lunedì 12 novembre 2018 08:18

```
typedef
struc node
{
   int info;
   struct node * next;
}

ElementoDiListe;
```

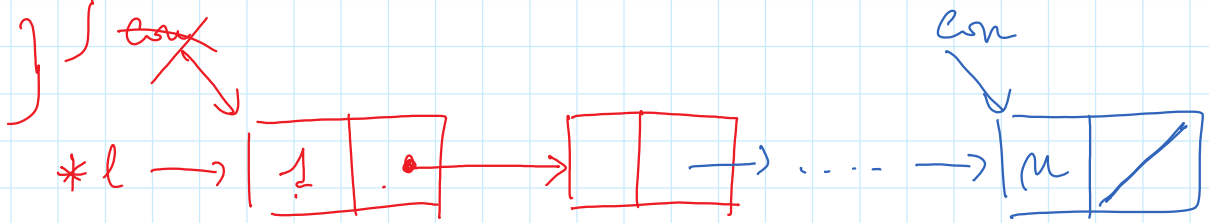
```
struct el
{ int info;
  struct el * next;
}

typedef struct el
        ElementoDiListe;

type def
   ElementoDiListe *
  listeDiElementi;
```

ElementoDiListe * ≡ listeDiElement

ElementoDiListe * * ≡ ListeDiElementi *

```
void crealista (ElementoDiLista ** l, int n)
{ int i; ElementoDiLista * corr;
  *l = NULL;
  if (n>0)
  { *l = malloc(sizeof(ElementoDiLista));
    (*l)->info = 1;
    corr = *l;
    for (i=2; i<=n; i++)
    {
      corr->next = malloc(sizeof(ElementoDiLista));
      corr = corr->next;
      corr->info = i;
    }
    corr->next = NULL;
  }
```

(*corr).next




```
{ corr->next = malloc( ... );
  corr->next->info = i;
  corr = corr->next;
```

equivalente

```
    corr -> next -> info = i;
    corr = corr -> next;
}
```

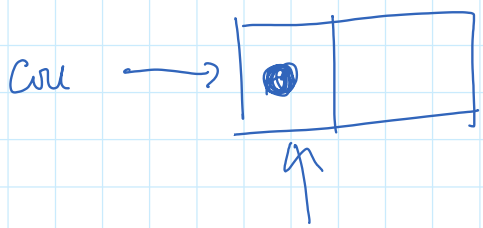
equivalente

Se avremmo voluto dare valori alle liste letti dall'esterno, al posto di:

