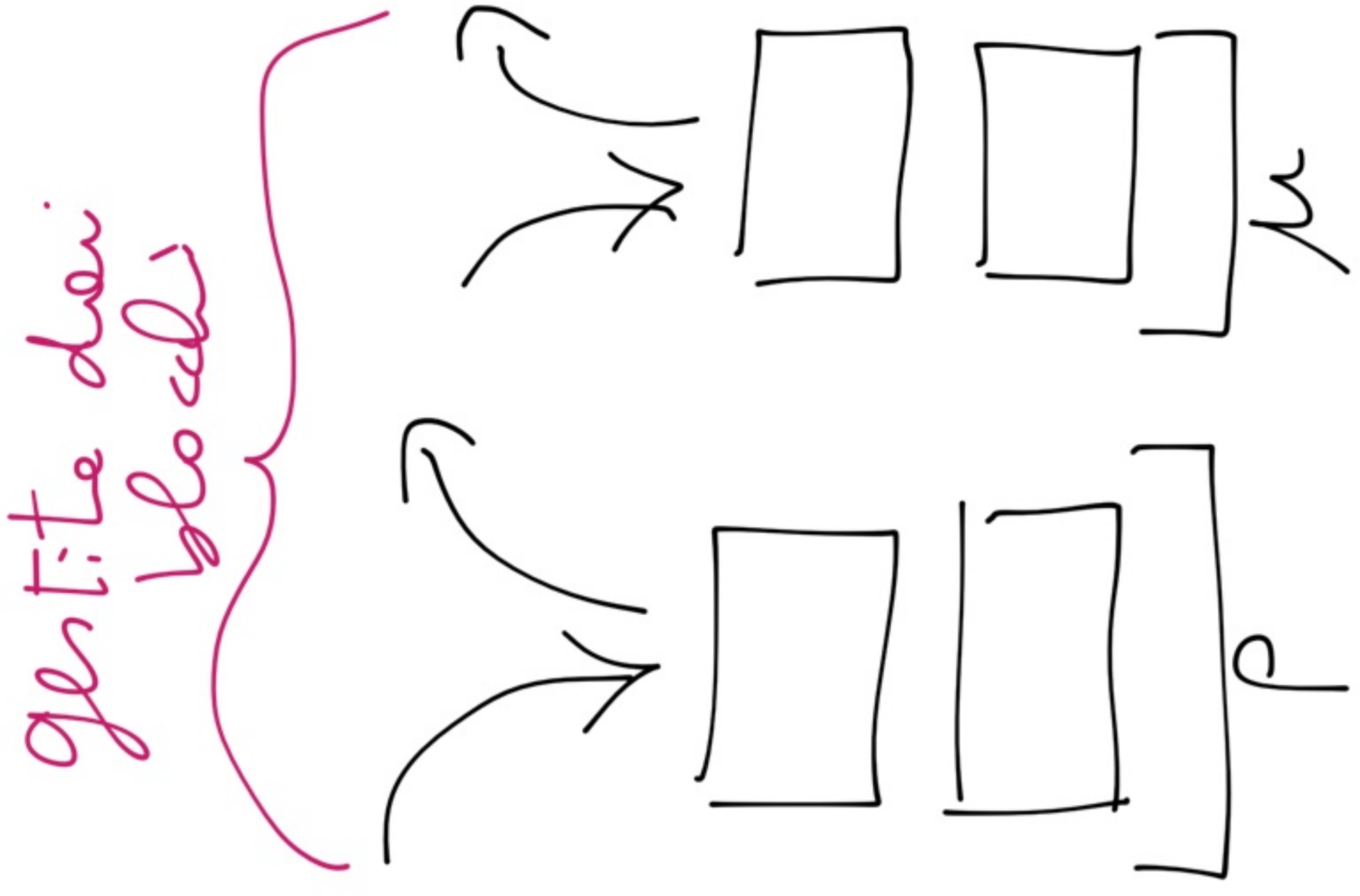


$\Rightarrow$ tot $x = 5$

- la memoria dinamica può anche allocate anche durante l'esecuzione di comandi.

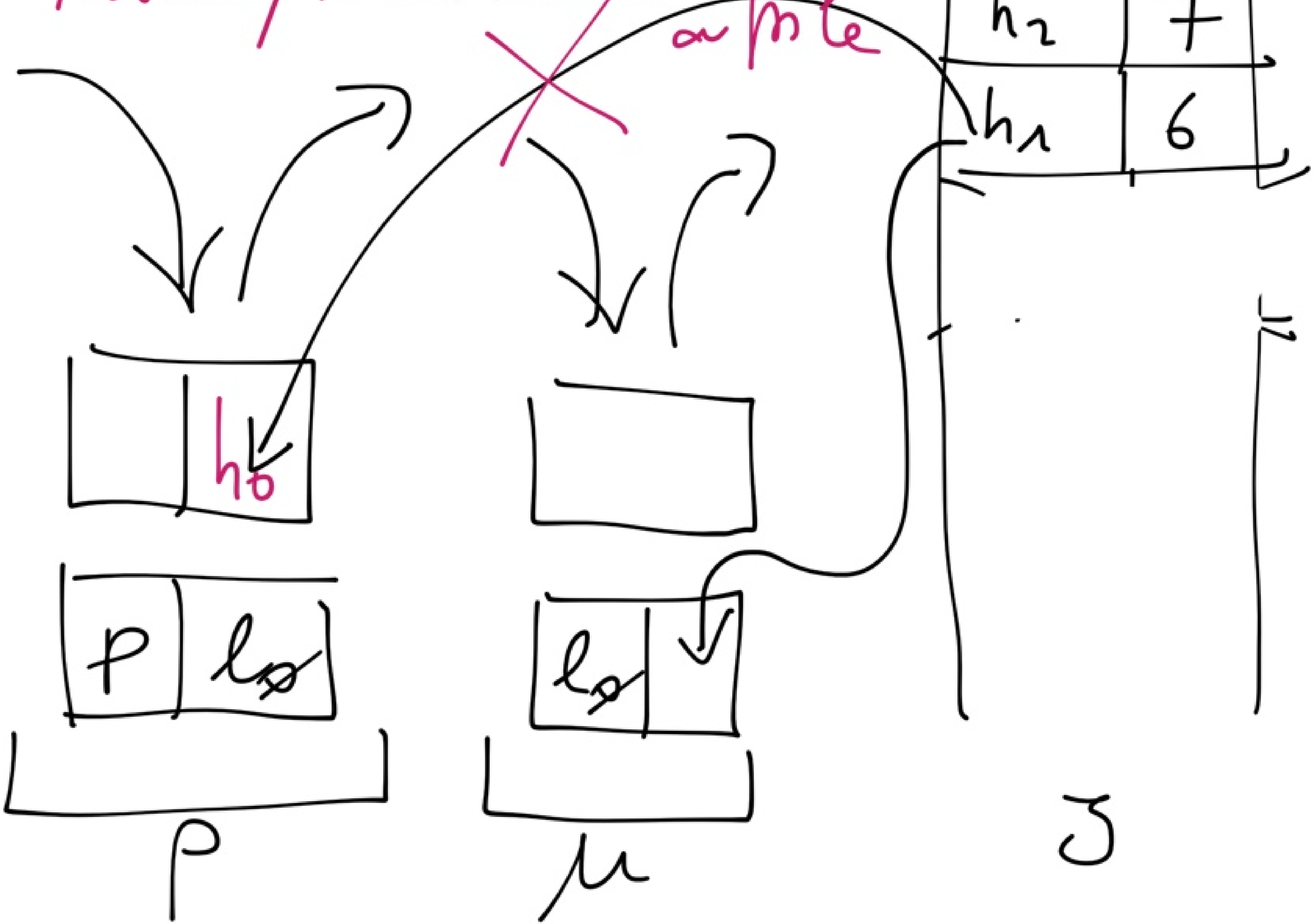
- la memoria dinamica deve anche deallocata esplicitamente (non viene gestita dai blocchi)


