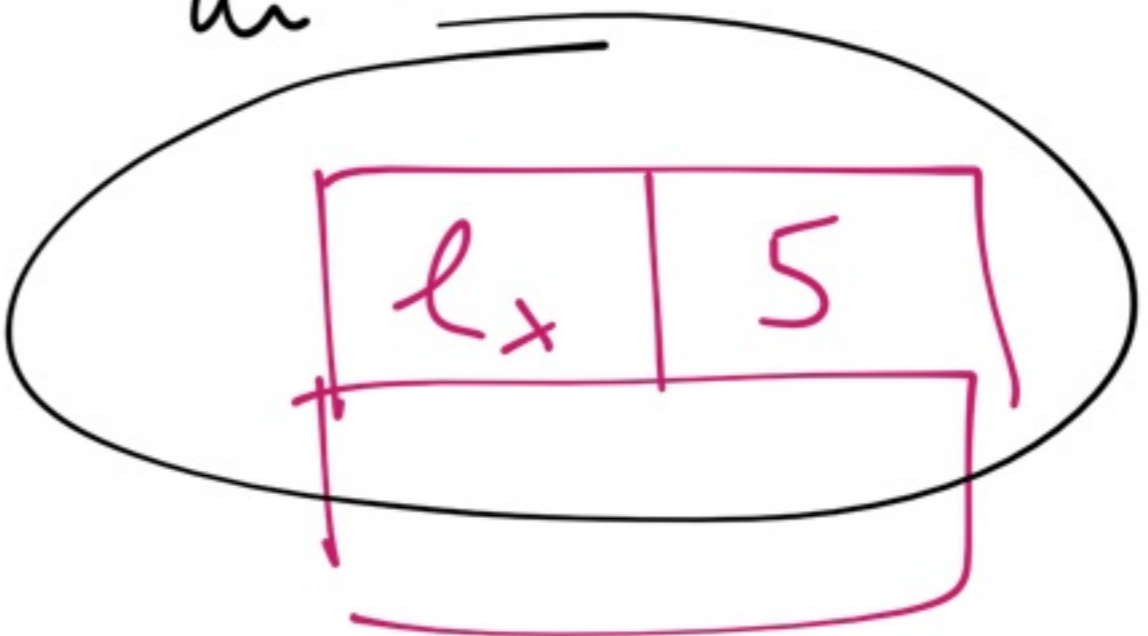


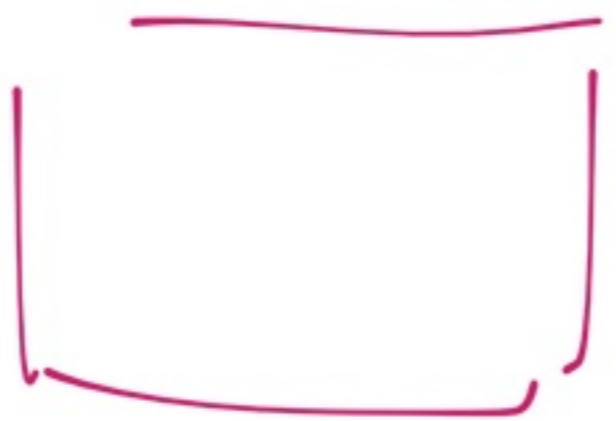
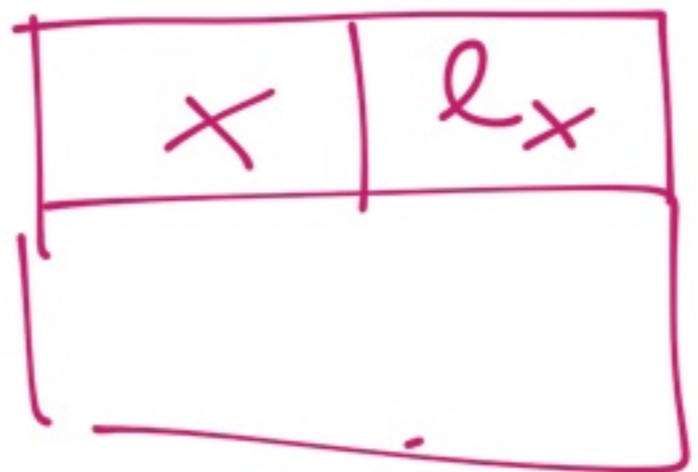
memoria dinamica
(memoria heap)

la allocazione (usando
parole di memoria)
viene fatta durante
le dichiarazioni

allo indirizzo
di memoria



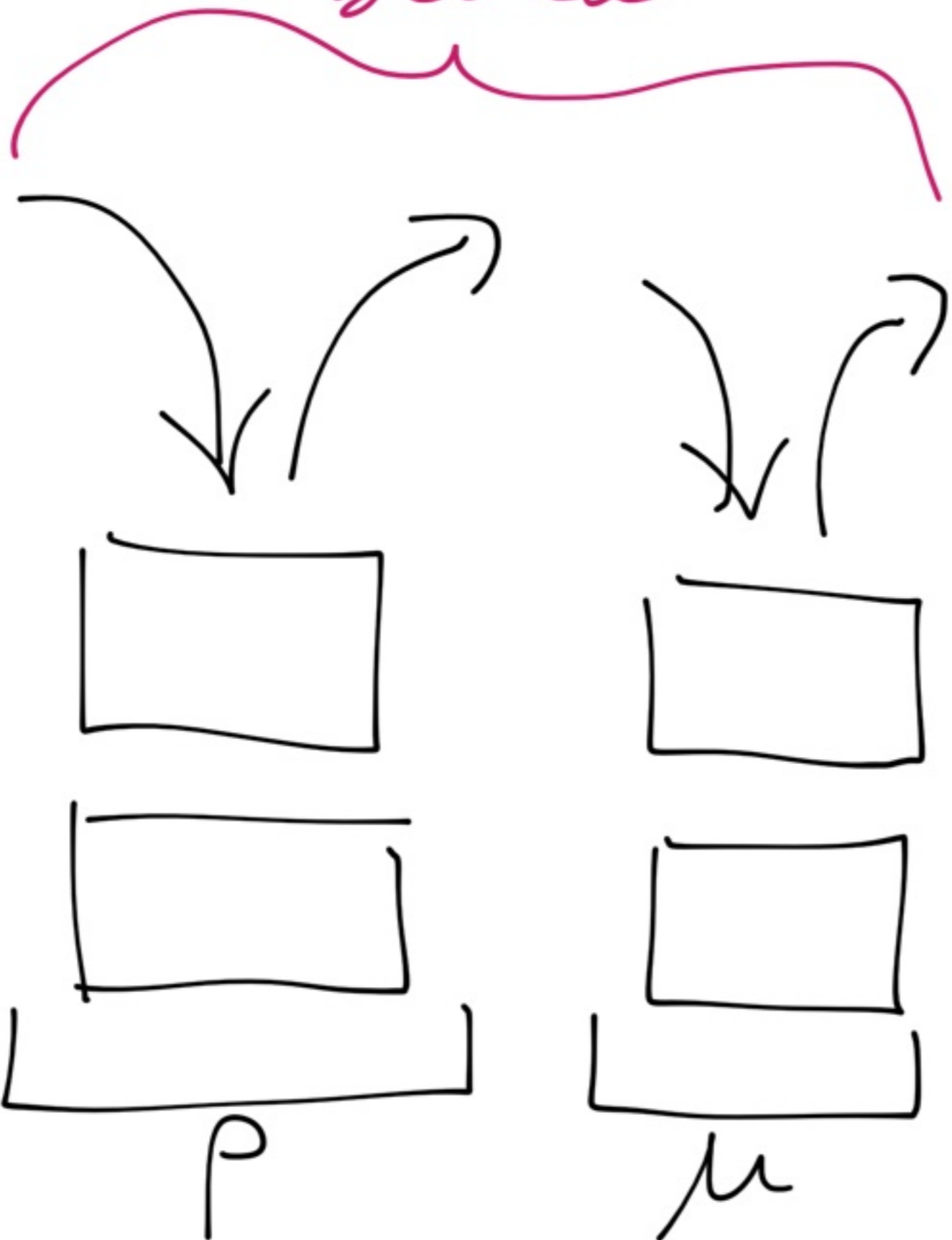
{ int x = 5;



