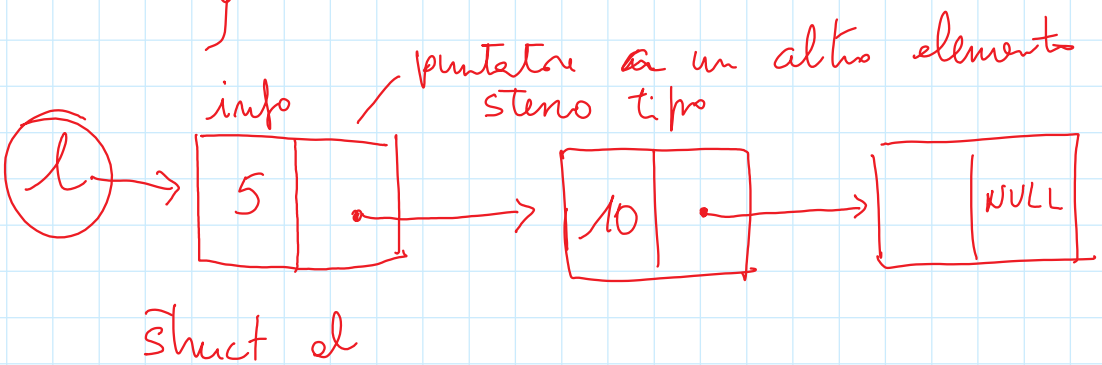
mercoledì 8 novembre 2017 11:14

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



Dare un nome (che si ricordi facilmente) a un tipo struct el

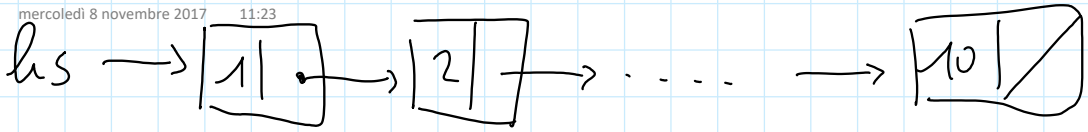
```
typedef  <tipo>  <nome>
```

```
typedef  struct el  ElementoDiListe;
```

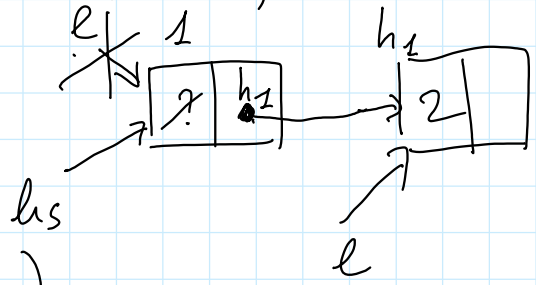
tipo: struct el — nome: ElementoDiListe

```
typedef  ElementoDiListe *  ListeDiElementi;
```

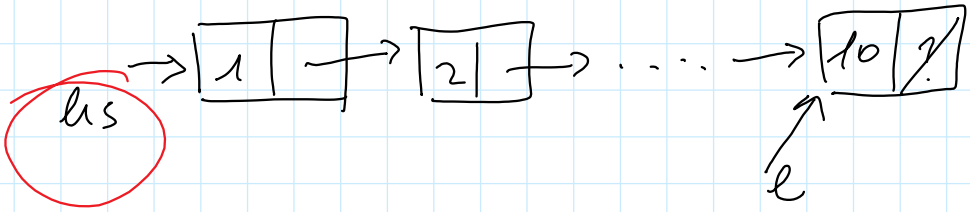
tipo: ElementoDiListe * — nome: ListeDiElementi




```
struct el { int info; struct el * next; }
typedef struct el ElementoDiLista;
typedef ElementoDiLista * ListeDiElementi;

main ()
{ ListeDiElementi l = malloc (sizeof (ElementoDiLista));
  ListeDiElementi lis = l; int i = 2;
  lis -> info = 1;
```

