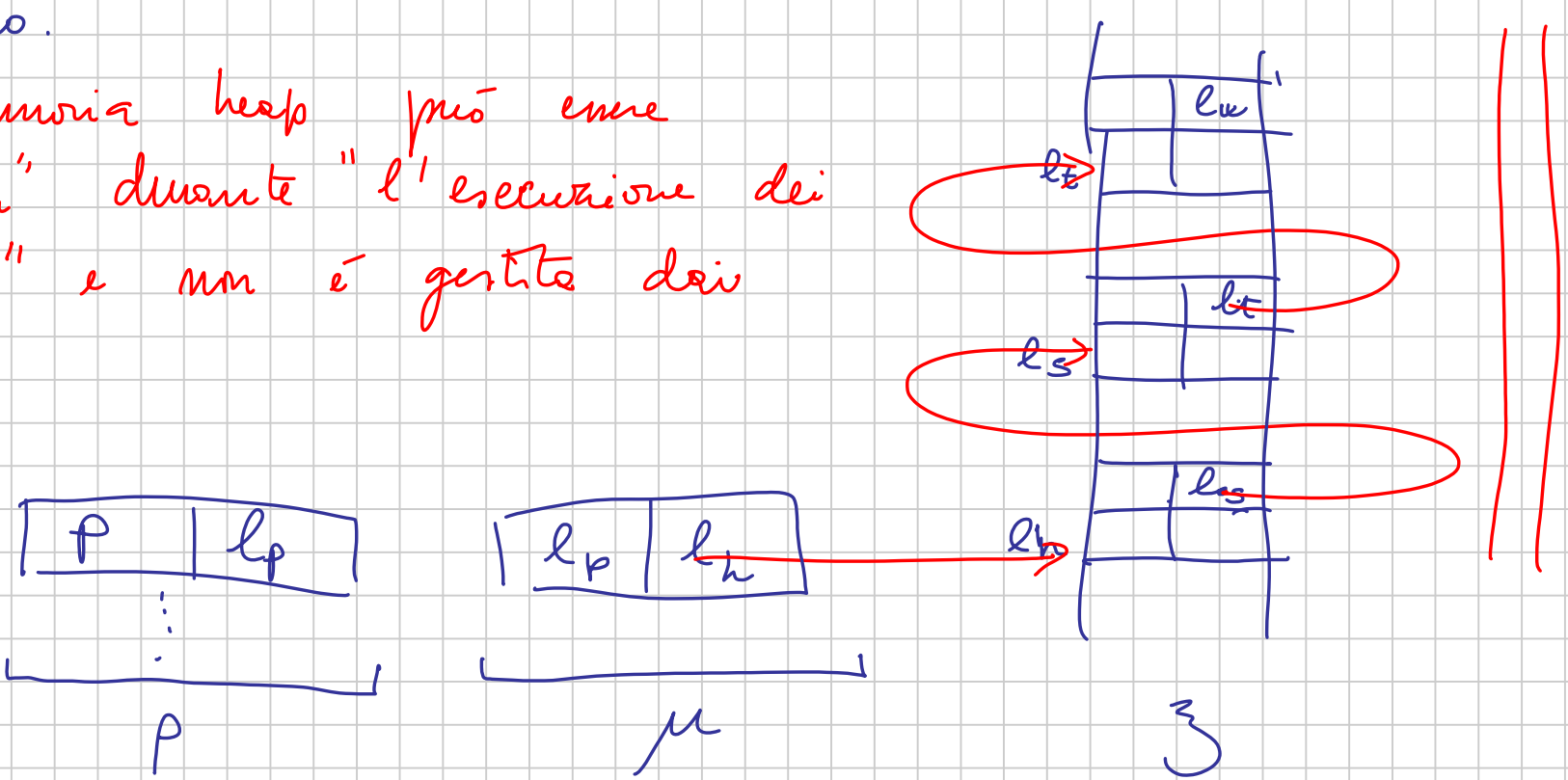
Liste in C (strutture dinamiche allocate nello HEAP)

Titolo nota 24/09/2015

Strutture in cui tutti gli elementi sono COLLEGATI tra di loro.

La memoria heap può essere "allocata" durante l'esecuzione dei comandi" e non è gestita dei blocchi



Come si alloca la memoria heap (memoria dinamica)?