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

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

gestite drei
blöcke

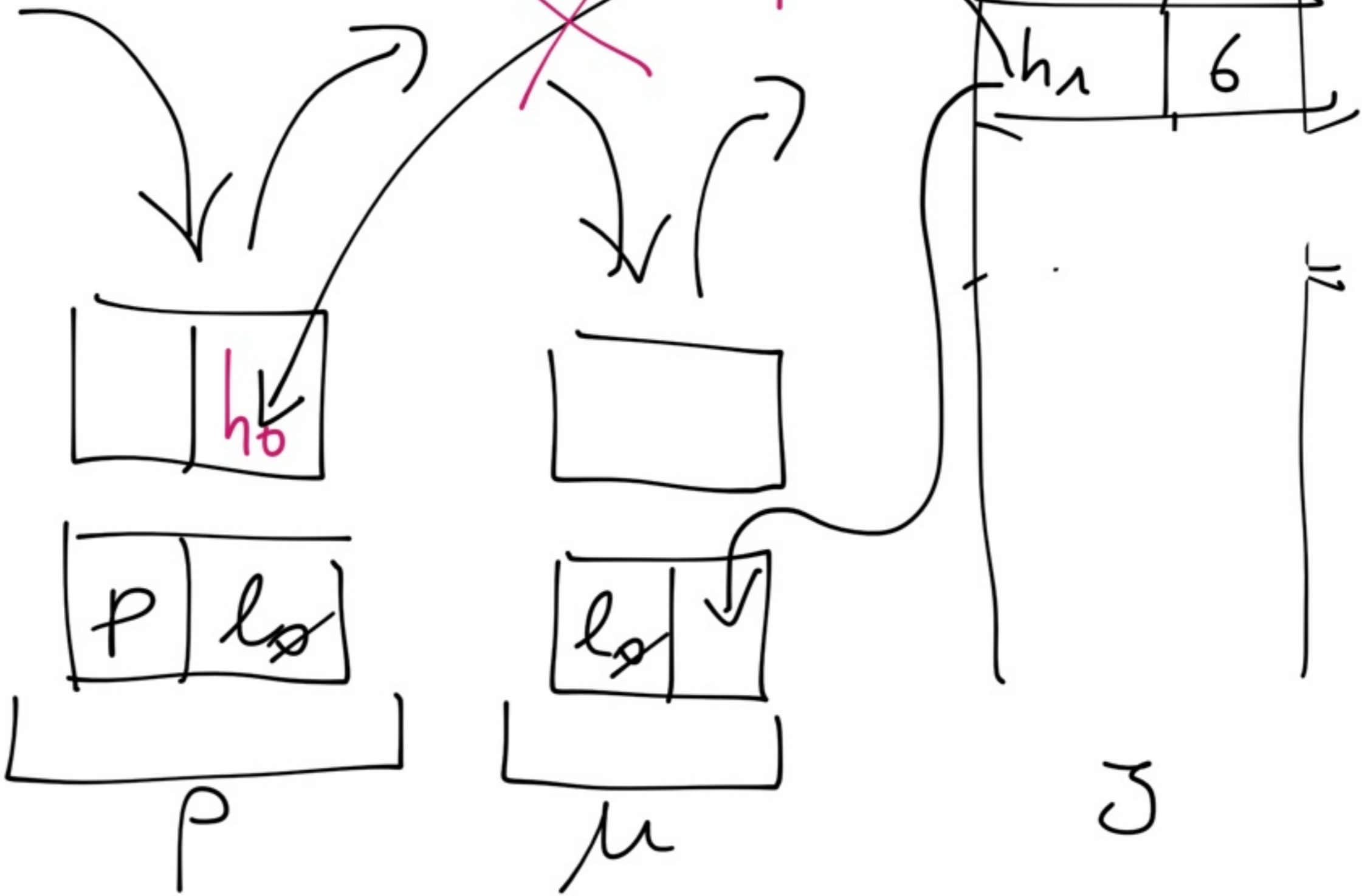


| | |
|-----------------------------|--------------|
| h_3 | 6 |
| h_2 | 7 |
| h_1 | 6 |
| h_0 | 5 |

μ

well'ambiente continuuans
 and also associated:

nome, no! stelleman
 a/ple



le variabili puntatore
possono avere come valori:

- indirizzi delle
memoria a p/b

- indirizzi delle
memoria heap

Non sono adeguate in ρ
 associazioni

(nome, p. ind. mem. heap)

\Rightarrow

il valore delle variabili
 può essere solo mem.
 memor. a pile.

* p1
* p2

salvo delle loc h2
" " " h2

| | |
|----|----|
| p2 | h2 |
| p1 | h1 |

| | |
|----|----|
| h2 | h2 |
| h1 | h2 |

| | |
|--------------|---------------|
| y | h2 |
|--------------|---------------|

| | |
|----|---|
| h2 | 5 |
|----|---|

| | |
|---|----|
| x | h2 |
|---|----|

| | |
|----|---|
| h2 | 5 |
|----|---|

⌈
⌋
p

⌈
⌋
u

3

espressioni particolari

new

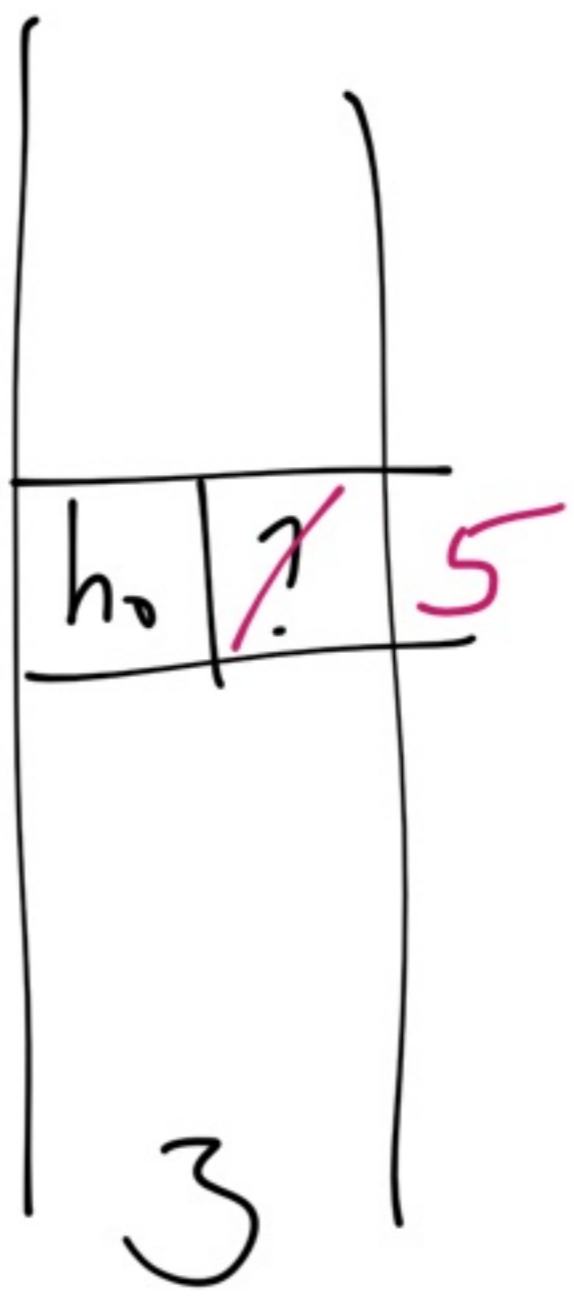
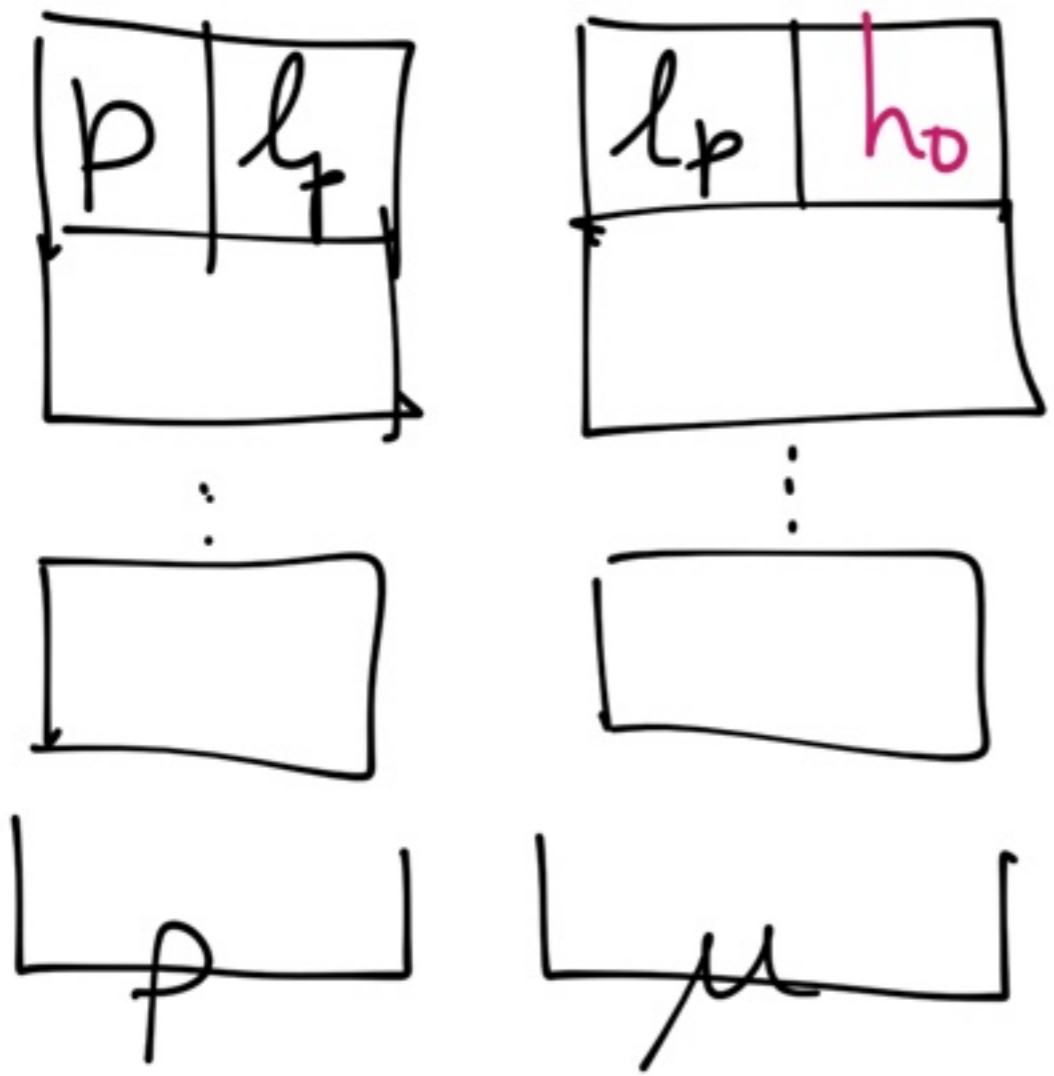
new alloca (nueva)
un indirizzo della
memoria heap

