Liste semplici – esercitazione

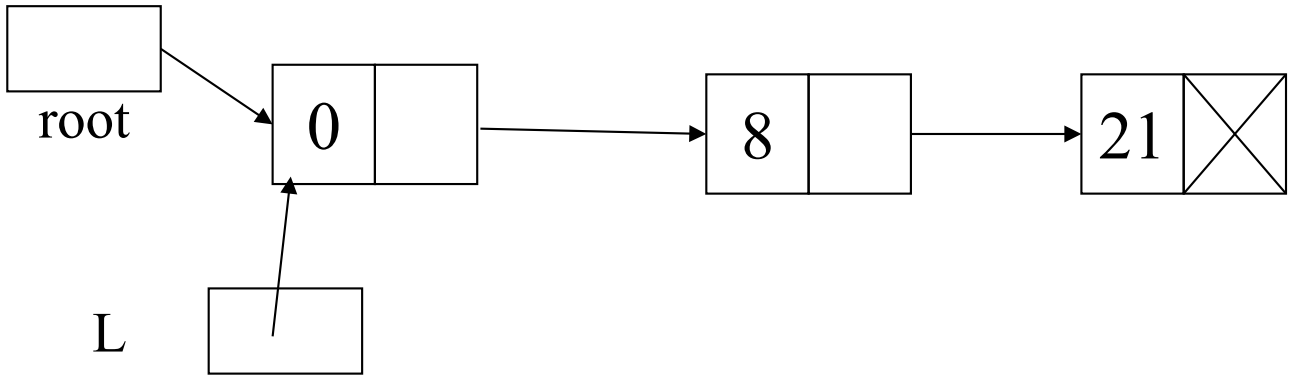
Obiettivi:

- Sperimentare la realizzazione collegata (puntatori a strutture) per la lista semplice
- Definire una funzione (cons) per la creazione di liste
- Definire funzioni di elaborazione sequenziale su liste, iterative e ricorsive (stampa)

Creazione di una lista

Il ***main*** da realizzare deve leggere la sequenza di interi (terminata con 0) e inserire ogni elemento letto in testa ad una una ***lista***

Per ogni inserimento, **`malloc`** e aggiustamento dei puntatori



root inizialmente NULL

Creazione di una lista semplice

```
#include <stdlib.h>
#include <stdio.h>
typedef struct list_element { int value;
        struct list_element *next; } item;
typedef item *list;
main() { list L; int i;
   list root = NULL;
   do { printf("\nIntrodurre valore: \t");
        scanf("%d", &i);
        L = (list) malloc(sizeof(item));
```

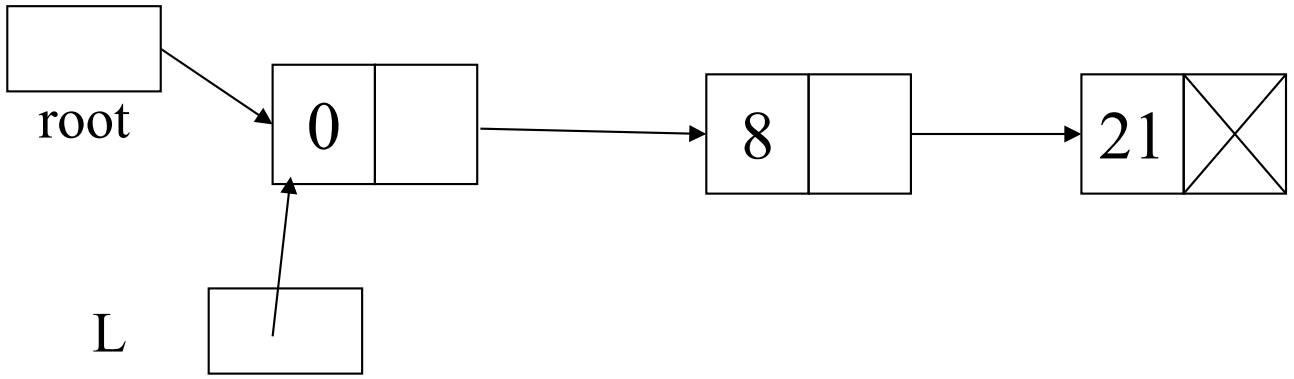
Inserisci **i** nella lista puntata da **root**

```
/*stampa della lista/*
}
```

Stampa a video di una lista

Il ***main*** da realizzare deve leggere la sequenza di interi (terminata da 0) e inserire ogni elemento letto in una ***lista***

Poi stampiamo la sequenza …
(facciamo insieme anche questo secondo passo)