6
7
6
5
3
2
1
0


gentile dei bloccki
n
p

nell'ambiente continuiamo ad avere associazioni:

nome, ind. della mem ... ambile



2015_12_05

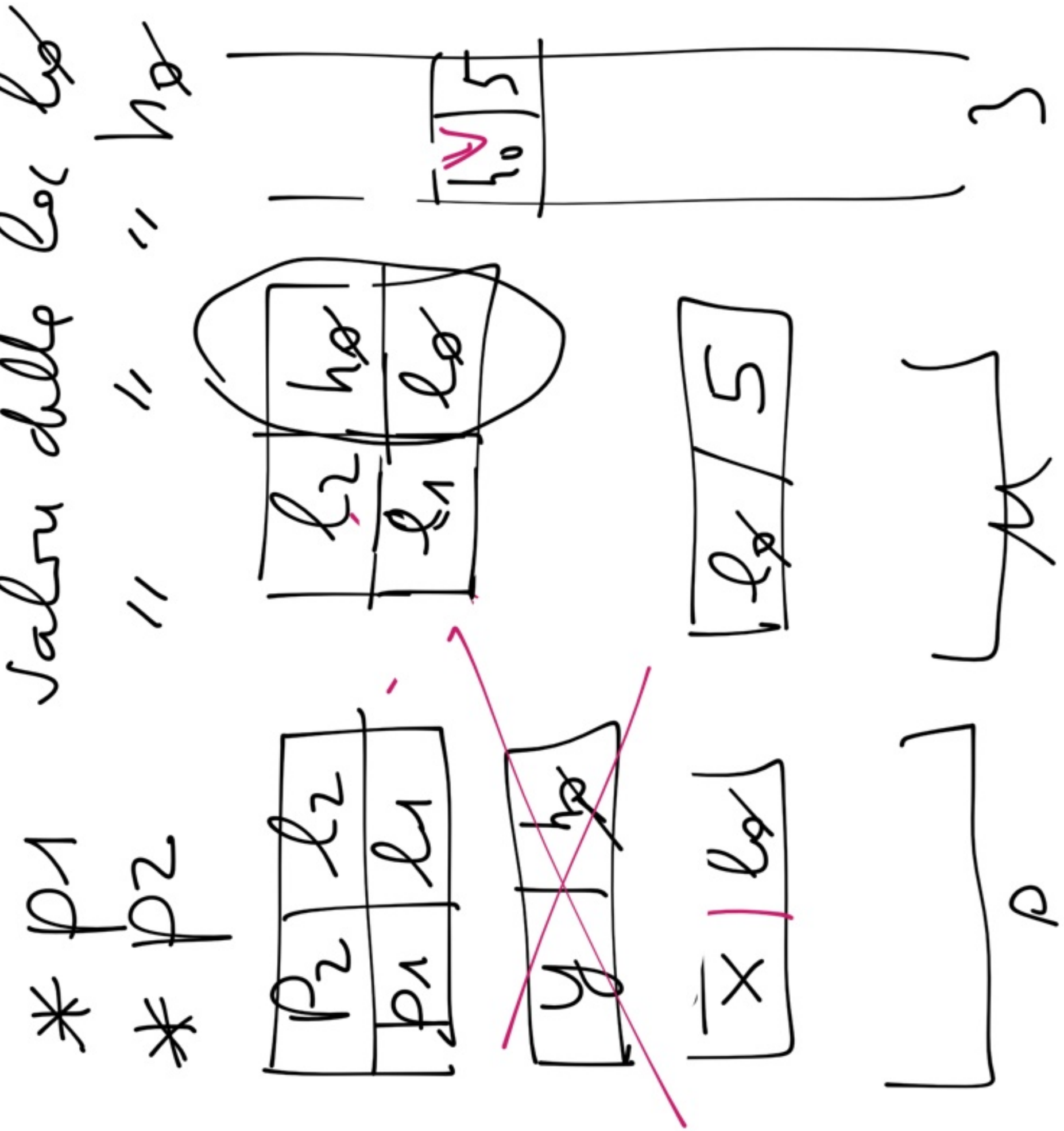
le variabili puntatore possono avere come valori:

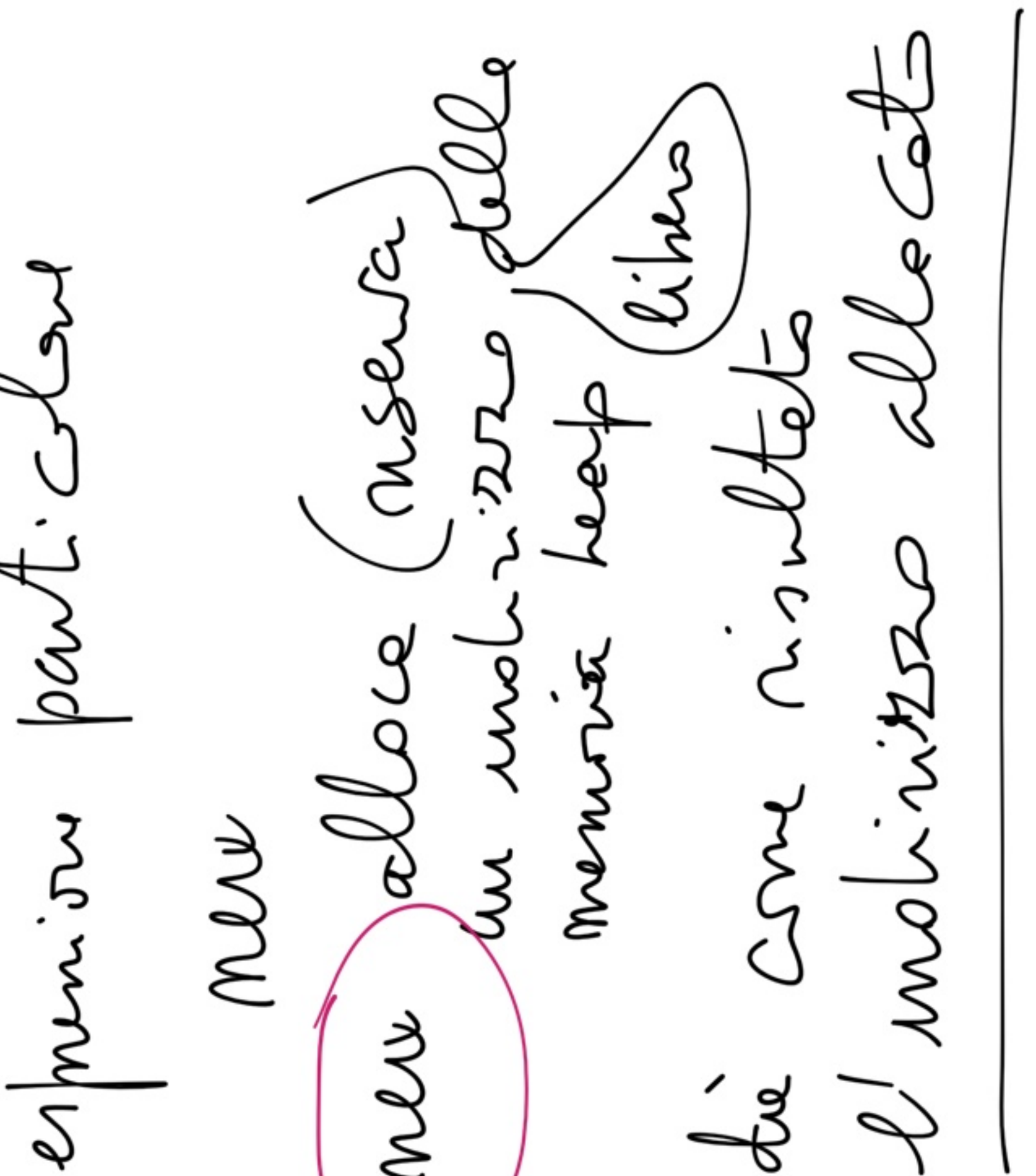
- indirizzi della memoria a pila
- indirizzi della memoria heap

Non posso avere un p
associazioni
(nome, ind. mem. heap)

⇒

il valore delle variabili
può stare solo nella
memoria a pila.