libera

de' come risultato
l'indirizzo allocato

$\{ \text{int } * p = \text{new};$

$*p = 5;$

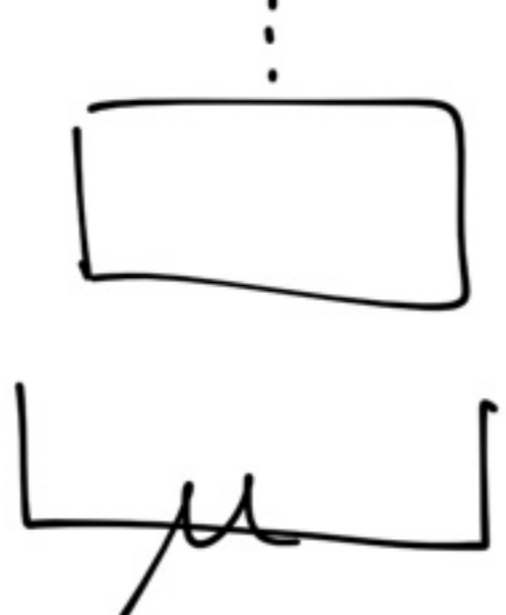
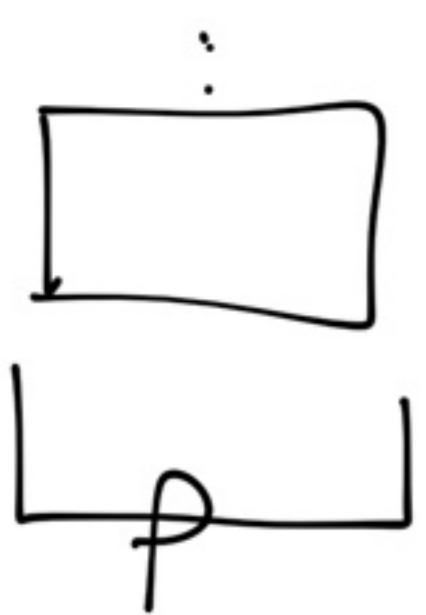
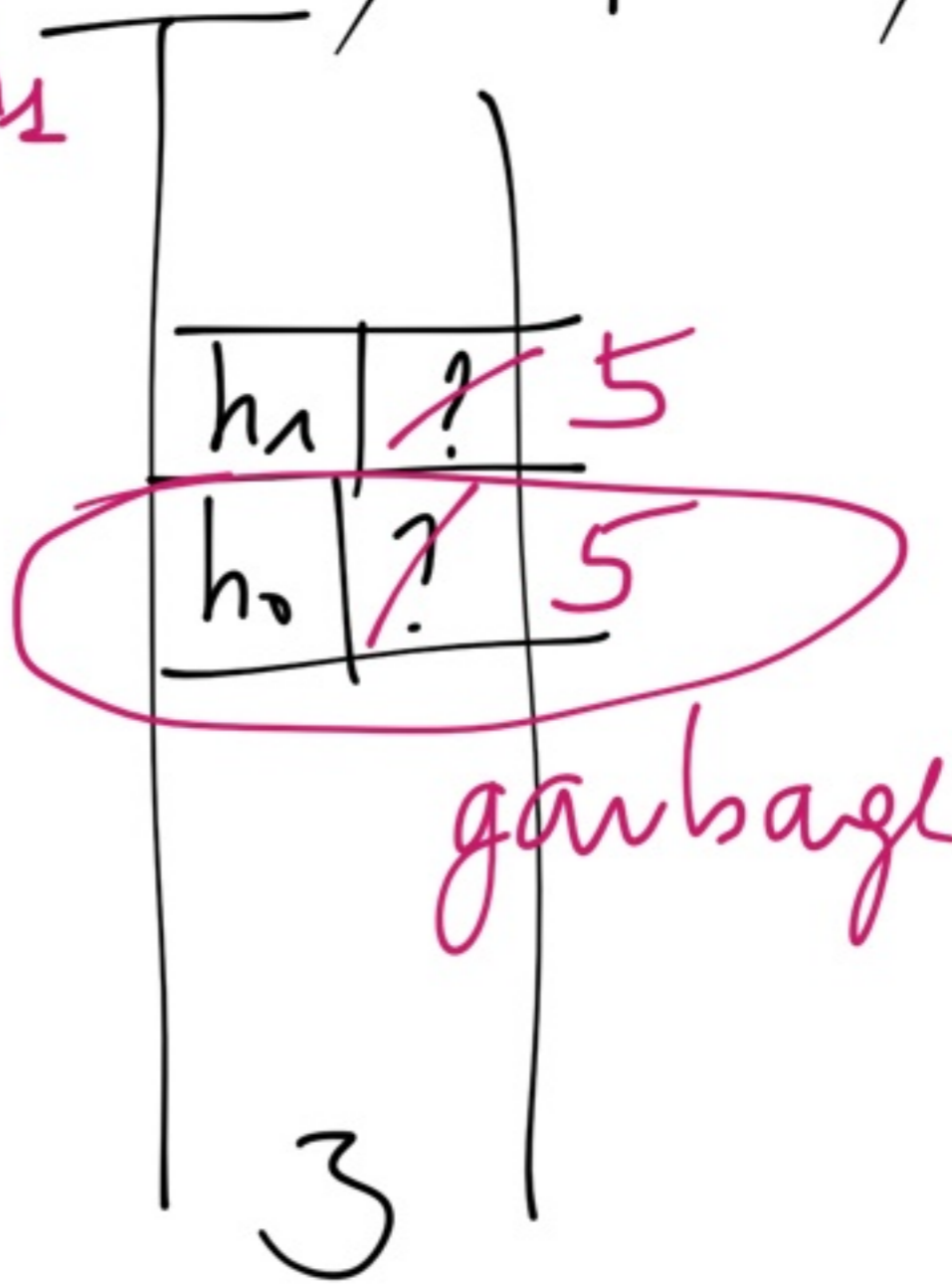
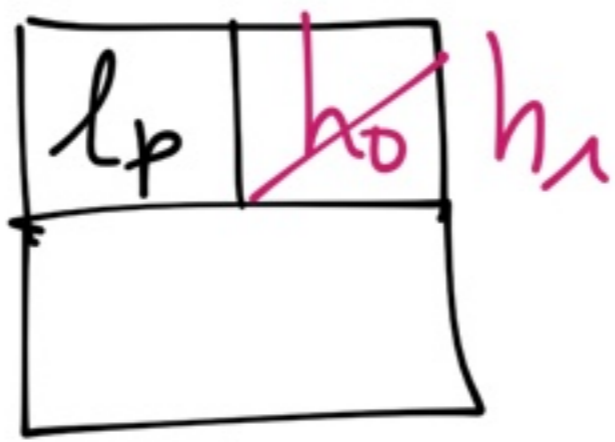
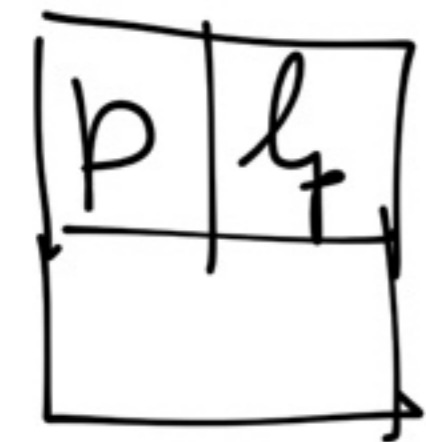