Titolo nota 24/09/2015

Funzione particolare in C

malloc( )

numero di "byte" di memoria da riservare

"memory allocation"

parola di memoria di 8 bit

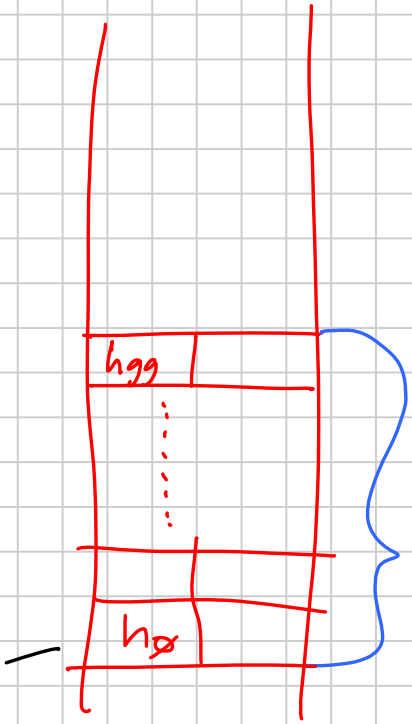
malloc (100)

riservare 100 "byte" di memoria!

il risultato è l'indirizzo del primo "byte" allocato

Titolo nota 24/09/2015

e la memoria heap è tutta libera
(il primo indirizzo disponibile è h_0)



3

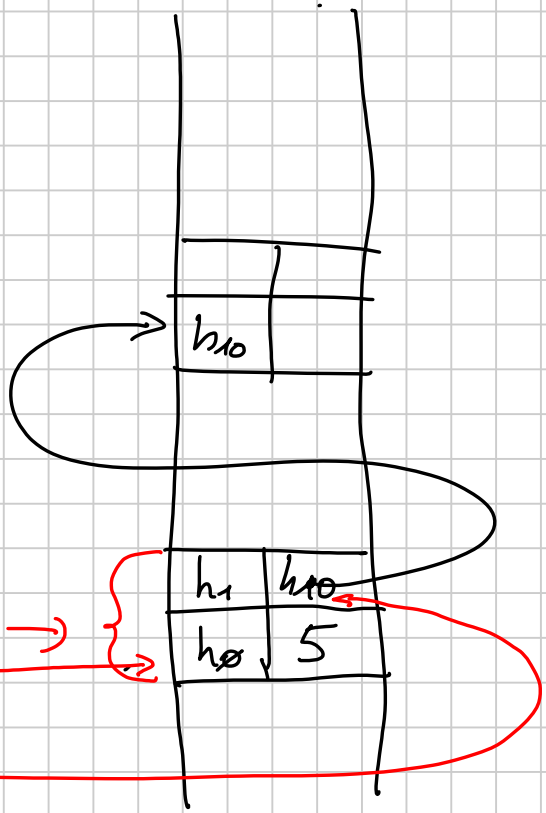
liste

Titolo nota 24/09/2015

Che cosa si usa (quale struttura) in C per rappresentare una collezione di oggetti eterogenei (di tipo diverso)

{ int
{ puntatore all'elemento che segue

```
struct el
{
    int info;
    struct el *next;
}
```