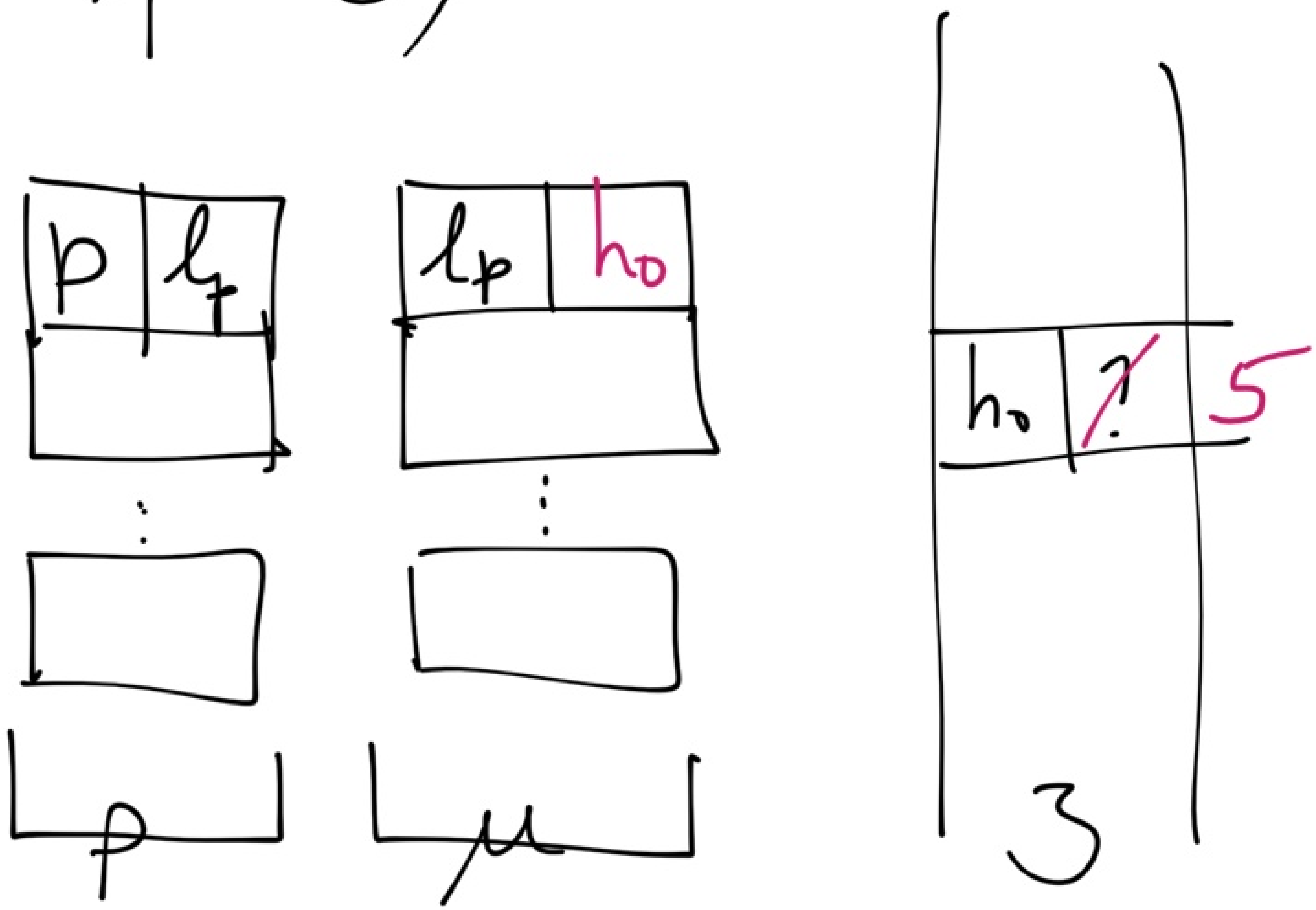




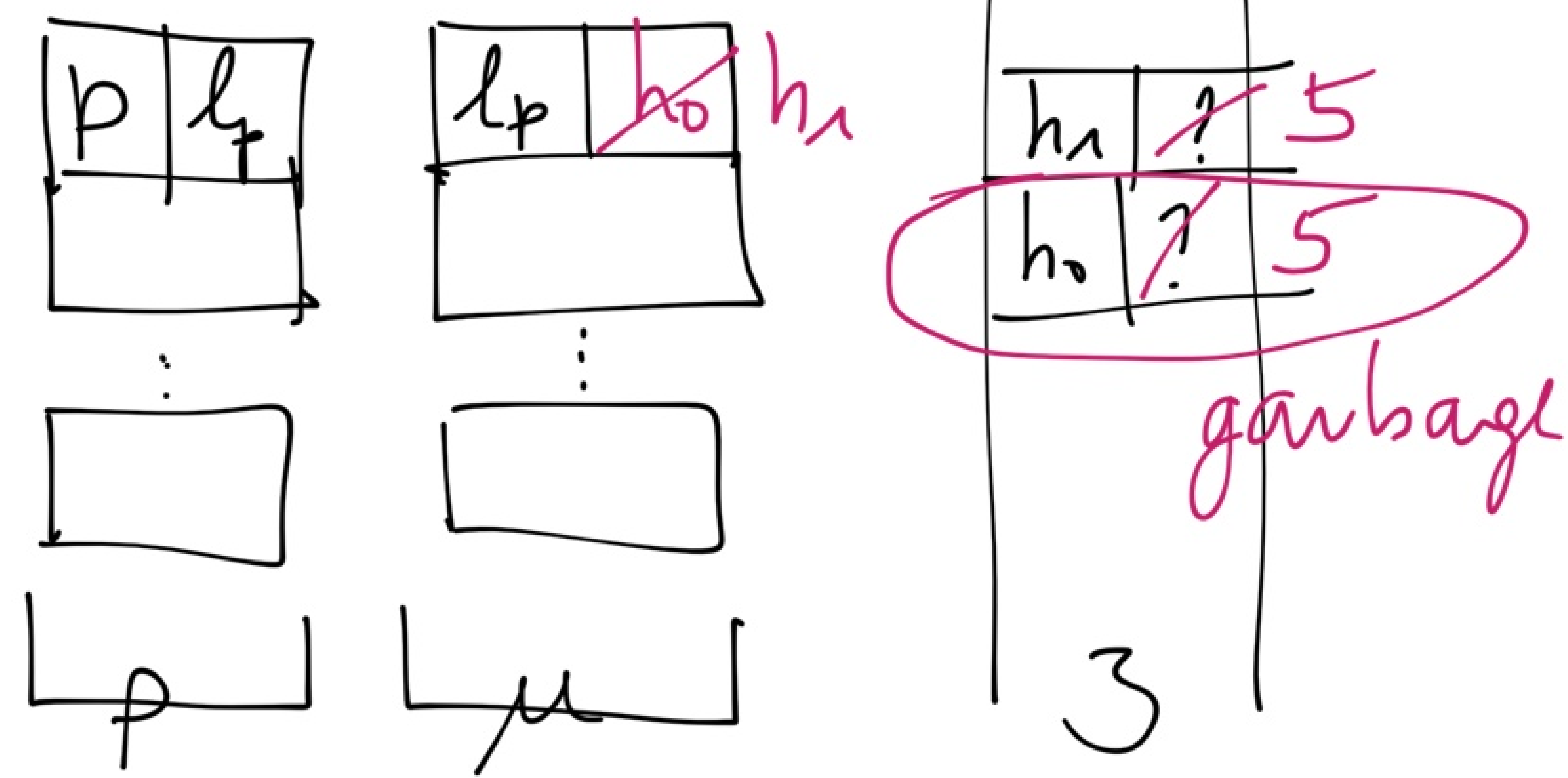
```
{ int * p = new ;
```

h_0

```
*p = 5;
```



```
{ int * p = new;      (h0)
  *p = 5;  p = new;  *p = 5;      (h1)
```