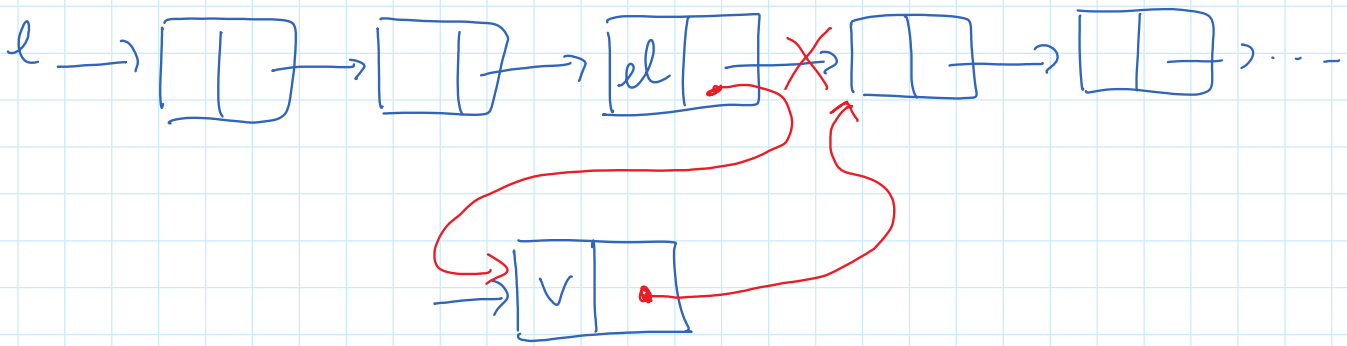
```
corr -> info = i;
```

avremmo dovuto scrivere:

```
scanf ("%d", &(corr -> info));
```

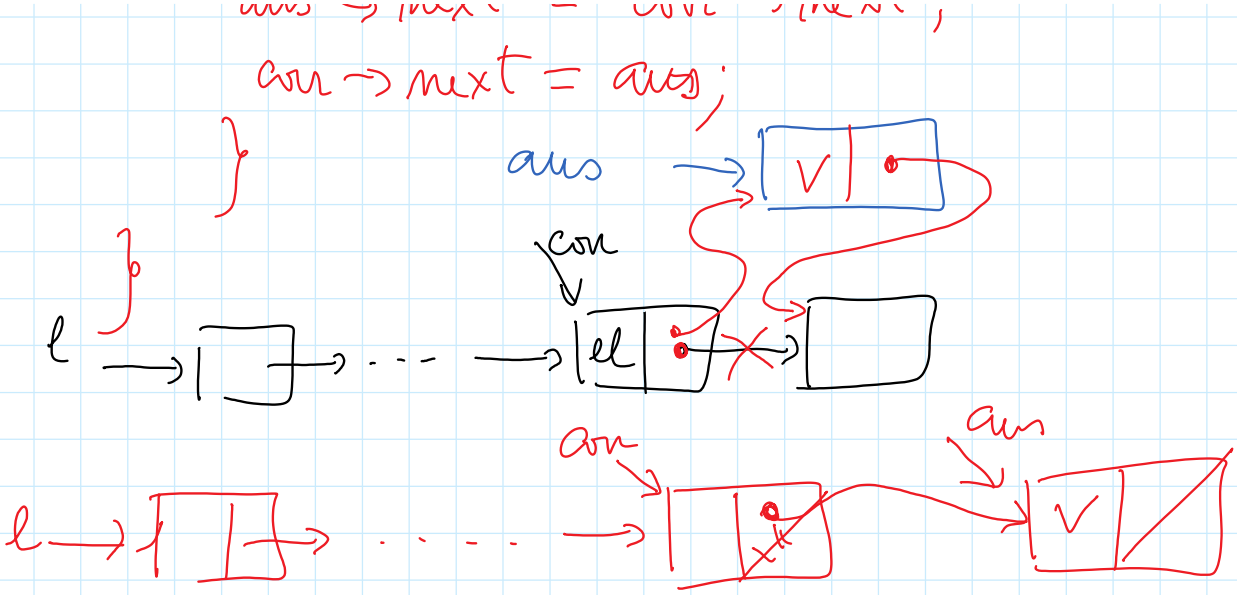


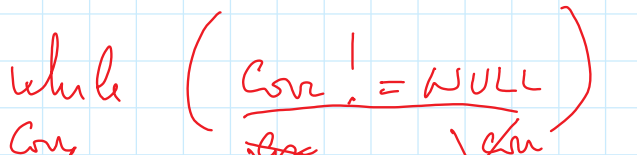
Aggiungere ad una lista un elemento V dopo la prima occorrenza del valore el.
Se el non esiste la lista rimane invariata

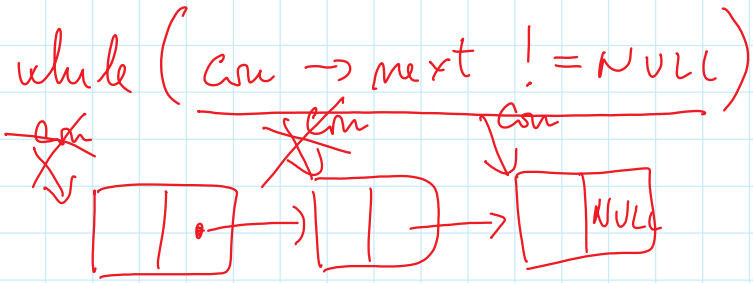



```
void addAfter(ElementoDiLista * l, int v,
              int el)
{ int trovato = 0;
  ElementoDiLista * corr = l;
  while(!trovato && corr != NULL)
    if (corr->inf == el) trovato = 1;
    else corr = corr->next;
  if (trovato)
  {ElementoDiLista* aus = malloc(sizeof(EDL));
   aus->info = v;
   aus->next = corr->next;
   corr->next = aus;
```