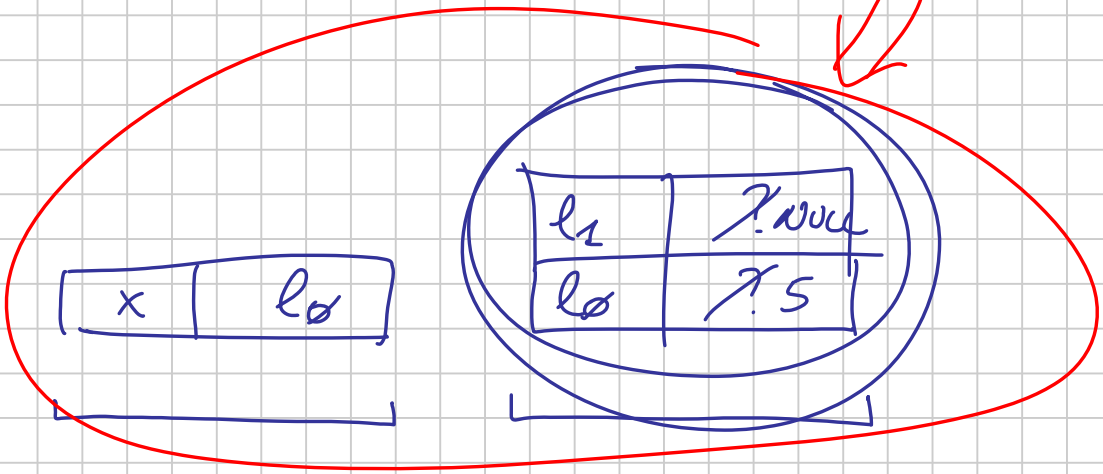


Titolo nota 24/09/2015

```
struct el *x;
x -> info = 5;
x -> next = NULL;
```

corretto

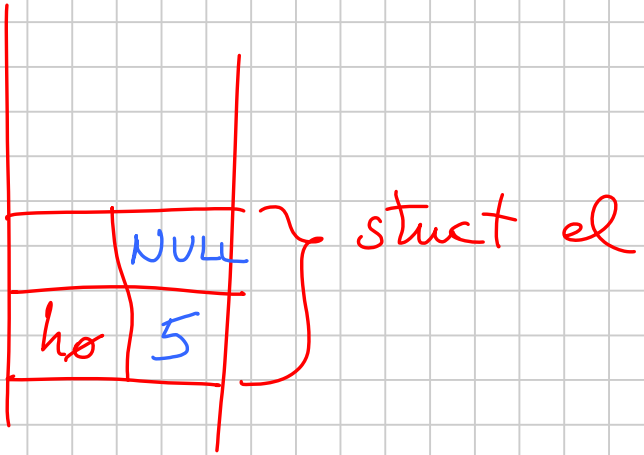
```
struct el x;
x.info = 5;
x.next = NULL;
```



Titolo nota 24/09/2015

```
struct el * x = malloc ( sizeof ( struct el ) )
x -> info = 5 ;
x -> next = NULL ;
```

il numero di "byte" per allocare un oggetto di tipo struct el



Titolo nota 24/09/2015

```
struct el
  { int info;
    struct el * next;
  }

typedef   struct el
type def  ElementoDiLista *

ListaDiElementi  l;
```

