

# List Example: list.h

---

## ◆ list.h

```
#ifndef _LIST_H
#define _LIST_H

struct listitem {
    int data;
    struct listitem *next;
};

typedef struct listitem Listitem;

struct list {
    Listitem *head;
};

typedef struct list List;

void initlist (List *); /* initialize an empty list
void insertfront(List *, int val); /* insert val at front */
void insertback(List *, int val); /* insert val at back */
int length(List); /* returns list length */
void destroy(List *); /* deletes list */
void setitem(List *, int n, int val); /* modifies item at n to v
int getitem(List, int n); /* returns value at n*/

#endif /* _LIST_H */
```

# List Example: list.c

---

## ◆ list.c

```
#include "list.h"
```

```
void initlist(List *ilist) {  
    ilist->head = 0;  
}
```

```
void insertfront(List *ilist, int val) {  
    Listitem *newitem;  
    newitem = (Listitem *)malloc(sizeof(Listitem));  
    newitem->next = ilist->head;  
    newitem->data = val;  
    ilist->head = newitem;  
}
```

# List Example: list.c continued

---

## ◆ list.c continued

```
void insertback(List *ilist, int val) {
    Listitem *ptr;
    Listitem *newitem;

    newitem = (Listitem *)malloc(sizeof(Listitem));
    newitem->data = val;
    newitem->next = 0;

    if (!ilist->head) {
        ilist->head = newitem;
        return;
    }

    ptr = ilist->head;
    while (ptr->next)
    {
        ptr = ptr->next;
    }
    ptr->next = newitem;
}
```

# List Example:list.c continued

---

## ◆ list.c continued

```
int length(List ilist){          /* returns list length */
    Listitem *ptr;
    int count = 1;

    if (!ilist.head) return 0;
    ptr = ilist.head;
    while (ptr->next) {
        ptr = ptr->next;
        count++;
    }
    return count;
}
```

# List Example:list.c continued

---

## ◆ list.c continued

```
void destroy(List *ilist) {           /* deletes list */
    Listitem *ptr1,*ptr2;

    if (!ilist->head) return; /* nothing to destroy */

    ptr1 = ilist->head;             /* destroy one by one */
    while (ptr1) {
        ptr2 = ptr1;
        ptr1 = ptr1->next;
        free(ptr2);
    }
    ilist->head = 0;
}
```

# List Example:list.c continued

---

## ◆ list.c continued

```
void setitem(List *ilist, int n, int val){
    /* modifies a value*/
    /* assume length is at least n long */
    Listitem *ptr;
    int count = 0;

    if (!ilist->head) return;
    ptr = ilist->head;
    for (count = 0;count < n;count ++)
    {
        if (ptr) ptr = ptr->next;
        else return;
    }
    if (ptr)
        ptr->data = val;
}
```

# List Example:list.c continued

---

## ◆ list.c continued

```
int getitem(List ilist, int n) {
    /* returns a list value,
     * assume length is at least n long */
    Listitem *ptr;
    int count = 0;

    if (!ilist.head) return 0;
    ptr = ilist.head;
    if (n==0) return ptr->data;
    while (ptr->next) {
        ptr = ptr->next;
        count++;
        if (n == count)
            return (ptr->data);
    }
    return 0;
}
```

# List Example: main program

---

## ◆ main.c

```
#include "list.h"

main ()
{
    List mylist;

    initlist(&mylist);
    printf("L=%d\n",length(mylist));
    insertback(&mylist,1);
    printf("L=%d\n",length(mylist));
    insertback(&mylist,2);
    printf("L=%d\n",length(mylist));
    insertback(&mylist,3);
    printf("L=%d\n",length(mylist));
    insertback(&mylist,4);
    printf("L=%d\n",length(mylist));
    printf("item 0 = %d\n",getitem(mylist,0));
    printf("item 1 = %d\n",getitem(mylist,1));
    printf("item 2 = %d\n",getitem(mylist,2));
    printf("item 3 = %d\n",getitem(mylist,3));
    destroy(&mylist);
}
```



# List Example: output of main program

---

- ◆ output of main.c

```
L=0
```

```
L=1
```

```
L=2
```

```
L=3
```

```
L=4
```

```
item 0 = 1
```

```
item 1 = 2
```

```
item 2 = 3
```

```
item 3 = 4
```