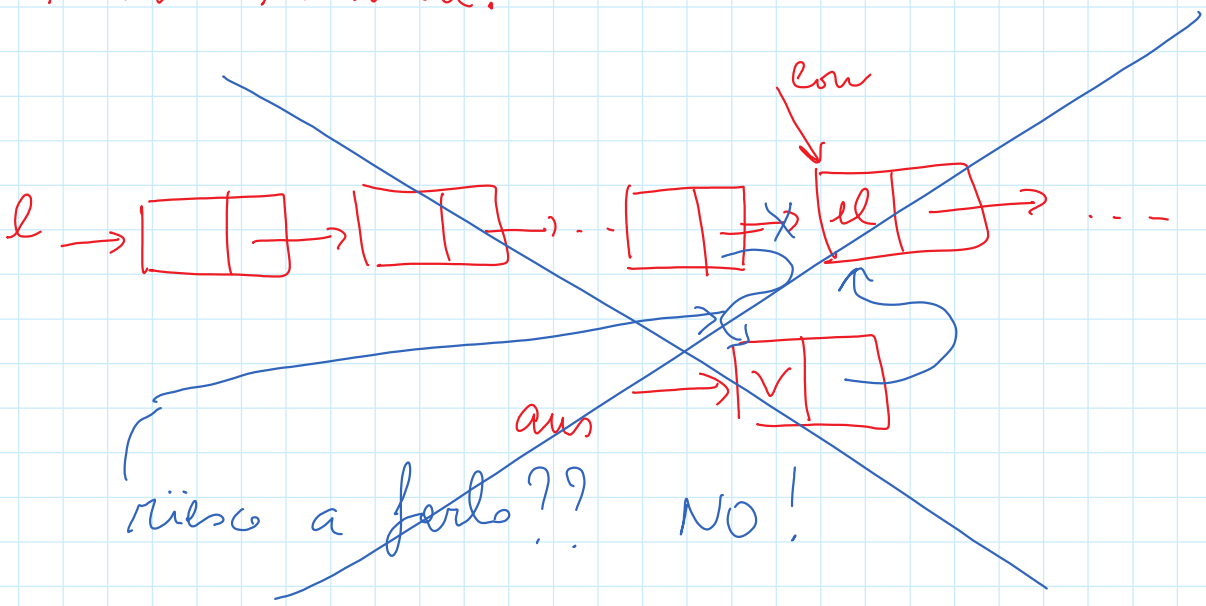
stiamo ancora esaminando elementi della lista


aus->next = aus;
cur->next = aus;
aus
V
cur
l
cur
aus
V

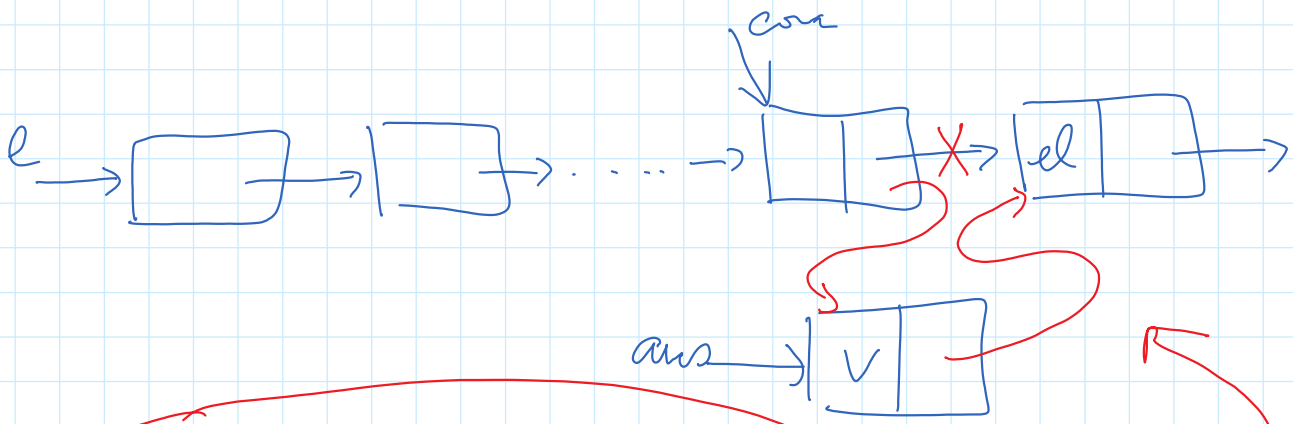

while (cur != NULL)
cur = cur->next;
cur
cur
cur
NULL
cur = NULL


while (cur->next != NULL)
cur = cur->next;
cur
cur
cur
NULL

Aggiungere un nuovo elemento contenente v ad una lista, PRIMA della prima occorrenza del valore el. Se el non c'è la lista rimane invariata.