$\{ \text{int} * p = \text{new};$

h₀

$*p = 5; p = \text{new}; *p = 5;$

h₁



$\{ \text{int } * p = \text{new};$

$\text{free}(p);$

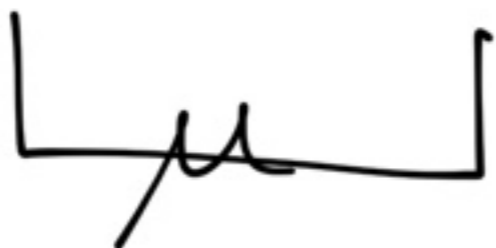
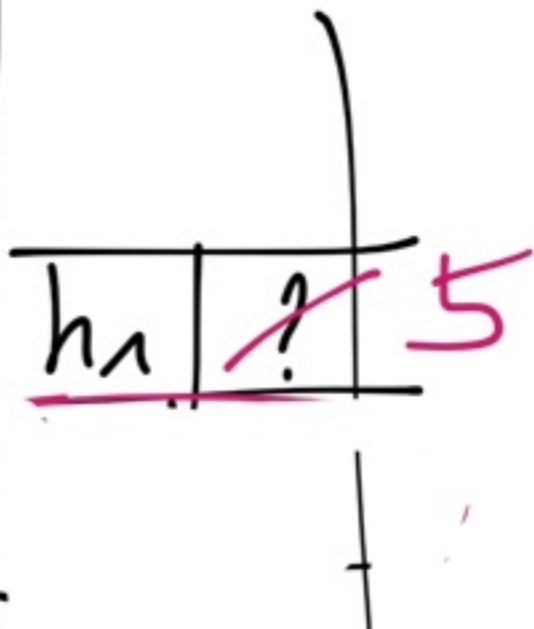
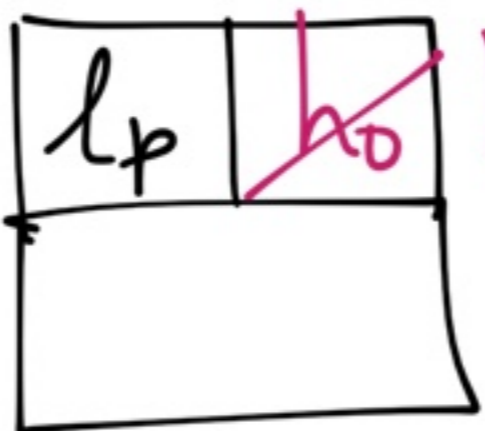
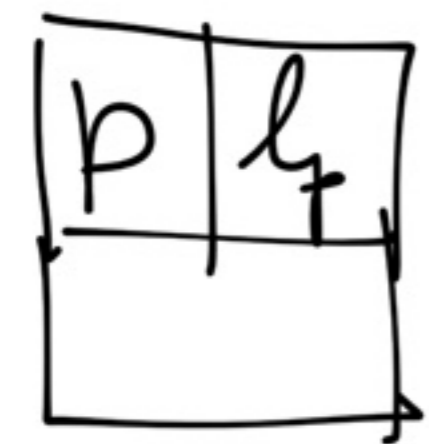
h_0

$*p = 5;$

$\Delta p = \text{new};$

$*p = 5;$

h_1

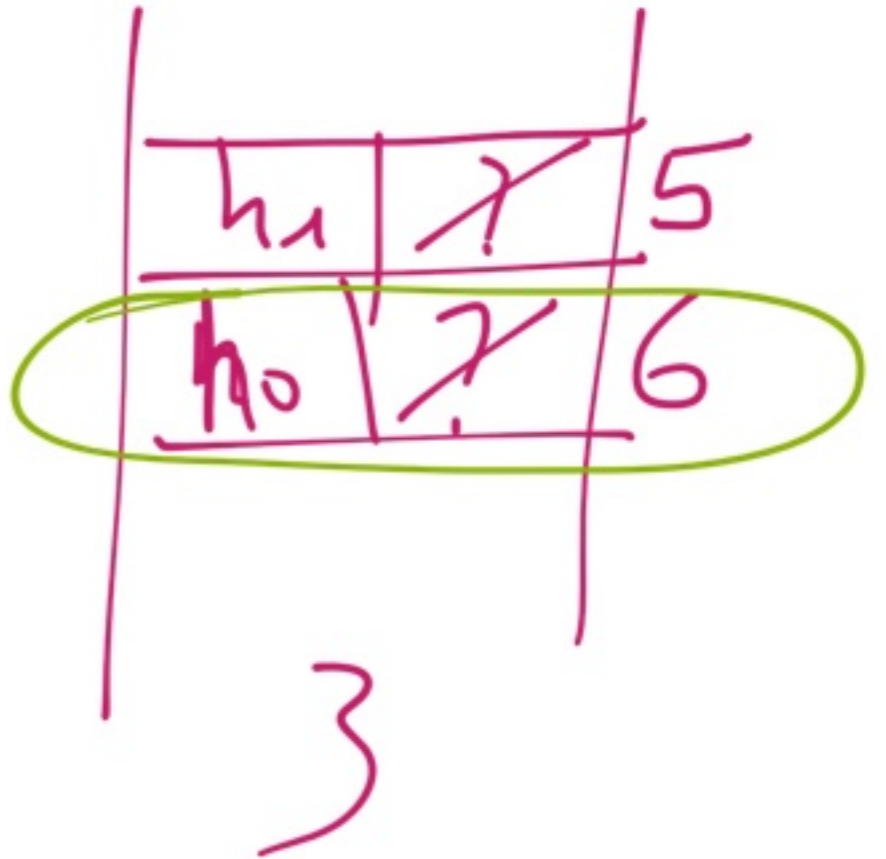
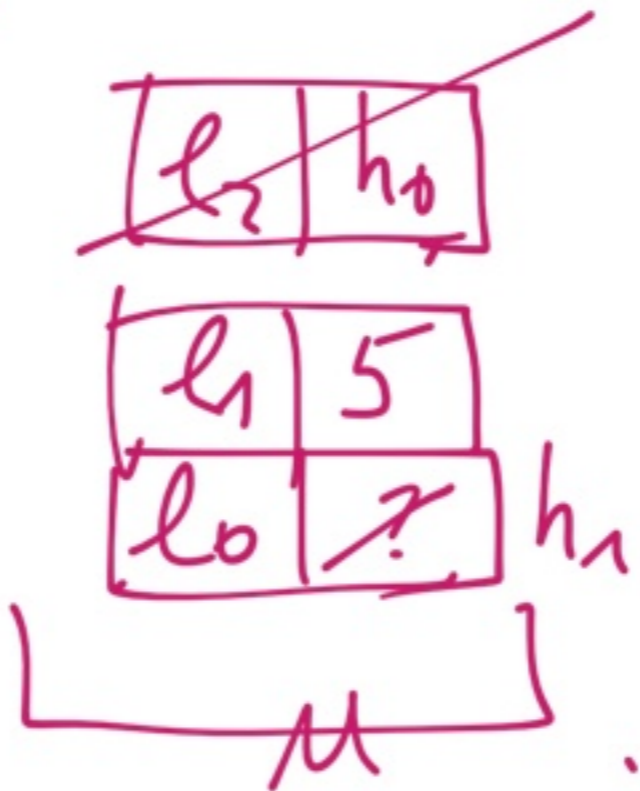
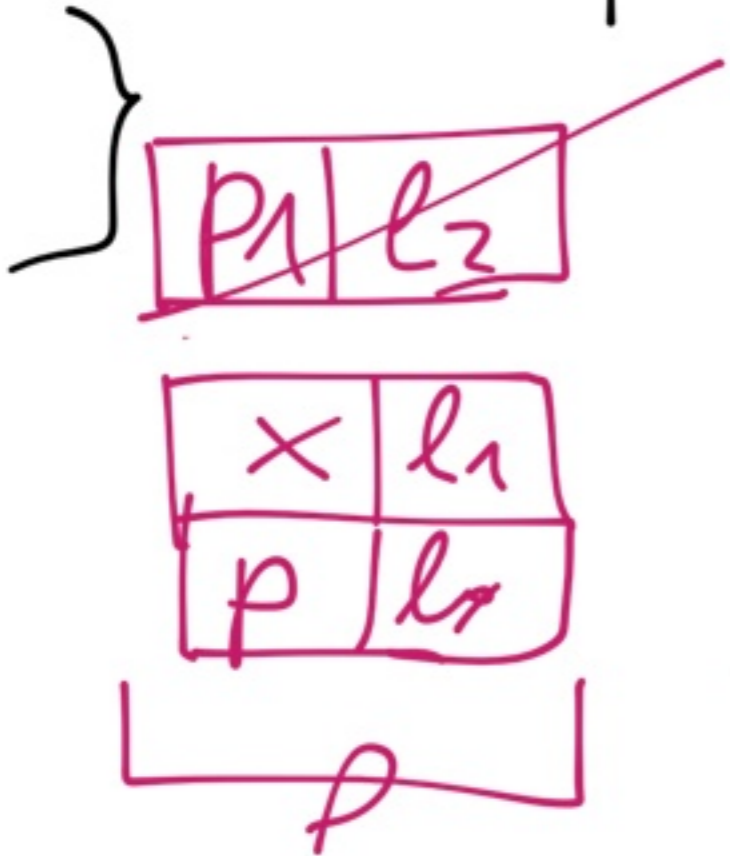