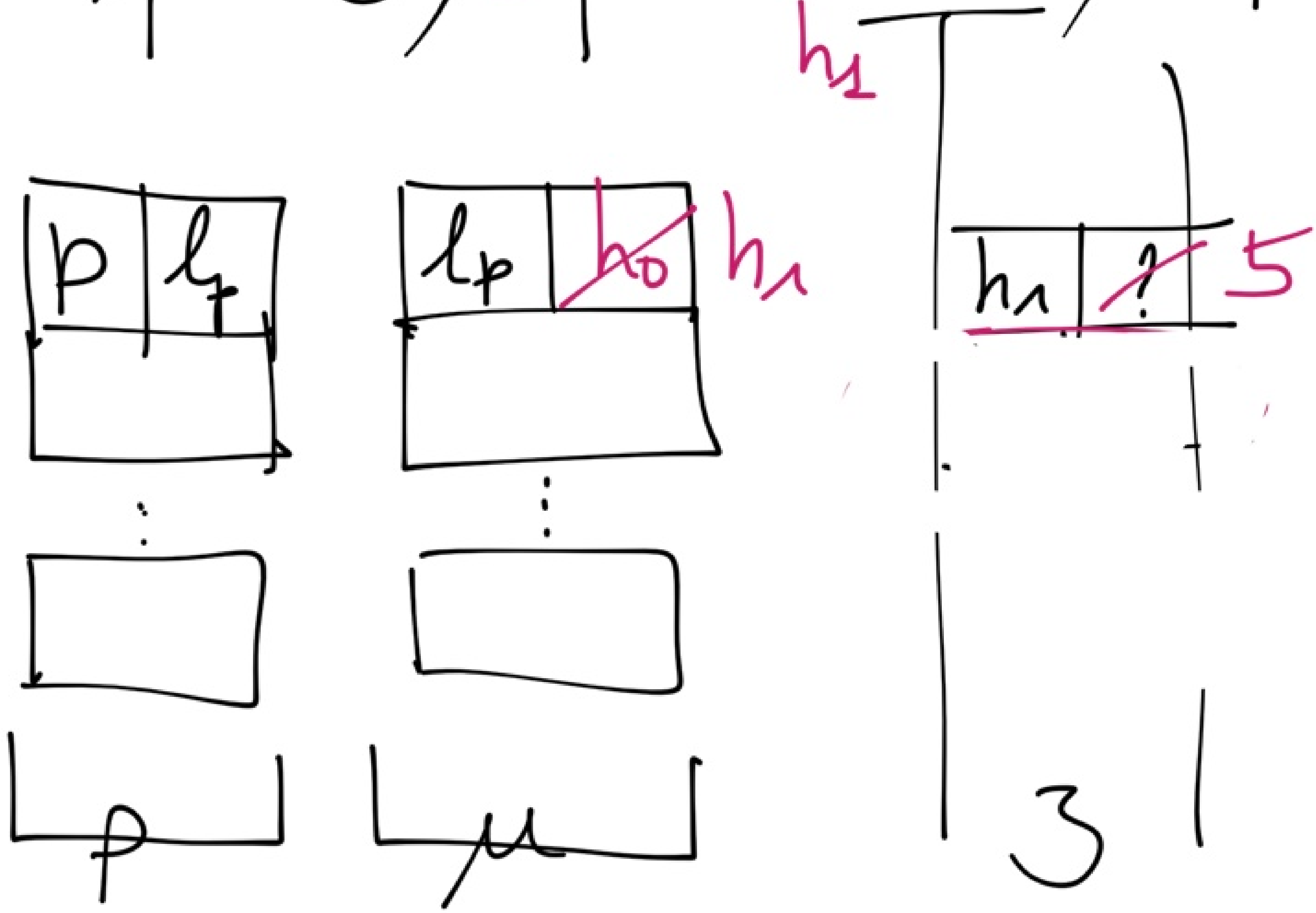


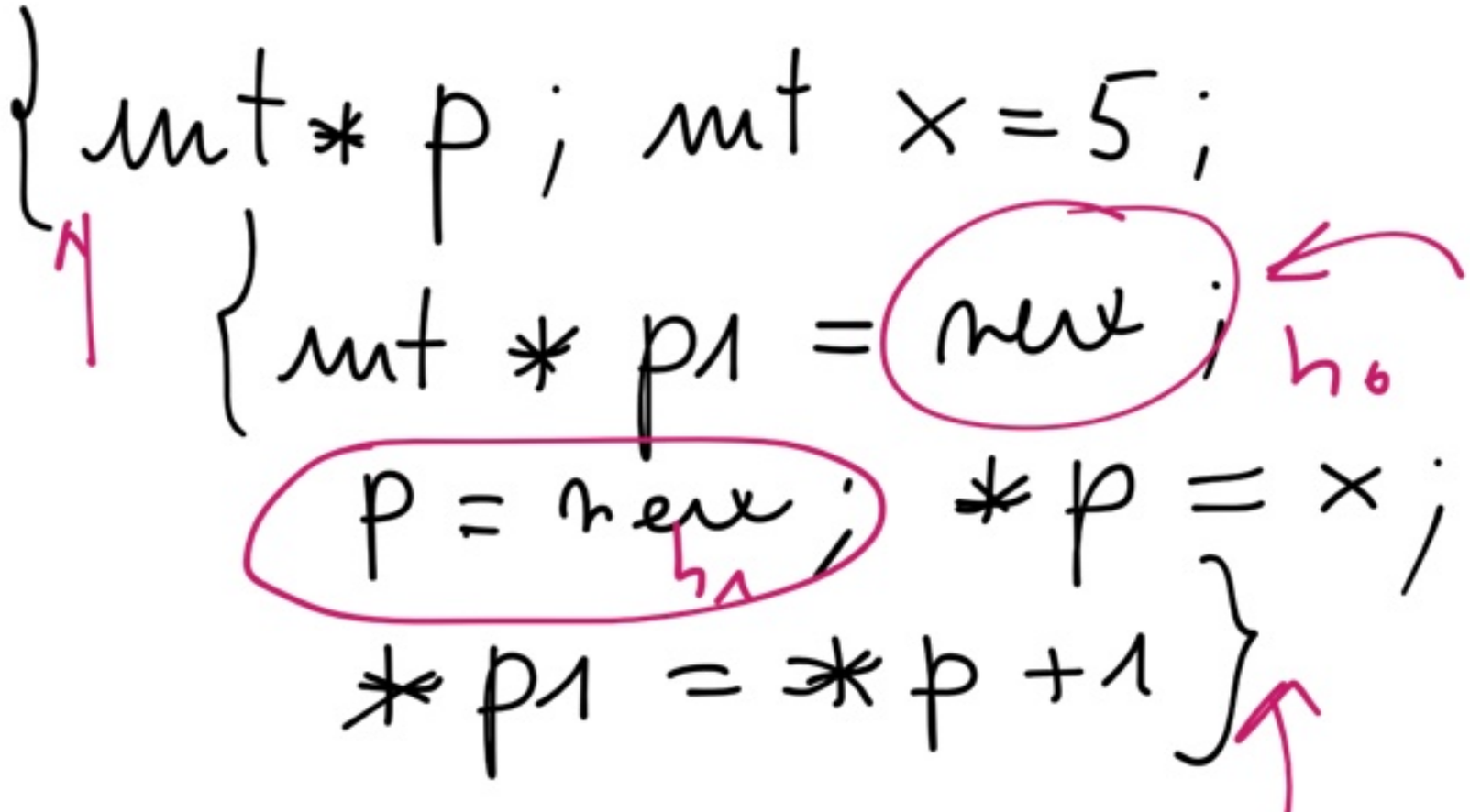
{ int * p = new ; free(p);

h_0

*p = 5; p = new; *p=5;