definizione

```
ElementoDiLista;
ListaDiElementi
```

nome del nuovo tipo

definizione del nuovo tipo

nome del nuovo tipo

dichiarazioni

```
ElementoDiLista x;
```

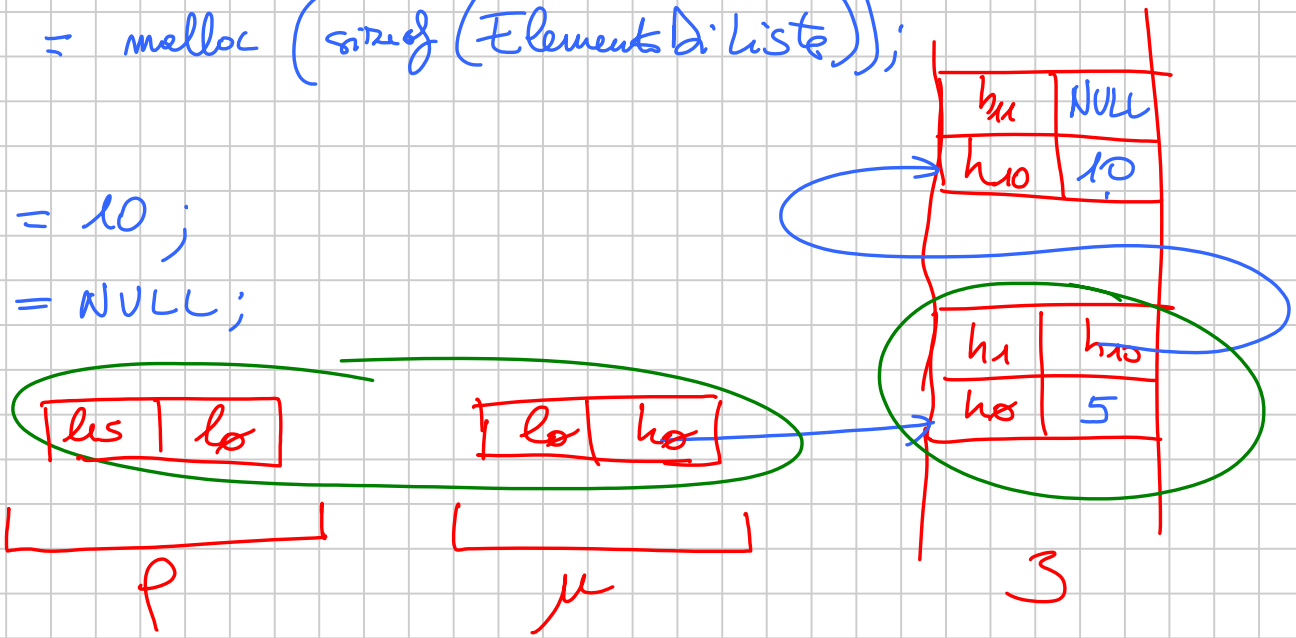
ha tipo struct el

Ho dato un nuovo nome al tipo struct el

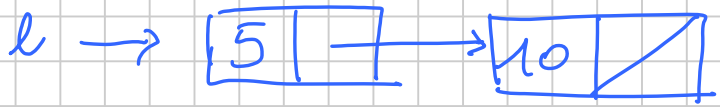
Titolo nota 24/09/2015

```
ListeDiElementi l;   ≡   ElementoDiLista * l;
```

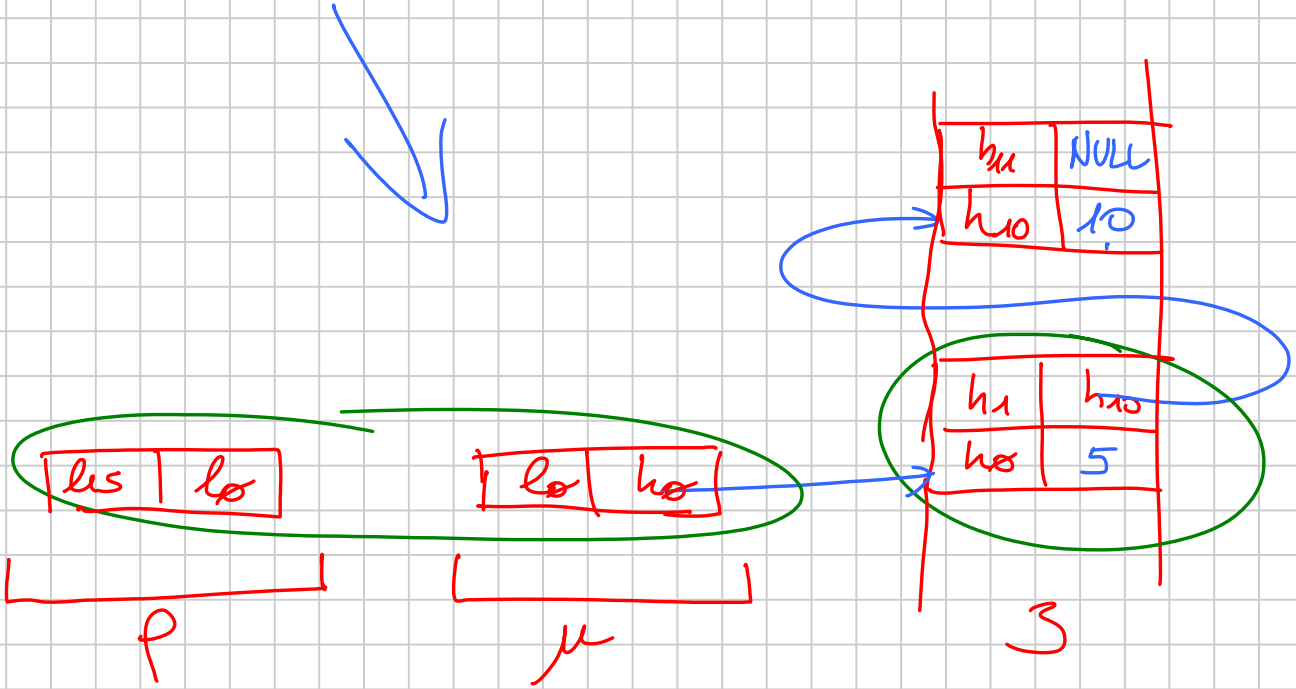
```
ListeDiElementi lis = malloc(sizeof(ElementoDiLista));
lis -> next         = malloc(sizeof(ElementoDiLista));
lis -> info = 5;
lis -> next -> info = 10;
lis -> next -> next = NULL;
```