3 |

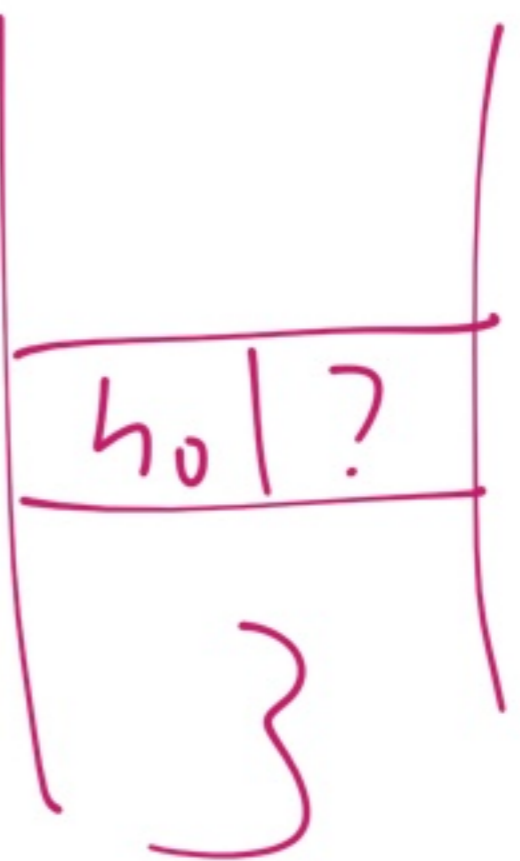
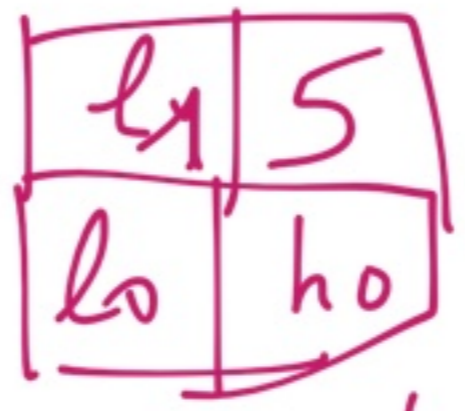
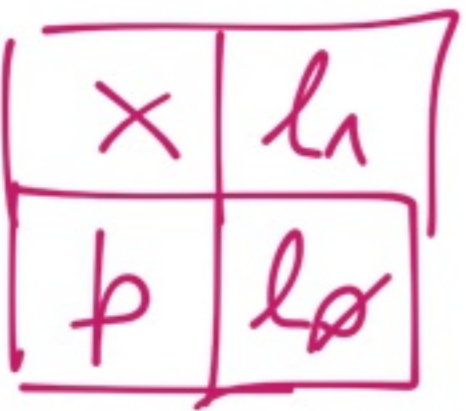
$\left\{ \begin{array}{l} \text{mut} * p; \text{ mut } x = 5; \\ \text{mut} * p1 = \text{new}; \\ p = \text{new}; \end{array} \right.$

$\left. \begin{array}{l} *p1 = *p + 1; \\ *p = x; \end{array} \right\}$

$x = *p$

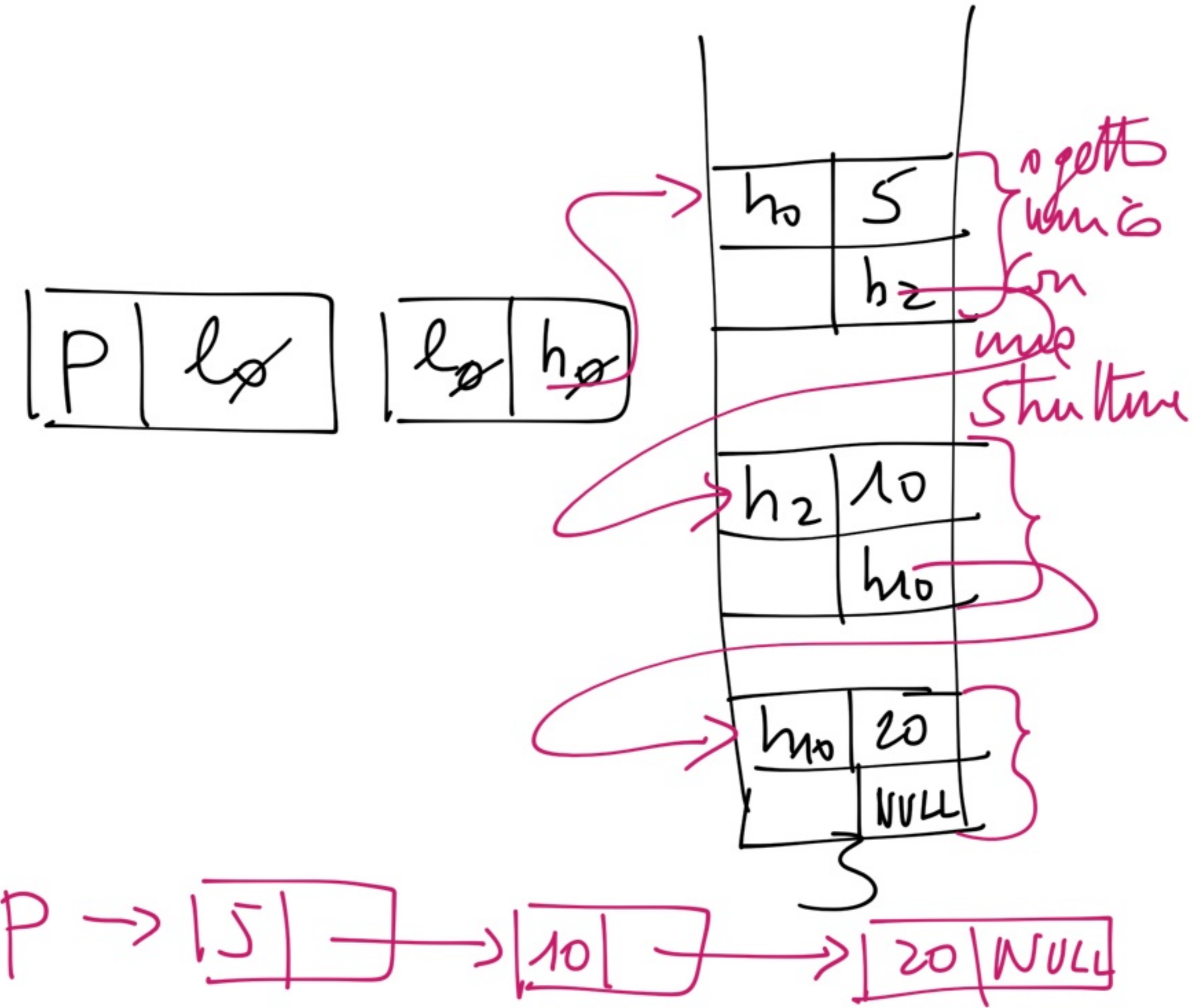


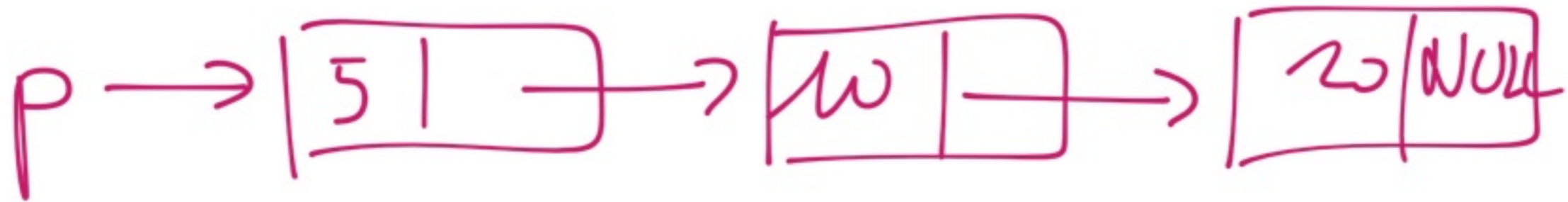
$\left\{ \begin{array}{l} \text{mit } *p = \text{new}, \\ \text{mit } x = 5 \end{array} \right.$
 $\left\{ \begin{array}{l} \text{mit } *p_1; \\ p_1 = p; \end{array} \right.$