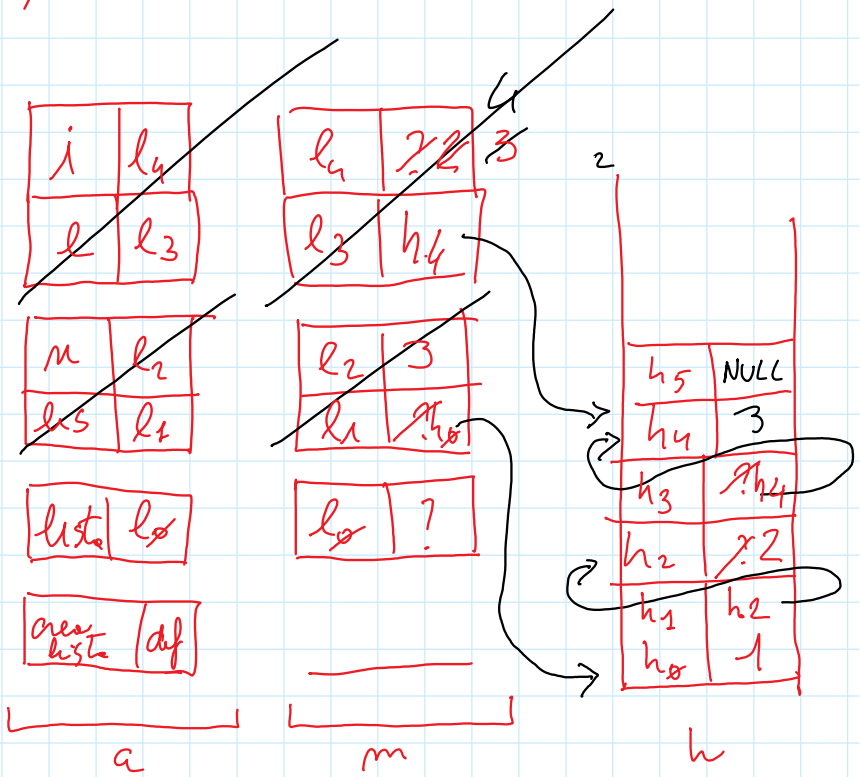



```
  for (i = 2; i <= 10; i++)
  { l -> next = malloc (sizeof (ElementoDiLista));
    l = l -> next;
    l -> info = i;
  }
```




```
  l -> next = NULL;
```

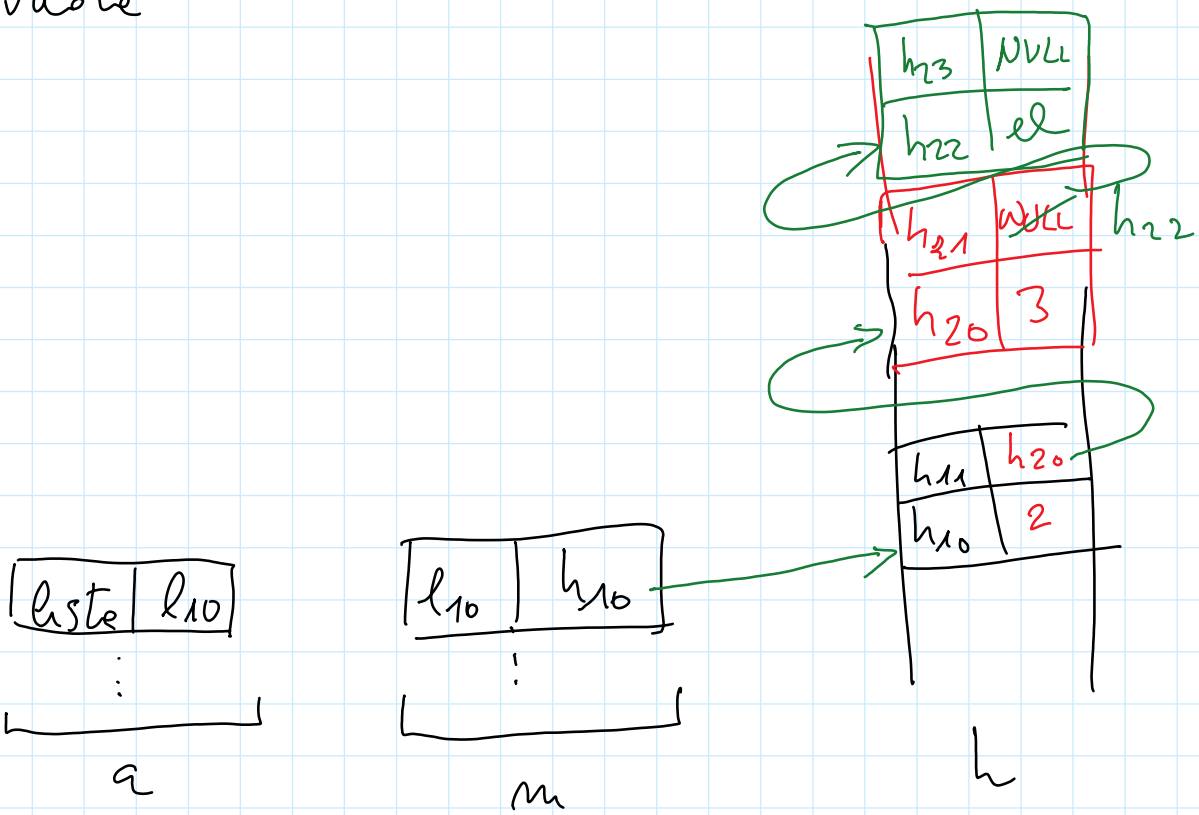
```
void creaLista (ListaDiElementi lis, int n)
{ ListaDiElementi l = malloc ( ... ); int i;
  lis = l;
  lis -> info = 1;
  for (i=2; i<=n; i++)
  { l -> next = malloc ( ... );
    l = l -> next;
    l -> info = i;
  }
  l -> next = NULL;
}

main ()
{ ListaDiElementi lista;
  creaLista ( lista, 3);
```