h_1

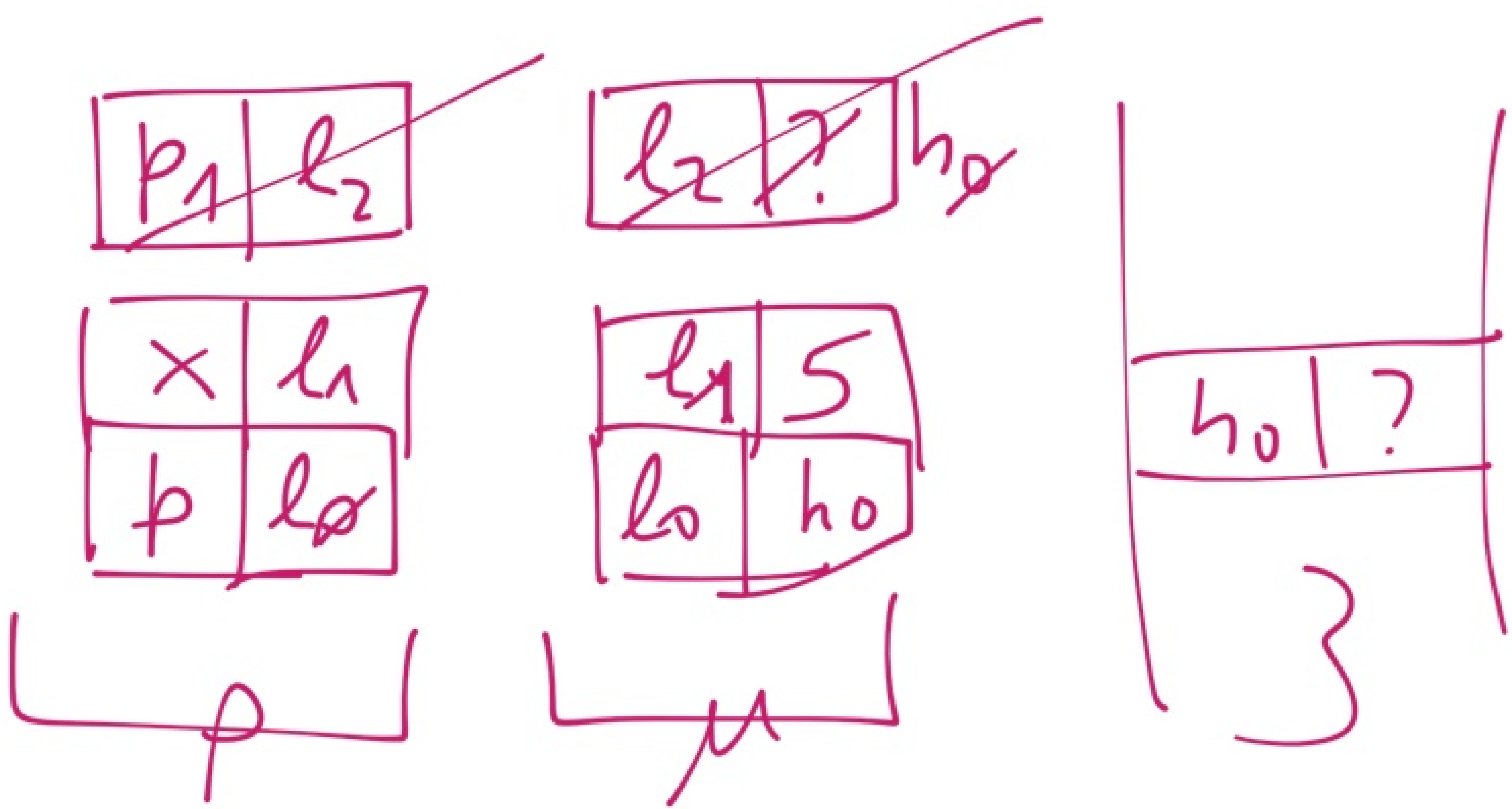




{ int *p = new, int x = 5,

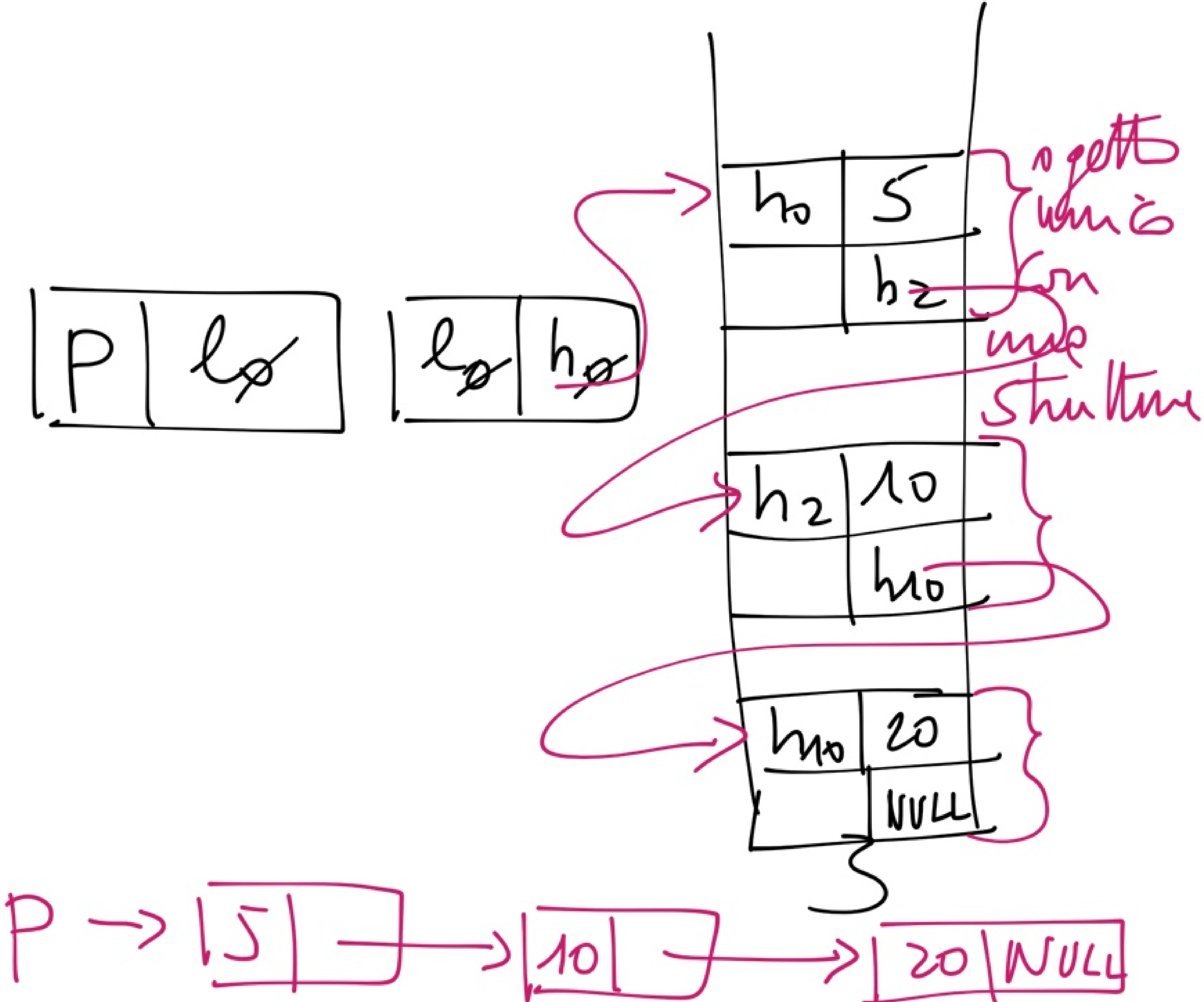
{ int *p1 ;

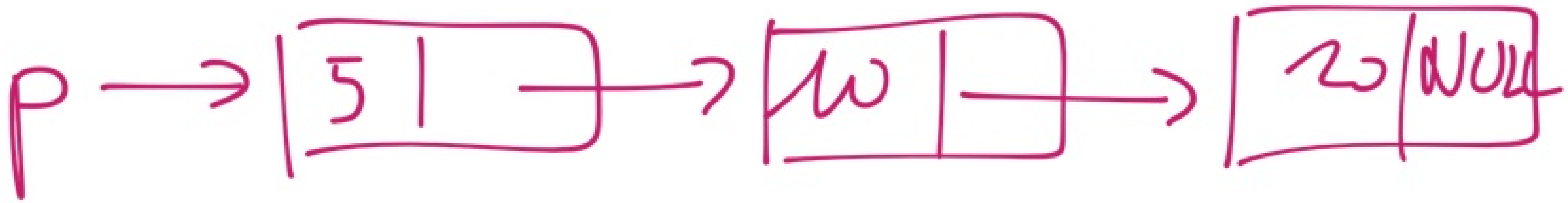
p1 = p ; }



- memoria dinamica
- si alloca e dealloca esplicitamente (new, free)
- non è regolata dei blocchi.

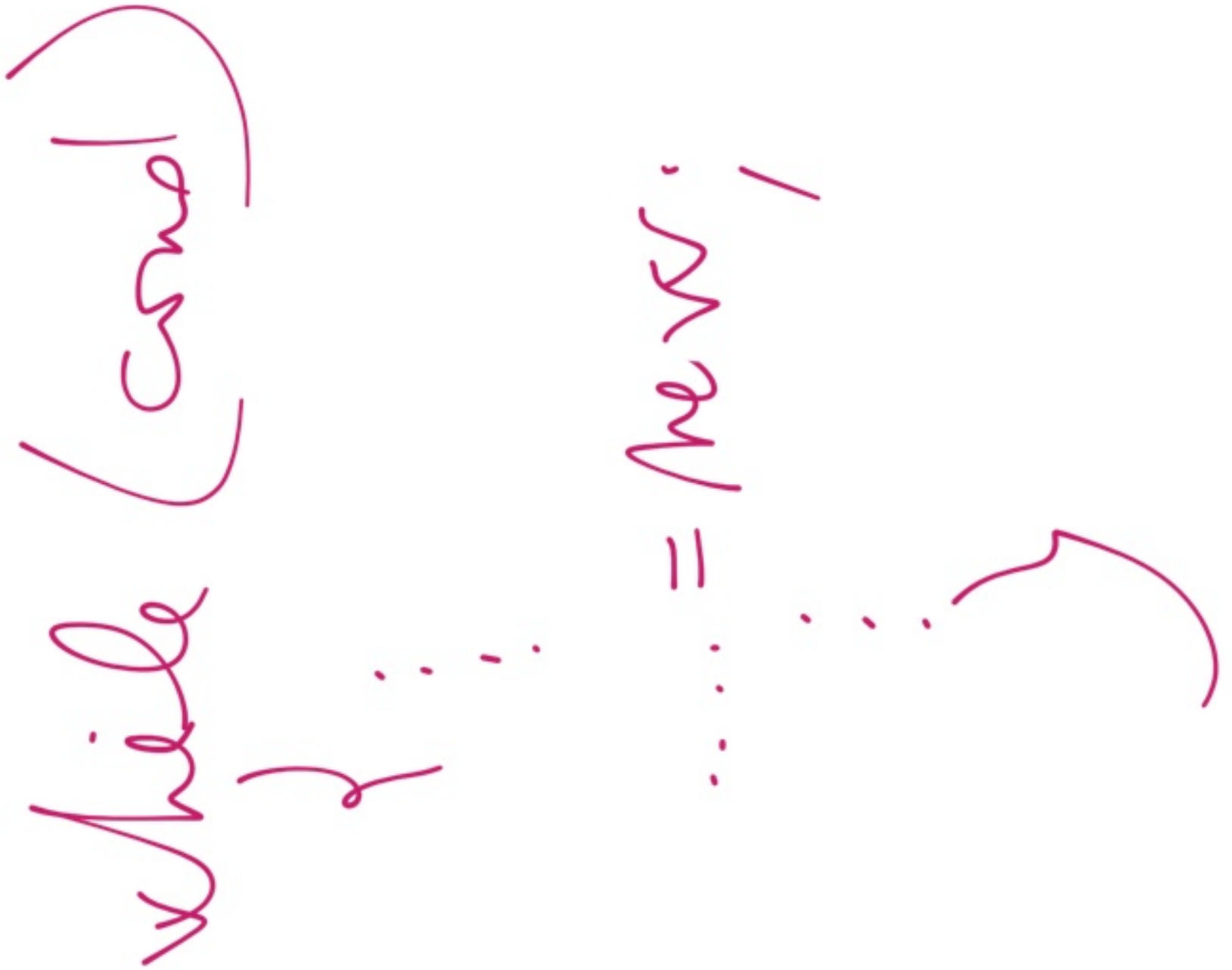
Come si fa a svincolarci dalle variabili?





Strutture dinamiche

possono modificare la lunghezza durante l'esecuzione dei comandi.