Titolo nota 24/09/2015



Rappresentazione grafica delle liste



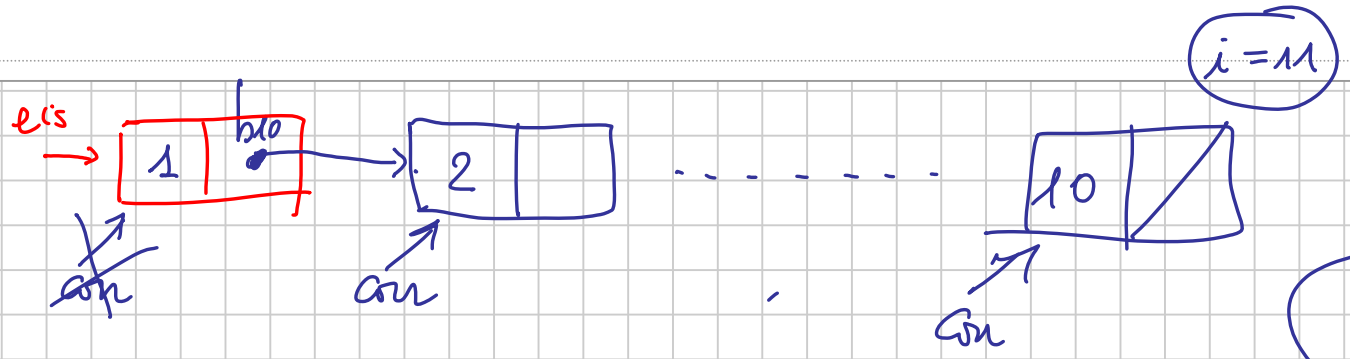
Titolo nota 24/09/2015

l → next → next → ... → info = 10 ;

10 volte

Vogliamo usare un ciclo (perché così possiamo creare liste lunghe quanto vogliamo)

Titolo nota 24/09/2015



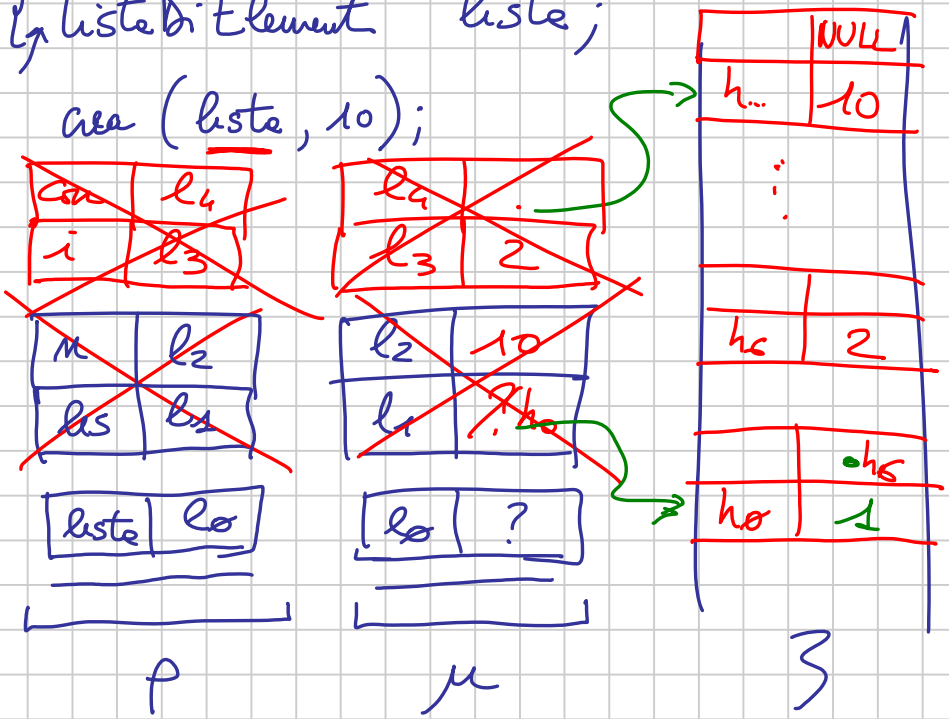

```
{ int i = 1;
  ListeDiElement lis = malloc ( sizeof (ElementoDiLista ));
  ListeDiElement corr = lis;
  lis -> info = i;
  i = i+1;
  while (i <= 10)
    { corr -> next = malloc (sizeof (ElementoDiLista));
      corr -> next -> info = i;
      corr = corr -> next;
      i = i+1;
    }
  corr = corr -> next;
  corr -> info = i;
  corr -> next = NULL;
```