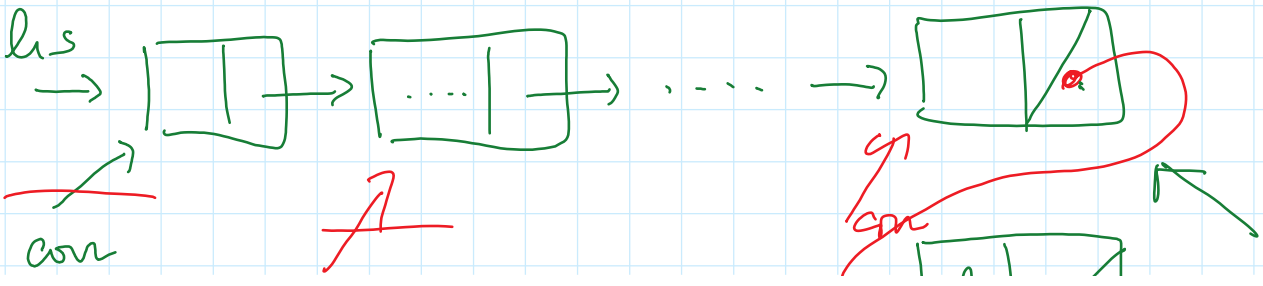
```
void creaLista (ListaDiElementi * lis, int n)
{ ListaDiElementi l = malloc(...); int i;
  *lis = l;
  l -> info = 1;
  for (i=2; i<=n; i++)
  { l -> next = malloc(...);
    l = l -> next;
    l -> info = i;
  }
  l -> next = NULL;
}

main()
{ ListaDiElementi lista;
  creaLista (&lista, 3);
```

mercoledì 8 novembre 2017 11:58

Scrivere una procedura che aggiunge un valore intero el in fondo a una lista non vuota