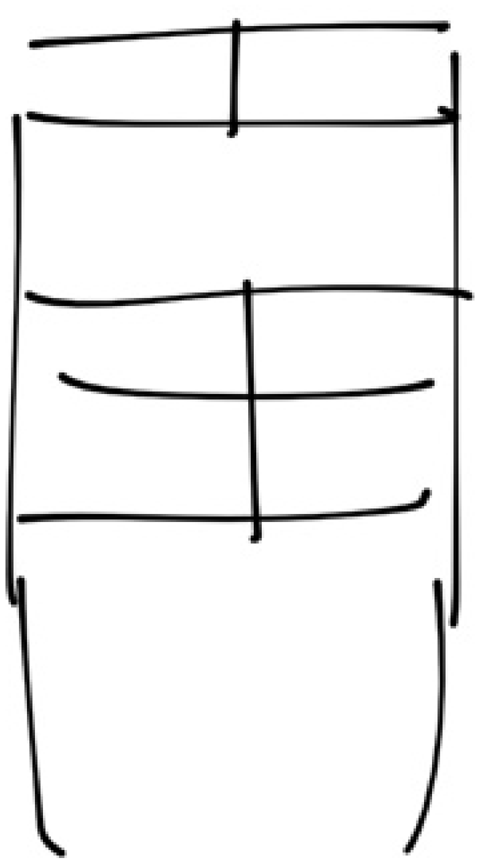
linguaggio C

new →

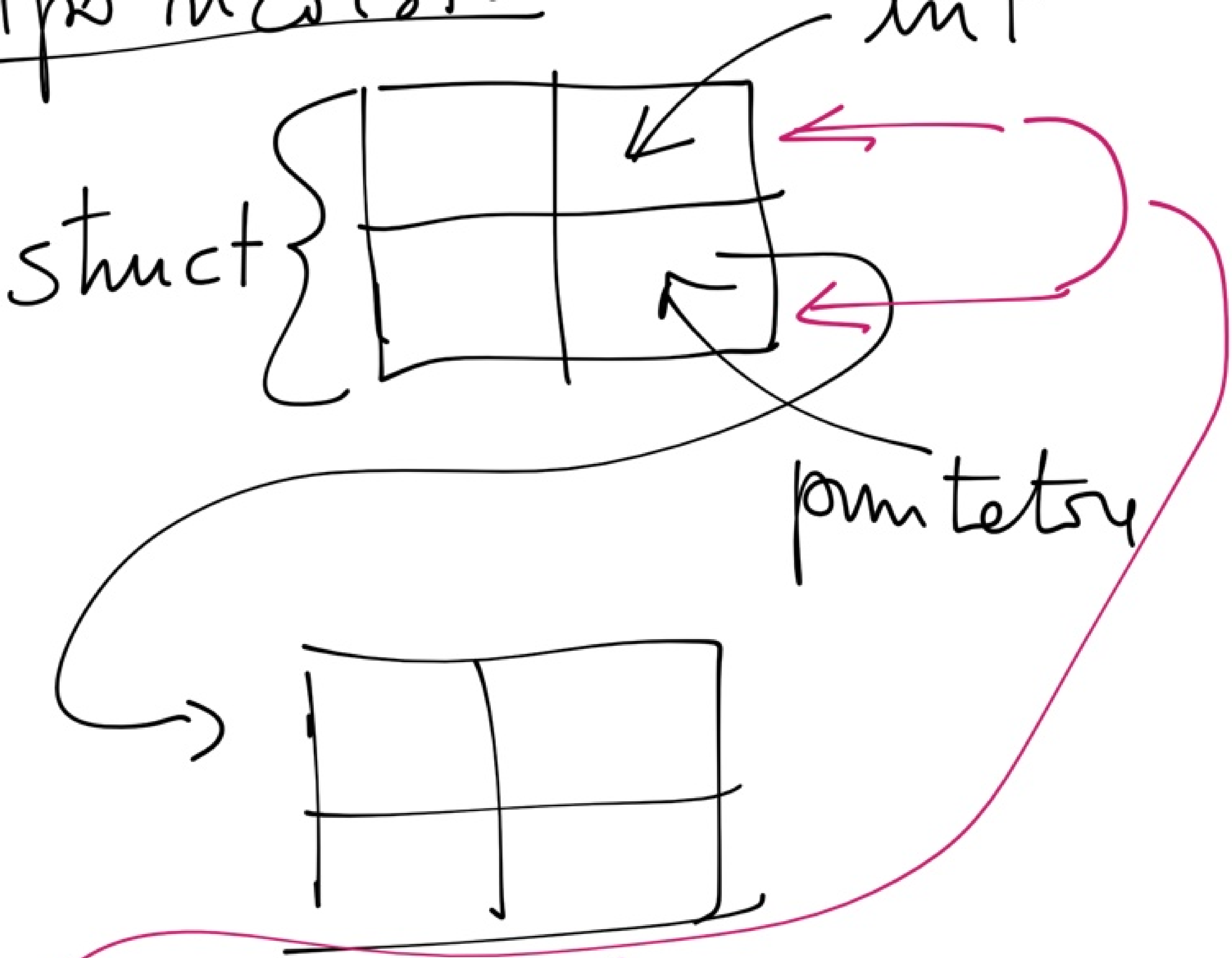
Tipo

malloc(sizeof(Tipo))

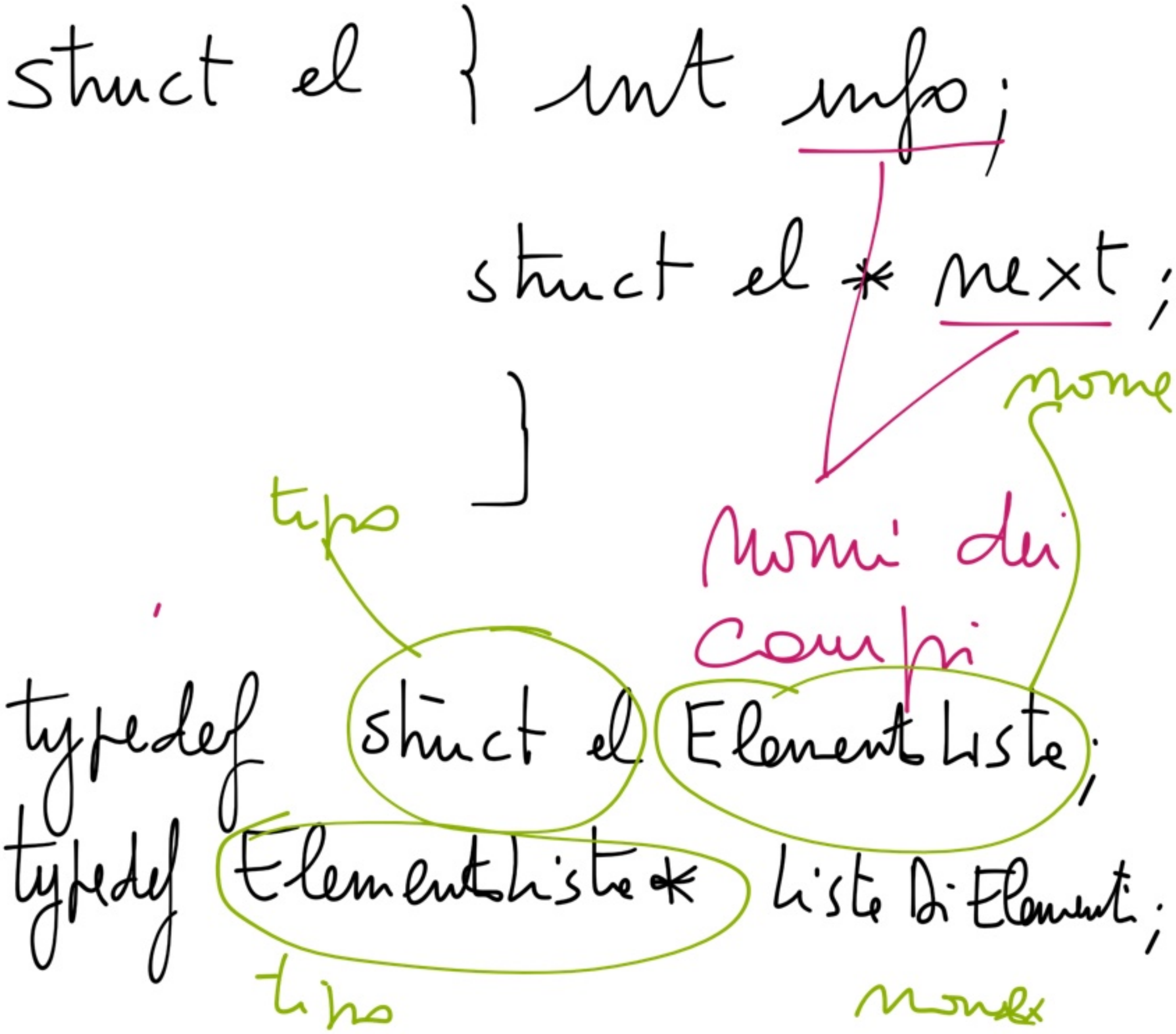
argomento lunghezza della struttura da allocare