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

Come si fa a svincolarci
delle variabili?





Strutture dinamiche

posso modificare le
lunghezze durante
l'esecuzione dei
comandi.

while (cond)



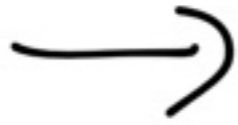
...

... = never;



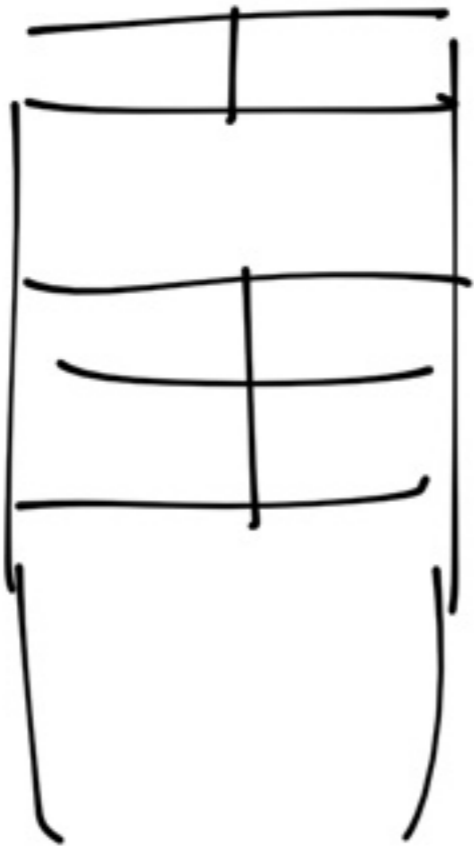
Impiegato C

new



malloc (sizeof (↓))

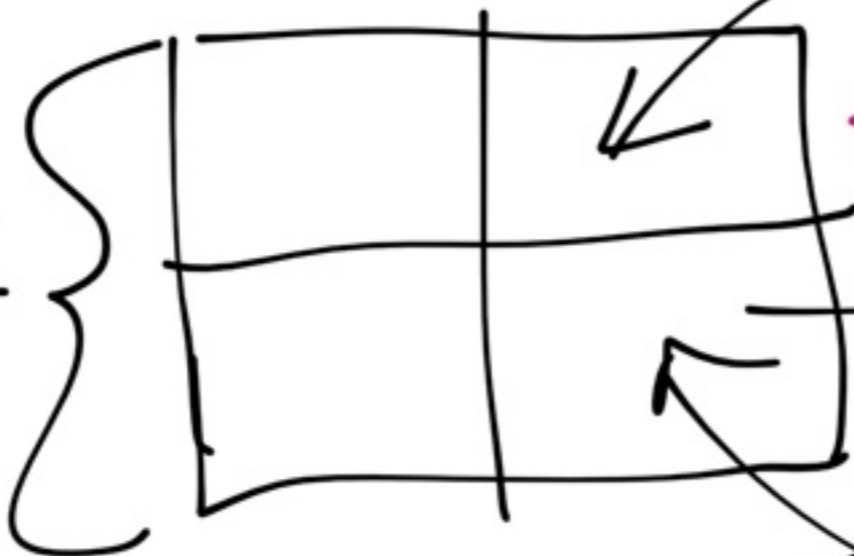
Tipo



argomento
lunghezza della
struttura da
allocare

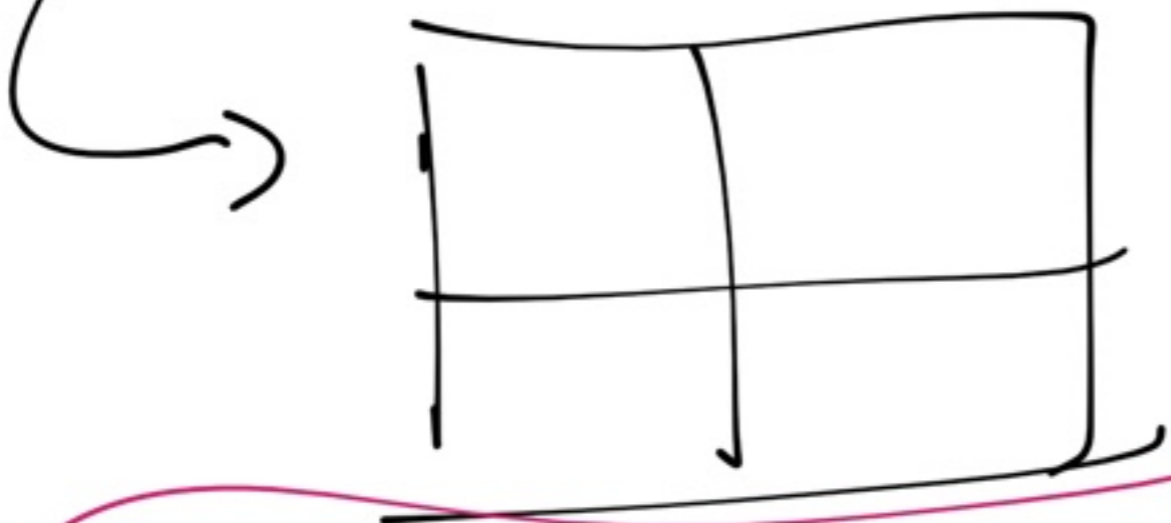
Tipos microssis

struct



mt

pointery



Ci campi hanno un nome

```
struct el { int info;
```

```
struct el * next;
```

```
}
```

nome

Nome dei
Campi

tipo

struct el

Elementliste;

typedef

typedef

Elementliste*

liste di Elementi;