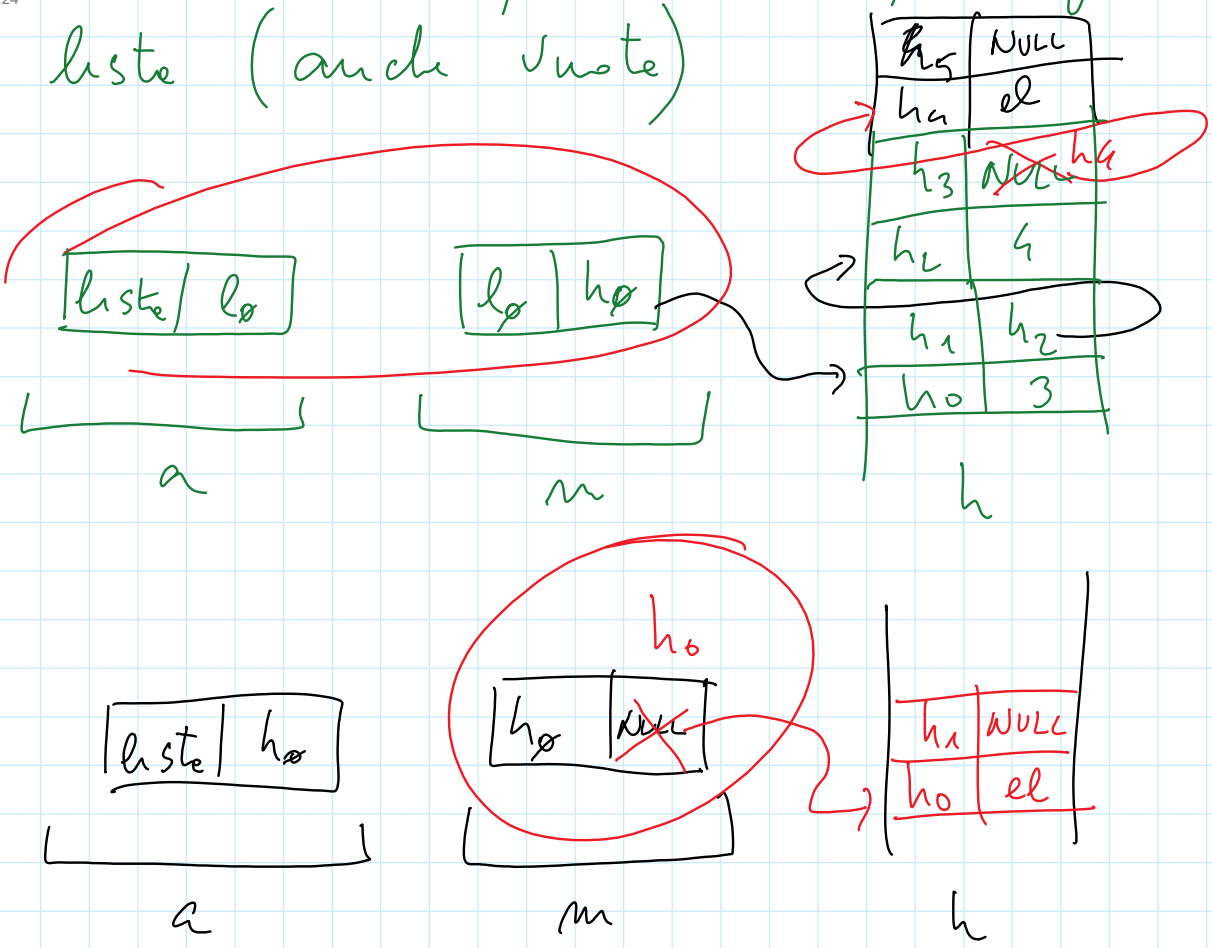

```
void aodd ( ListaDiElementi lis, int el)
{ ListaDiElementi cor = lis;
  ListaDiElementi new = malloc( .... );
  new -> info = el;
  new -> next = NULL;
```




```
while (curr -> next != NULL)
        curr = curr -> next;

curr -> next = new;
}
```

mercoledì 8 novembre 2017 12:24

Aggiungere un elemento, con valore el, in fondo a una lista (anche vuota)