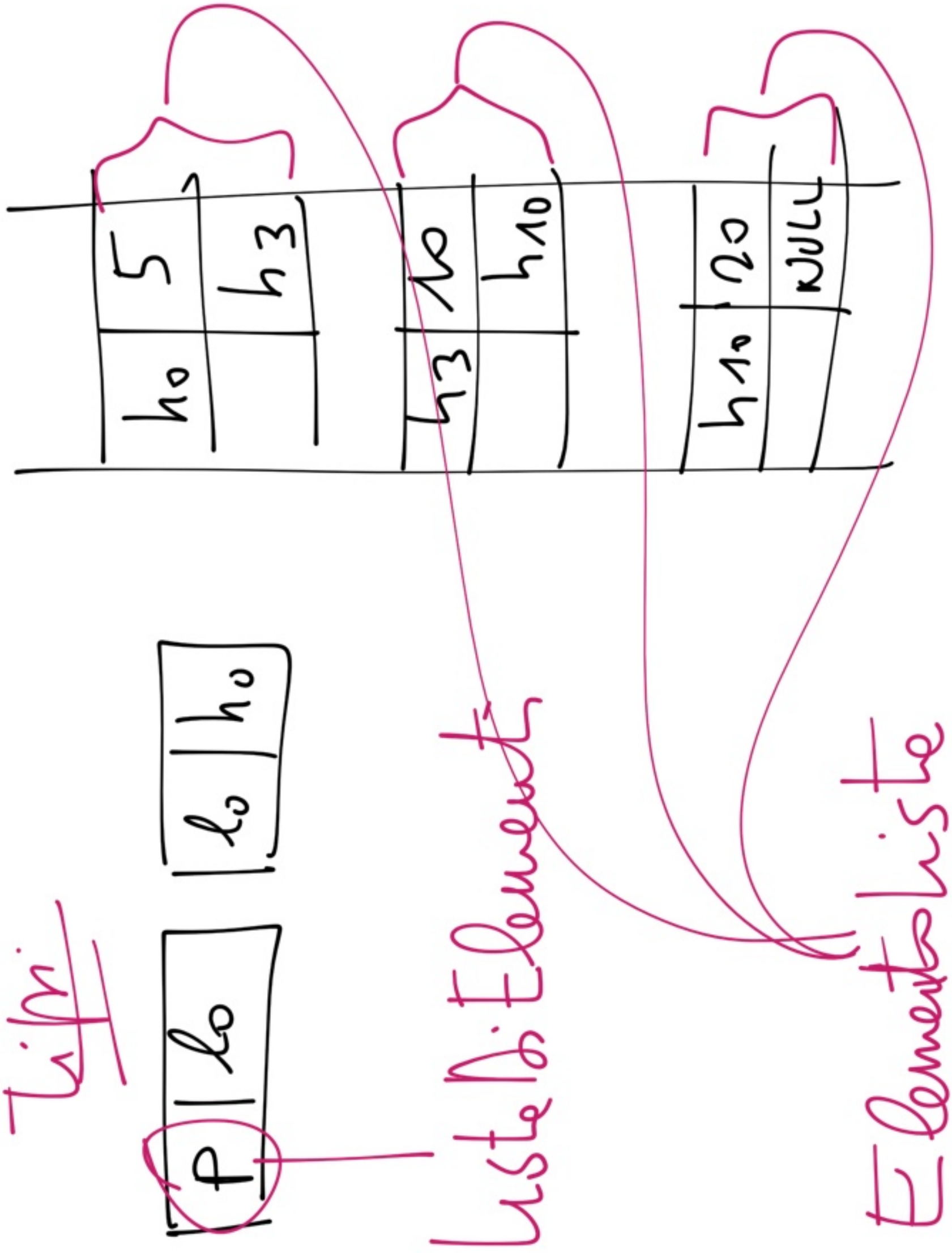


Tipo ricorsivo



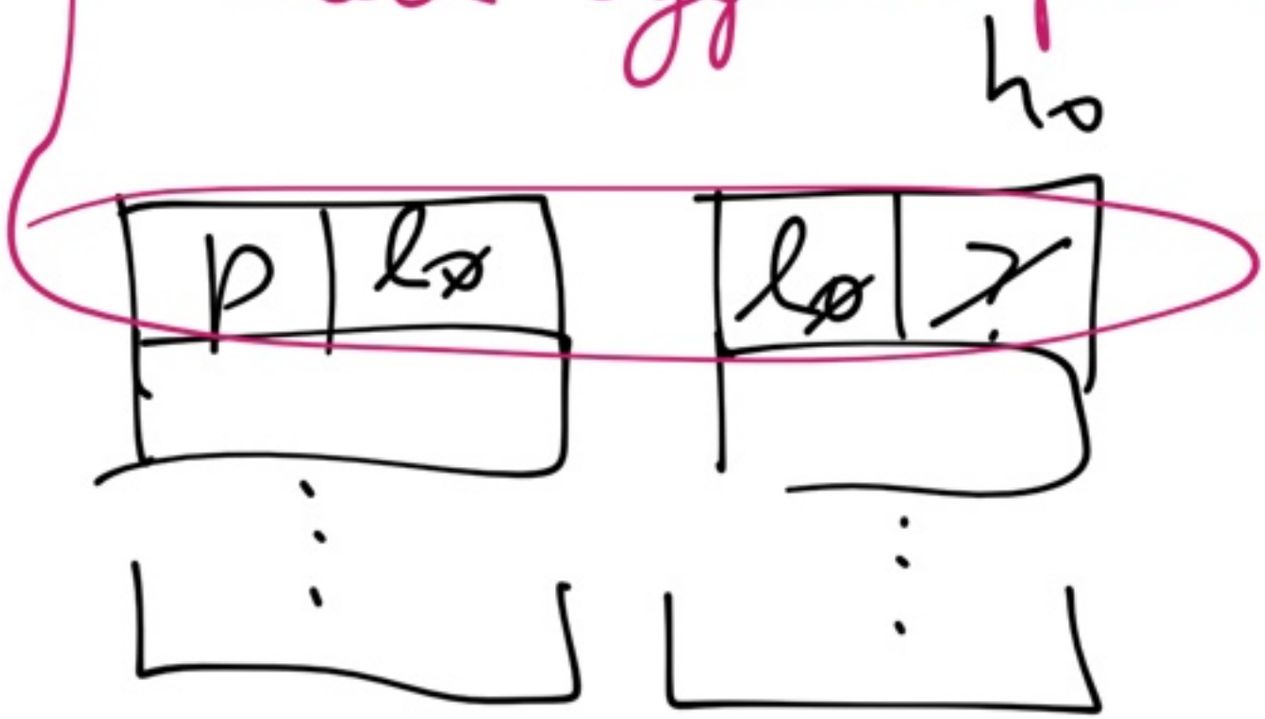
i campi hanno un nome

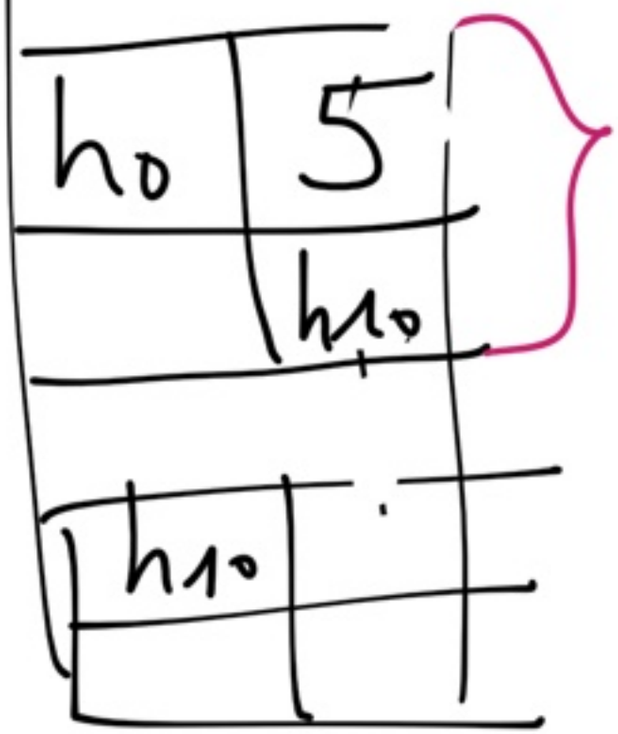




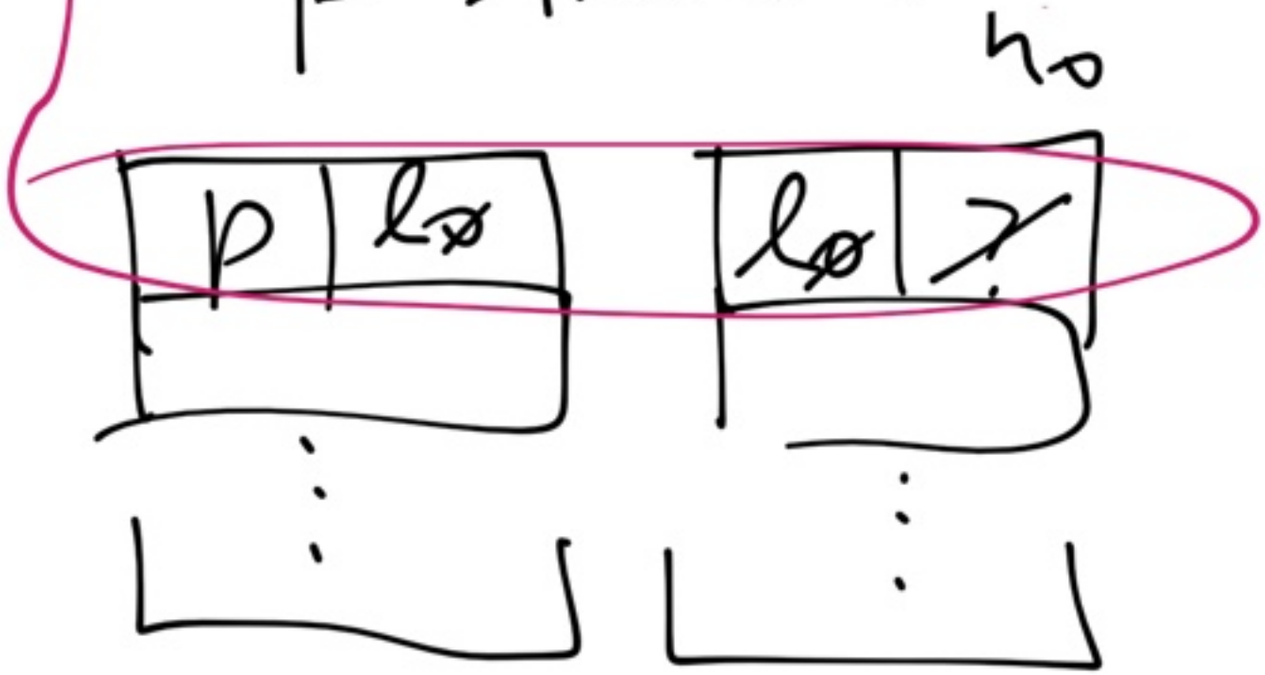
```
{ ListeDiElementi p;
  p = malloc(sizeof(ElementoLista));
  p->info = 5; p->next = malloc(sizeof(ElementoLista));
```