Titolo nota 24/09/2015

```
void crea ( listaDiElementi lis, int n)
{ int i = 2;
  ListaDiElementi corr;
  lis = malloc (sizeof( ElementoDiLista));
  corr = lis;
  lis -> info = 1;
  while (i <= n)
    { corr -> next = malloc ( sizeof (EDL));
      corr = corr -> next;
      corr -> info = i;
      i = i+1;
    }
  corr -> next = NULL;
}
```

Creare una lista con almeno un elemento;

```
main ()
{ ListaDiElementi liste;
  crea (liste, 10);
```



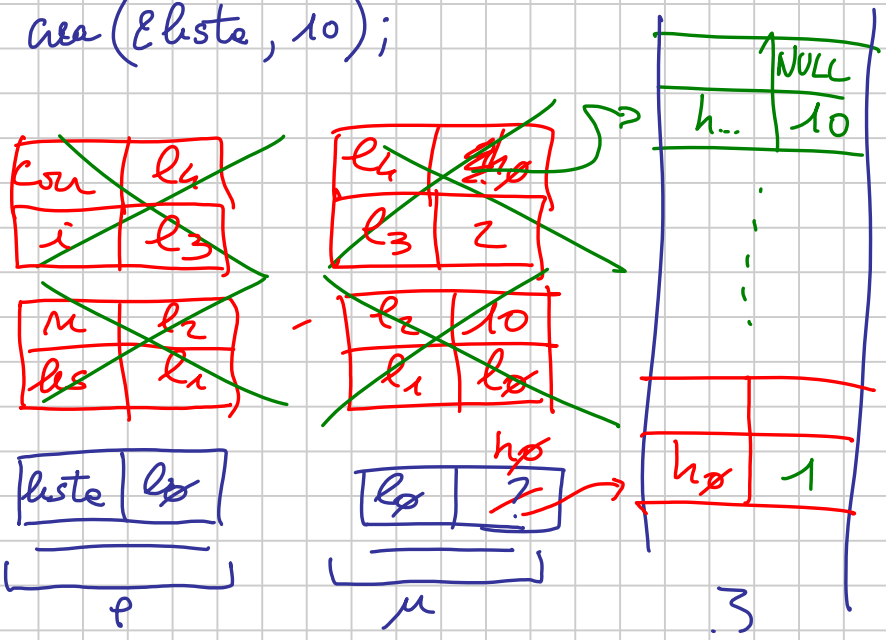
ElementoDiLista **

Titolo nota 24/09/2015

```
void crea ( listaDiElement *lis, int n)
{ int i = 2;
  listaDiElement corr;
 *lis = malloc (sizeof( ElementoDiLista));
  corr =*lis;
 *lis -> info = 1;
  while (i <= n)
    { corr -> next = malloc ( sizeof (EDL));
      corr = corr -> next;
      corr -> info = i;
      i = i+1;
    }
  corr -> next = NULL;
}
```

Creare una lista con almeno un elemento;

```
main ()
{ listaDiElement liste;
  crea (&liste, 10);
}
```



Aggiungere un elemento in fondo a una lista non vuota (con una procedura)

Titolo nota 24/09/2015

```
void aodd (listaDiElement lis, int el)
{ listaDiElement corr = lis;
  while (corr -> next != NULL)
      corr = corr -> next;
  corr -> next = malloc (sizeof (ELL));
  corr = corr -> next;
  corr -> info = el;
  corr -> next = NULL;
}
```

Se non si deve modificare il puntatore all'inizio della lista è inutile passare il puntatore al puntatore