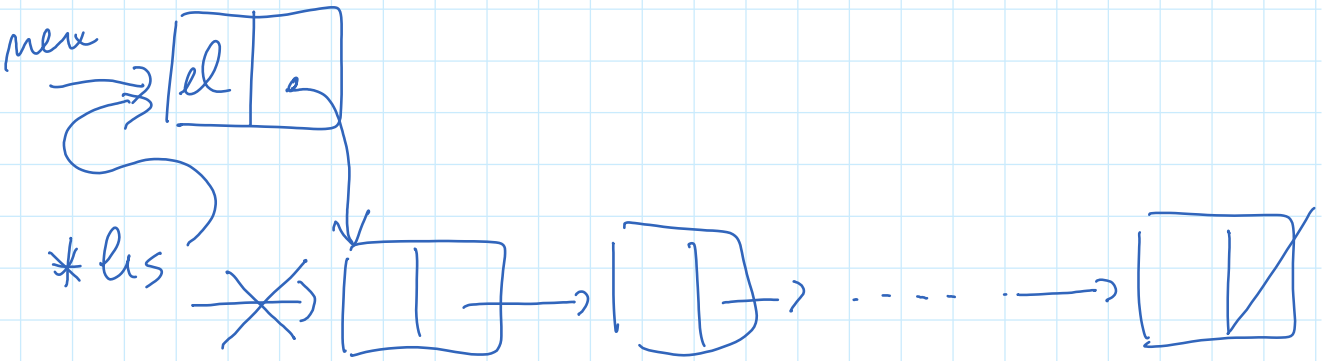

```
void addel (ListeDiElement *lis, int el)
{ ListeDiElement new = malloc(....);
  new -> info = el;
  new -> next = NULL;
  if (*lis == NULL)  *lis = new;
  else
     { ListeDiElement corr = *lis;
       while (corr -> next != NULL)
            corr = corr -> next;
```

```
        curr->next = new;
    }
}
```

mercoledì 8 novembre 2017 12:33

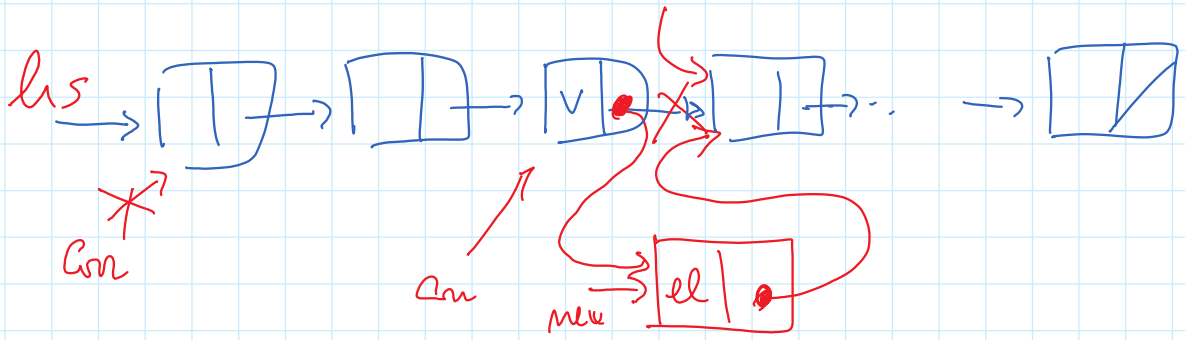
Aggiungere un elemento el in testa alla lista

```
void add (ListaDiElementi * lis, int el)
{ ListaDiElementi new = malloc(....);
  new -> info = el;
  new -> next = *lis;
  *lis = new;
}
```