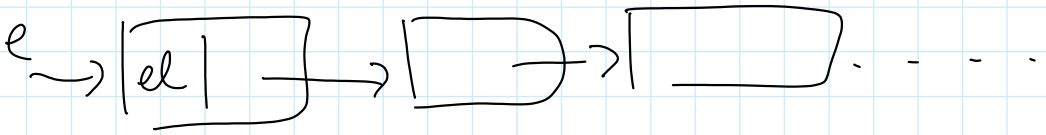
riesco a farlo?? NO!




```
aus -> next = con -> next;
con -> next = aus;
```

```
while (!trovato && con != NULL)
    if (con -> next -> info == el) trovato = 1;
```

```
if (cur -> next -> info == el) trovato = 1;
else cur = cur -> next;
```



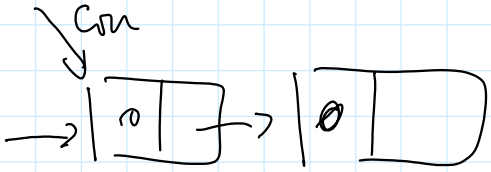
il parametro l che tipo dovrà avere?

ElementoDiLista * * l !!

```
void addbefore (ElementoDiLista **l, int v; int el)
{ int trovato = 0;
  ElementoDiLista *corr = *l;
  while (!trovato && corr != NULL)
    if (corr->next->info == el) trovato = 1;
    else corr = corr->next;
```

corr->next != NULL

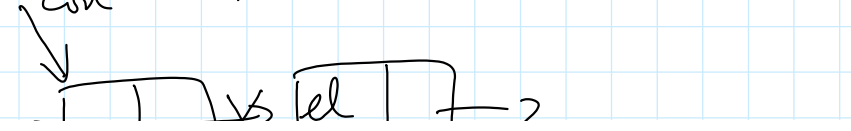
OK



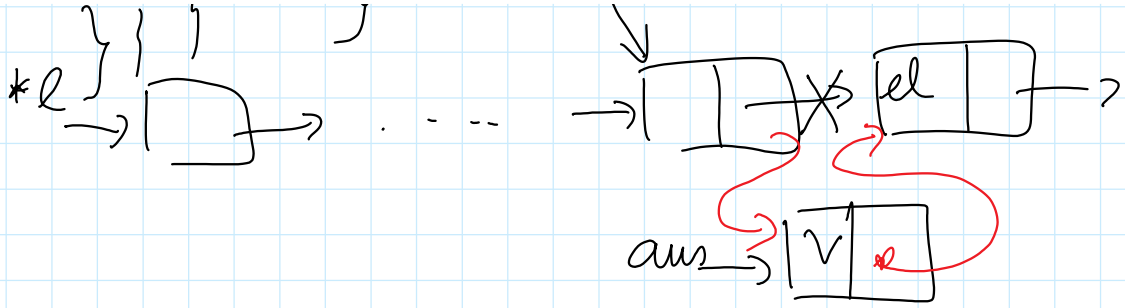
Non guarda il primo elemento.

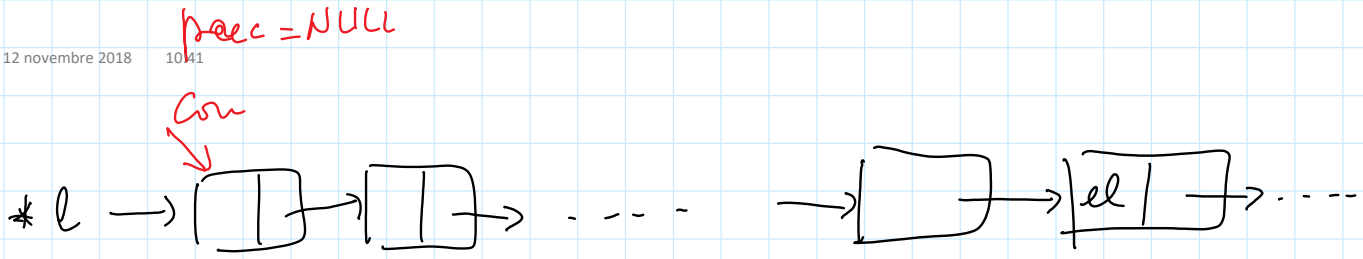
```
void addBefore (ElementoDiLista **l, int v, int el)
{ if (*l != NULL)
  { if ((*l)->info == el)
    { ElementoDiLista * aus = malloc(sizeof(EDL));
      aus->info = v;
      aus->next = *l;
      *l = aus;
    }
  else
  { ElementoDiLista * corr = *l;
    int trovato = 0;
    while (!trovato && corr->next != NULL)
      if (corr->next->info == el) trovato=1;
      else corr = corr->next;
    if (trovato)
    { ElementoDiLista * aus = malloc(sizeof(EDL));
      aus->info = v;
      aus->next = corr->next;
      corr->next = aus;
    }
  }
}
```