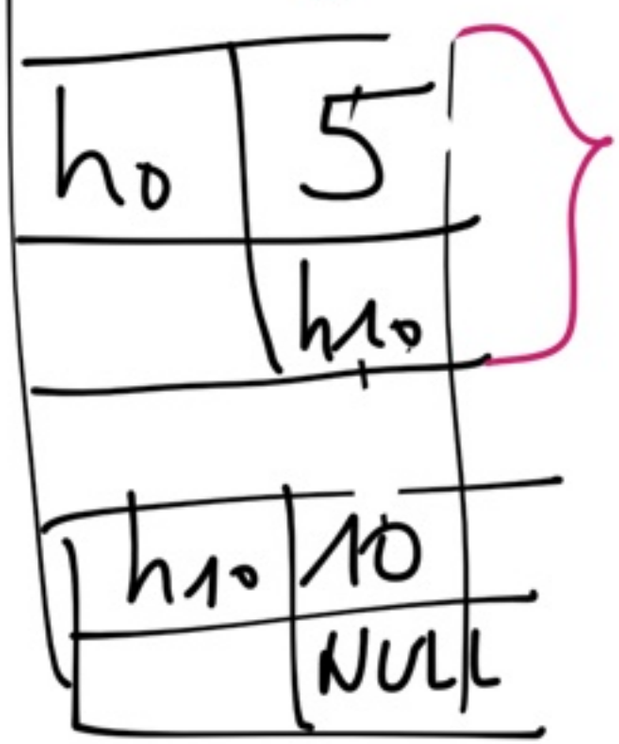
il campo di nome info dell'oggetto puntato da p





```
{ ListeDiElement p;
  p = malloc(sizeof(ElementoLista));
  p->info = 5; p->next = malloc(sizeof(ElementoLista));
  p->next->info = 10;
  p->next->next = NULL;
```



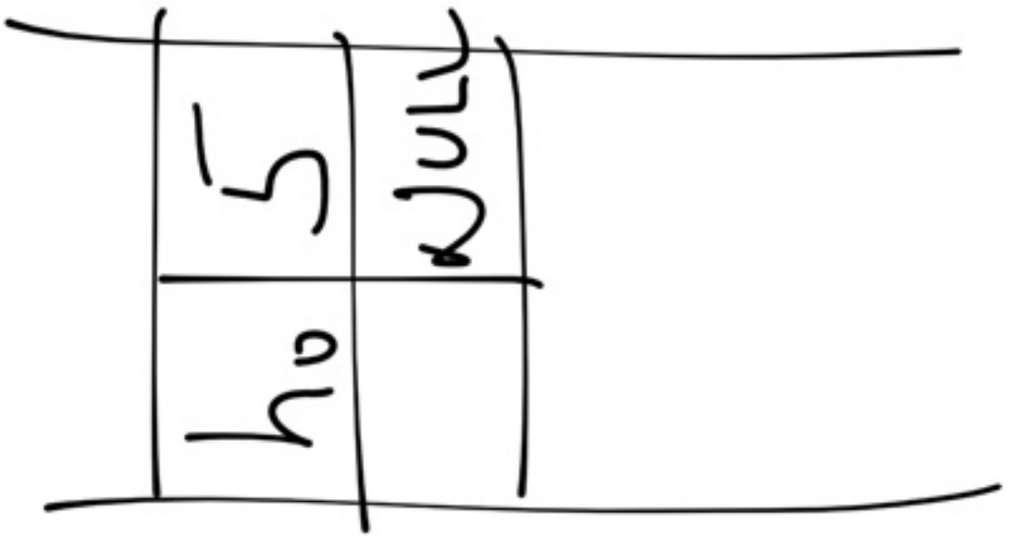


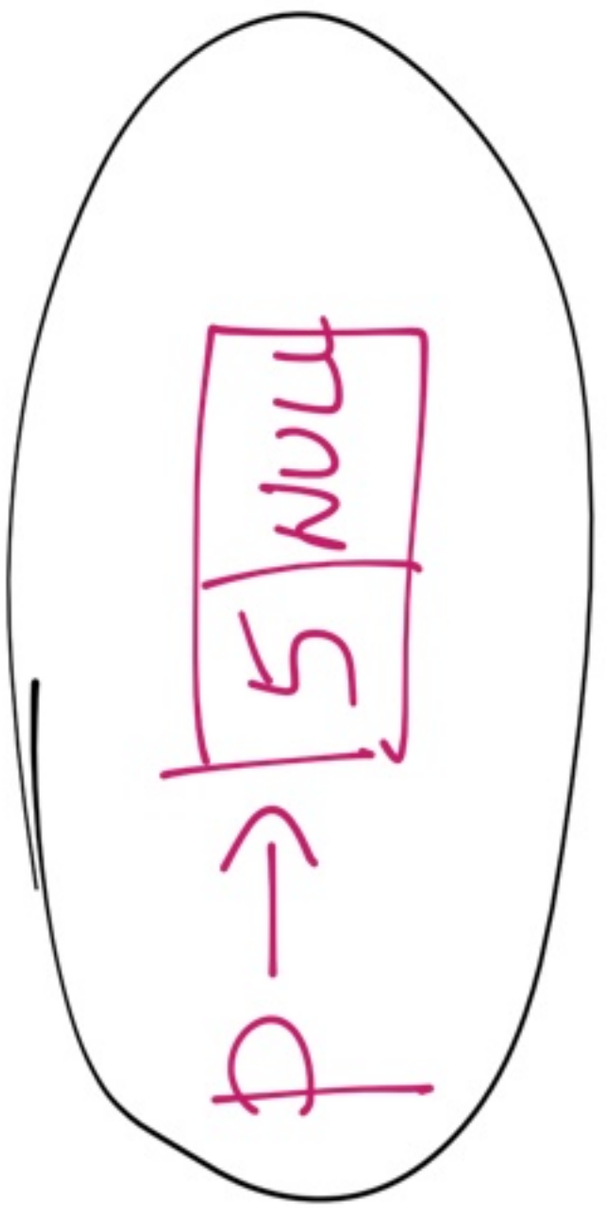
Supponendo una lista non vuota aggiungere un elemento

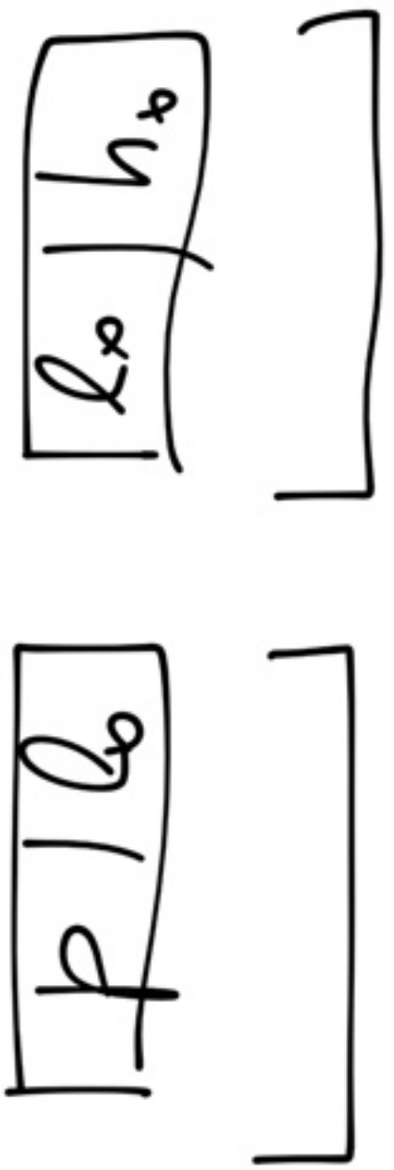
Void aggiungi (Lista* L, Elemento e)

p = malloc(sizeof(ElemLista));

p->info = 5;
p->next = NULL;







Lista di Elementi

```
void aggiungi(ListaDiElementi l, int el)
{
  ListaDiElementi aus = l;
  while(aus->next != NULL)
    aus = aus->next;
  aus->next = malloc(...);
  aus->next->inf = el;
  aus->next->next = NULL;
}
```