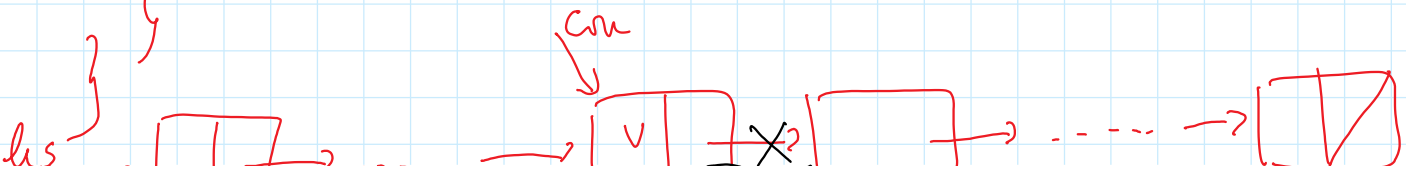


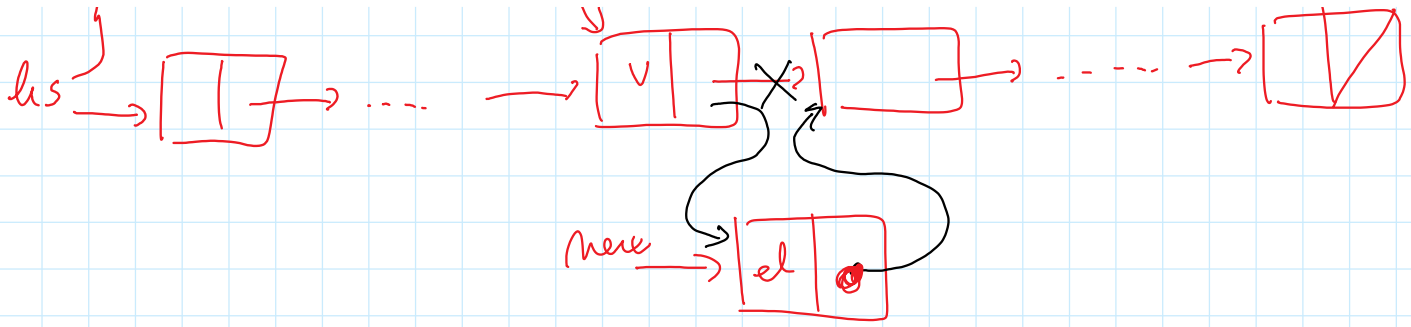
mercoledì 8 novembre 2017 12:38

Aggiungere un elemento che contiene il valore el dopo la prima occorrenza di un elemento che contiene il valore V.
Se v non compare le liste rimane immutate.



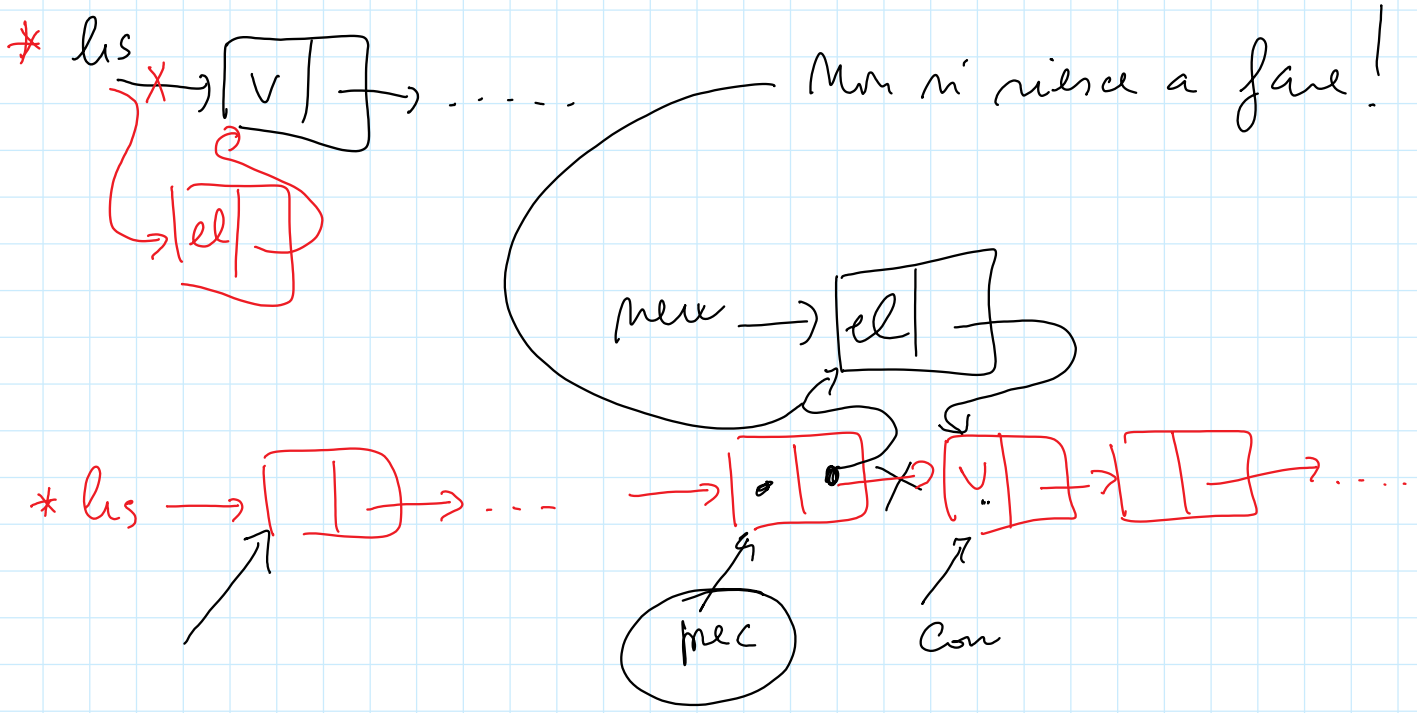

```
void add (ListaDiElementi lis, int el, int v)
{ listaDiElementi curr = lis;
  int trovato = 0;
  while (curr != NULL && !trovato)
      if (curr->info == v) trovato = 1;
      else curr = curr->next;

  if (trovato)
  listaDiElementi new = malloc(...);
  new->next = curr->next;
  curr->next = new;
  new->info = el;
}
```