tipo

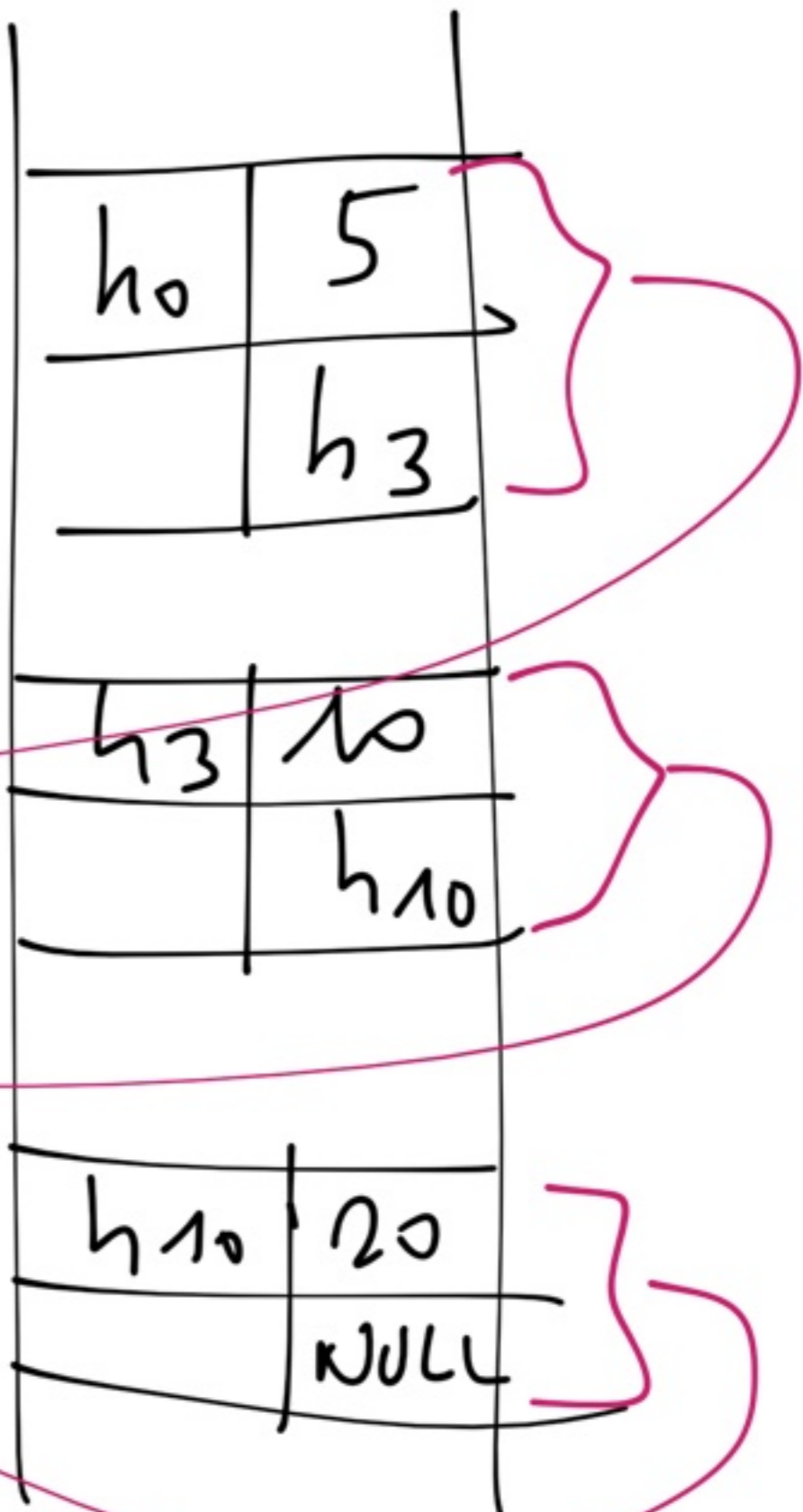
nome

Tippi



Liste Di Elementi

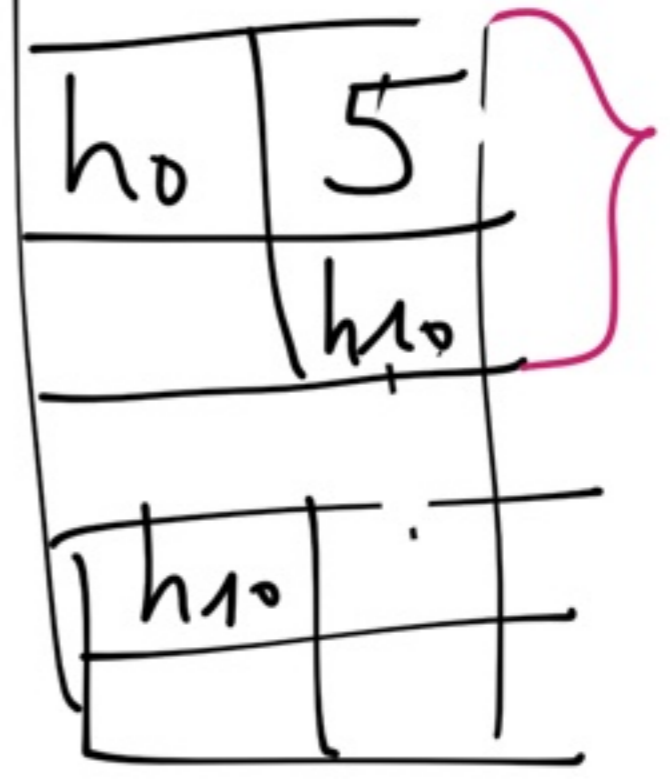
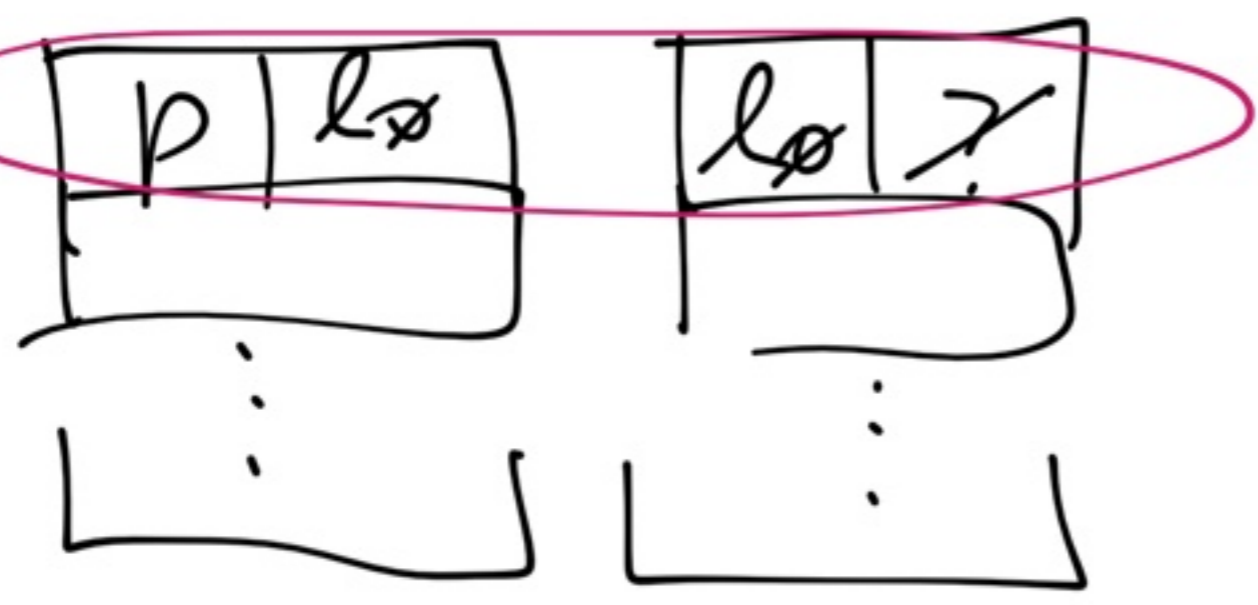
Elemento lista



Liste di Element p;

p = malloc(sizeof(ElementListe));

p->info = 5; p->next = malloc(sizeof(Element));
il campo di nome info dell'oggetto puntato da p



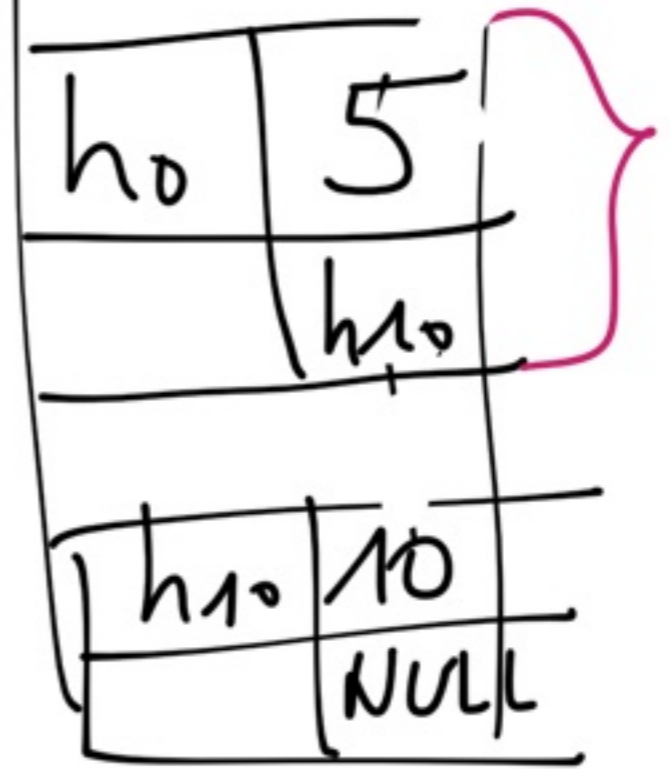
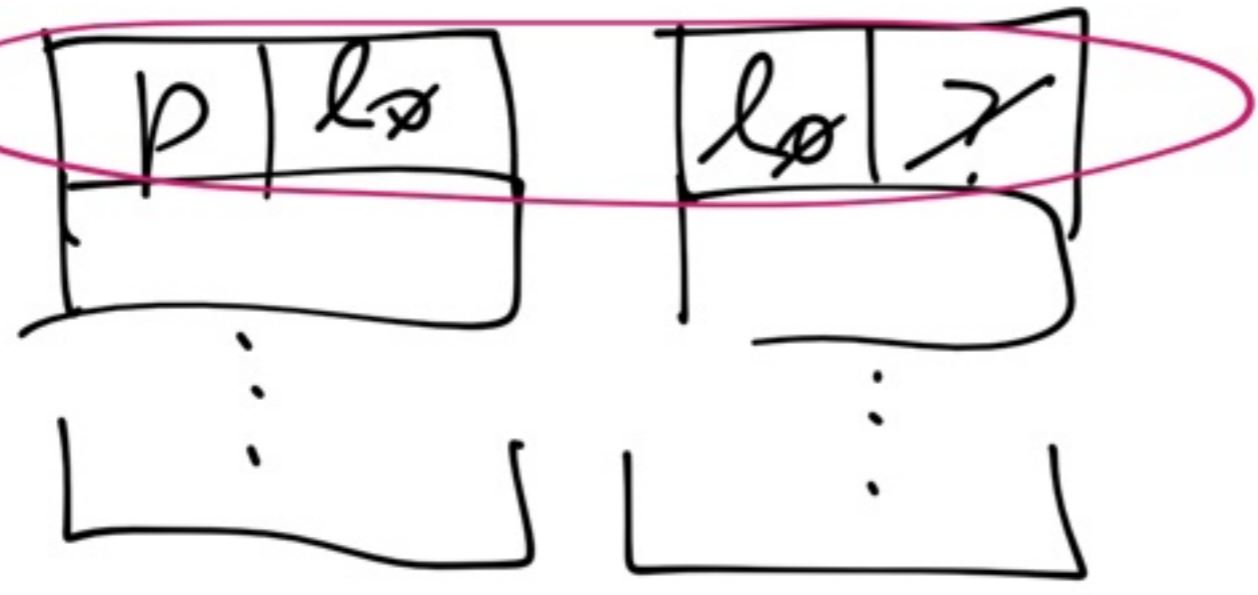
Liste Di Element p;