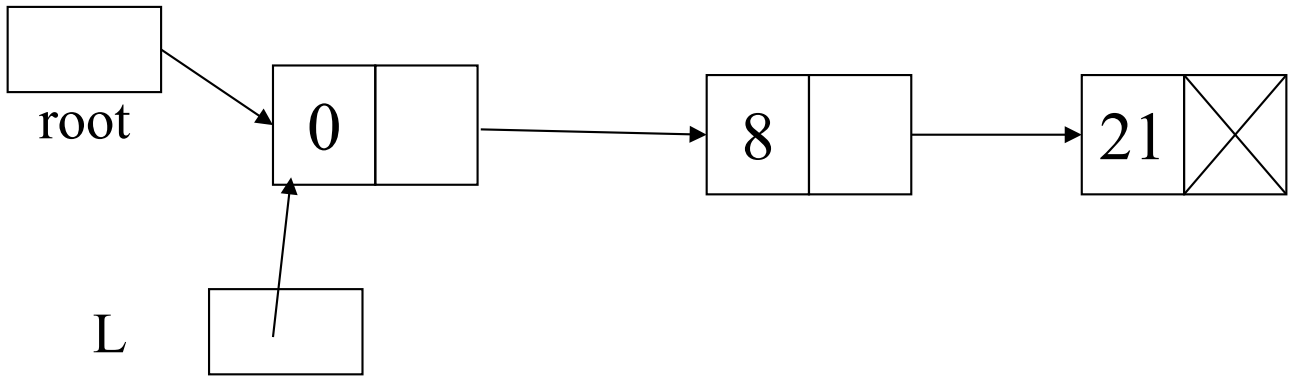
Non c'è un "indice" da incrementare come per vettori, ma va aggiornato un puntatore con cui si scorre la lista

Creazione di una lista semplice

```
#include <stdlib.h>
#include <stdio.h>
typedef struct list_element { int value;
        struct list_element *next; } item;
typedef item *list;

main() { list L; int i;
  list root = NULL;
  do { printf("\nIntrodurre valore: \t");
       scanf("%d", &i);
       L = (list) malloc(sizeof(item));
       L->value = i;
       L->next = root;
       root = L;
     } while (i!=0);
/* stampa della lista */  }
```

Stampa di una lista di interi (segue)

```
L = root;

while (L!=NULL) {
 printf("\nValore estratto: \t%d", L->value);
 L = L->next; }
}
```



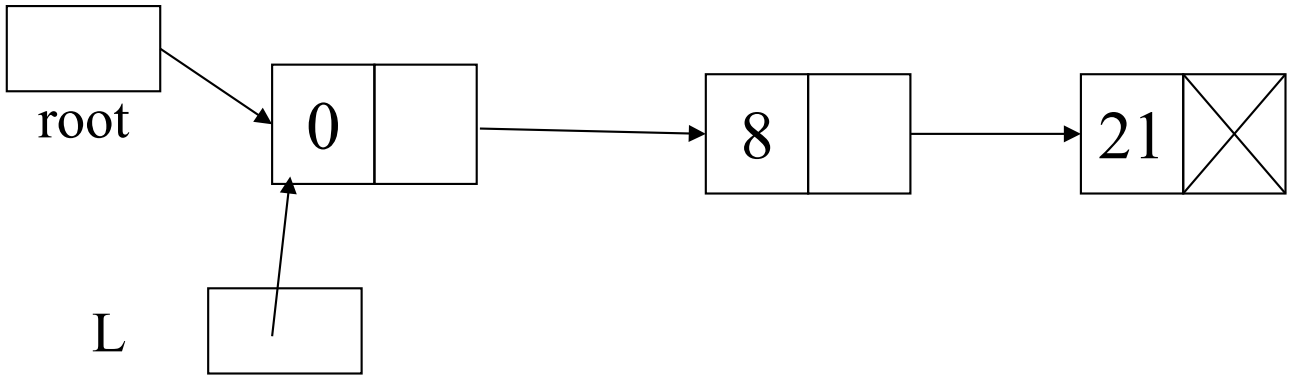
Per "non perdere" il puntatore al primo elemento della lista, la lista va scandita con un puntatore ausiliario (ad esempio L)

FUNZIONI cons, showlist

Prepariamo opportune funzioni di inserimento in testa alla lista e di stampa della lista, e riscriviamo il main invocandole

```
list cons(int, list);
void showlist (list);
```