lis
v
new
el

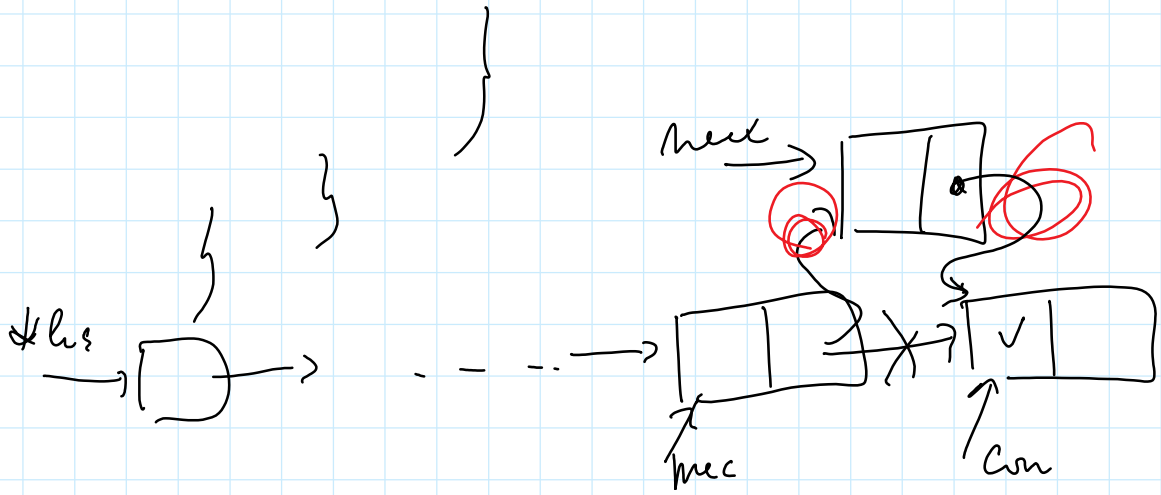
mercoledì 8 novembre 2017 12:54

Aggiungere un elemento el, primo della prima occorrenza di v.



mercoledì 8 novembre 2017 12:58

```
void aodd (ListeDiElement * lis, int el, int v)
{ int trovato = 0;
  ListeDiElement cor = *lis;
  ListeDiElement prec = NULL;
  while (cor != NULL && !trovato)
     if ( cor -> info == v ) trovato = 1;
      else { prec = cor;
             cor = cor -> nex;
           }
  if (trovato)
    { ListeDiElement new = malloc( ... );
      new -> info = el;
      if (prec == NULL)
            { new -> next = *lis;
              *lis = new;
            }
      else { new -> next = cor;
             prec -> next = new;
```


*lis
mec
con