p = malloc(sizeof(ElementListe));

p->info = 5; p->next = malloc(sizeof(Element));

p->next->info = 10;

p->next->next = NULL;

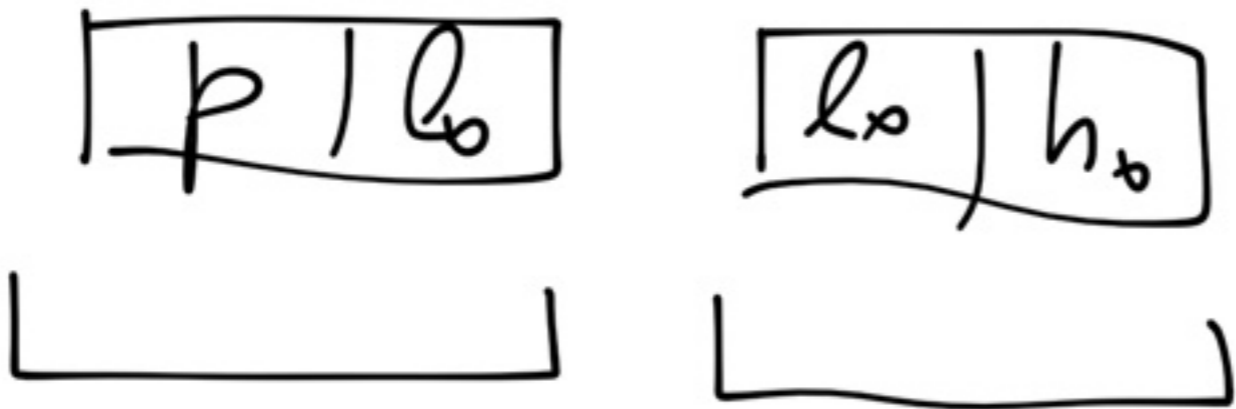
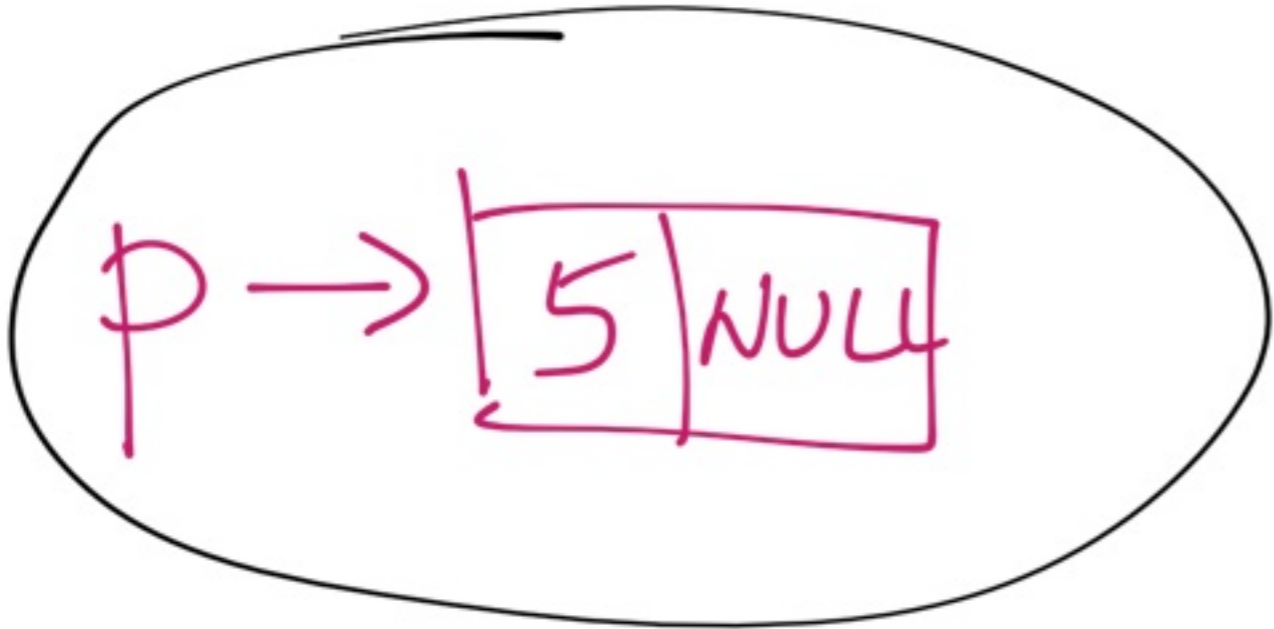


Supponendo una lista non
vuota aggiungere in fondo
un elemento

void aggiungi (lista di Elementi l,
int el)



} Liste Di Elementi $p = \text{malloc}$
 (size of (Elementh
 ite));
 $p \rightarrow \text{val} = 5$;
 $p \rightarrow \text{next} = \text{NULL}$;



```
{ void aggiungi (ListDiElementi l,  
               int el)
```

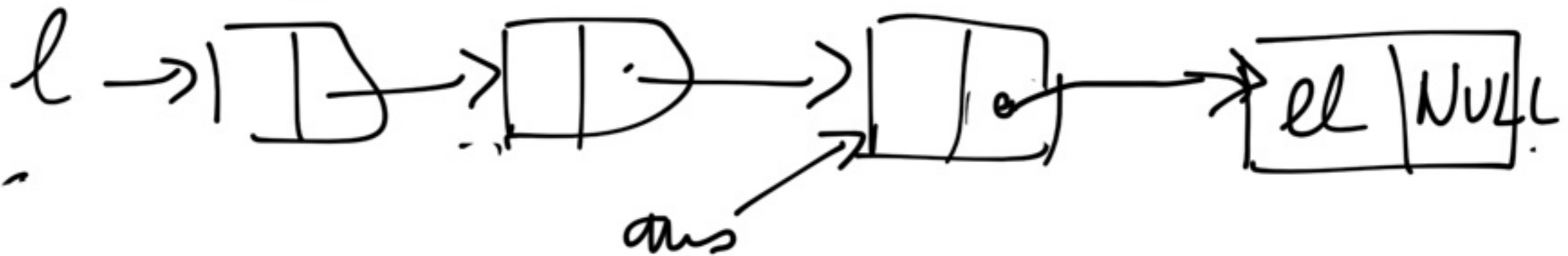
```
{ ListDiElementi ans = l;  
  while (ans->next != NULL)
```

```
    ans = ans->next;
```

```
    ans->next = malloc( ... );
```

```
    ans->next->inf = el;
```

```
    ans->next->next = NULL;  
}
```



$p \rightarrow [5 | \text{NULL}]$

aggiungi (p, 20);

$p \rightarrow [5] \rightarrow [20 | \text{NULL}]$

aggiungi (p, 25);

$p \rightarrow [5] \rightarrow [20] \rightarrow [25 | \text{NULL}]$