Prepariamo **anche una versione ricorsiva** per la stampa della lista

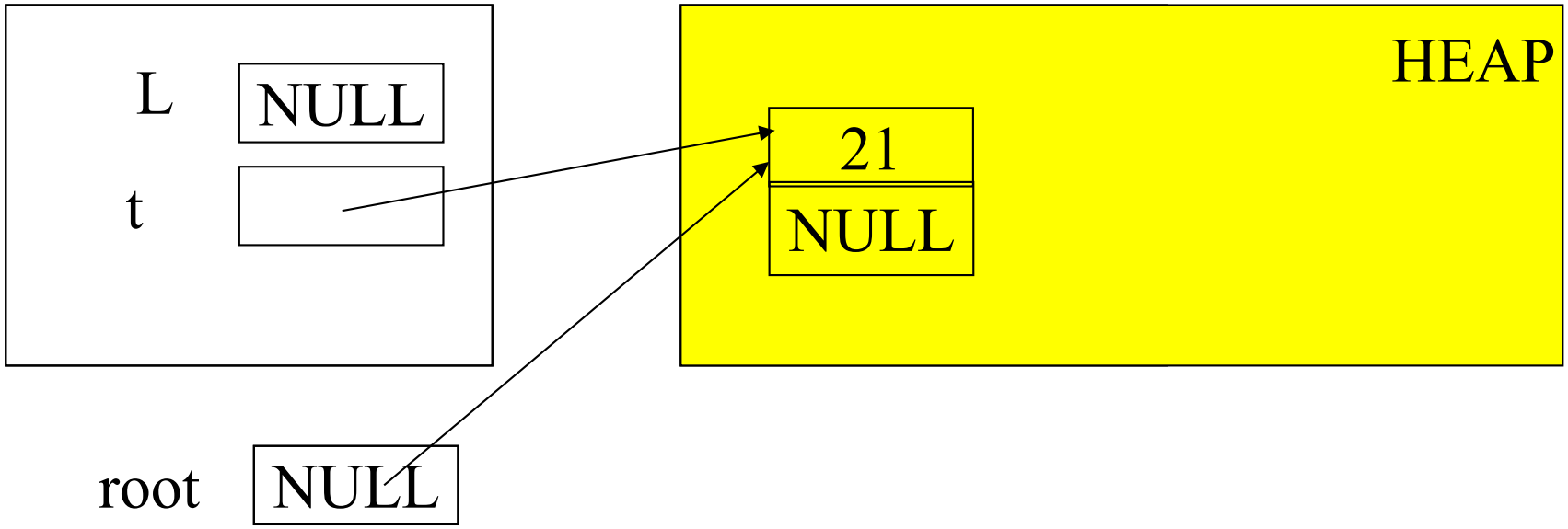


Creazione di una lista semplice

```
#include <stdlib.h>
#include <stdio.h>
typedef struct list_element
    { int value;
        struct list_element *next; } item;
typedef item *list;

main() { list L; int i;
  list root = NULL;
  do { printf("\nIntrodurre valore: \t");
       scanf("%d", &i);

       root = cons( i, root);

/*stampa lista*/
showlist(root);}
```

Inserimento in testa: *cons*

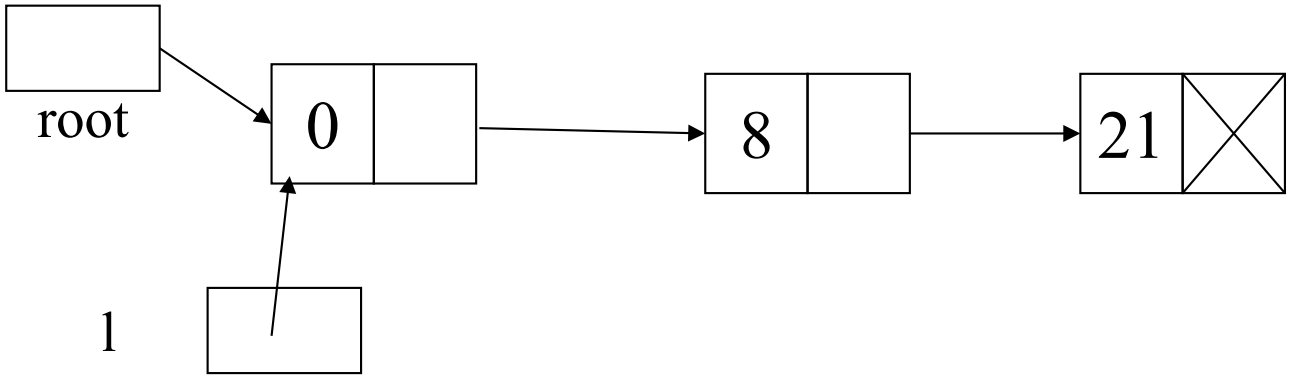
```
list cons(int e, list L) {
      list t;
      t = (list) malloc(sizeof(item));
      t->value = e; t->next = L;
      return t; }
```



Stampa di una lista: `showList`

Stampa di una lista: la lista va ***scandita (sequenzialmente) con un puntatore:***

```
void showList(list l) {
      while ( l!=NULL )
            {   printf("%d", l->value);
                l=l->next;  }
}
```



Stampa di una lista: `showListr` ricorsiva

… oppure ***versione ricorsiva***

Stampa di una lista (ricorsiva):

```
void showListr(list l) {
      if( l!=NULL )
            { printf("%d", l->value);
              showListr( l->next ); }
}
```