corr



*l


*l
el
aus
v



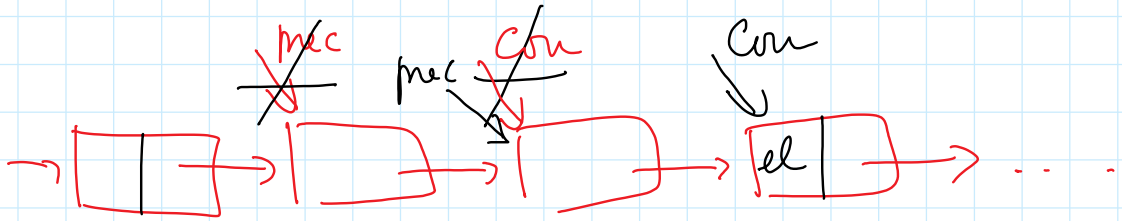
prec = NULL

curr = *l

Guardiamo il campo info dell'elemento puntato da curr.

Facciamo scorrere i due puntatori

```
prec = curr;
curr = curr->next
```



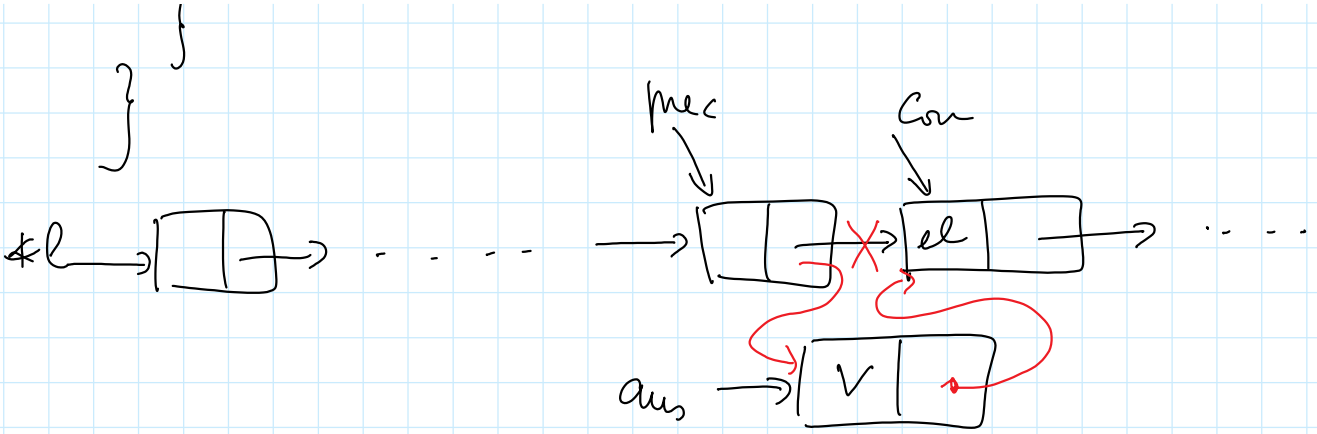
Se el è contenuto nel primo elemento?

```
void addbefore(ElementoDiLista **l, int v, int el)
{ int trovato=0;
  ElementoDiLista *prec=NULL, *corr=*l;
  while(!trovato && corr!=NULL)
    if(corr->info == el) trovato=1;
    else { prec = corr;
           corr = corr->next;
         }
  if(trovato)
  { ElementoDiLista *aus = malloc(sizeof(EDL));
    aus->info = v;
    if(prec==NULL)
    { aus->next = *l;
      *l = aus;
    }
    else
    { aus->next = corr;
      prec->next = aus;
    }
  }
```

{ prec = prec->next;
corr = corr->next;
}

questo è scorretto quando prec == NULL

aus->next = corr;


*l
prec
cor
el
aus
v

lunedì 12 novembre 2018 10:57

Cancellare il primo elemento di una lista, se c'è.

```
void canc (ElementoDiListe ** l)
{ if (*l != NULL)
  { ElementoDiListe * aus = *l;
    *l = (*l)->next;
    free(aus);
  }
}
```