1 一般删除单链表中元素的方法

**typedef** **struct** node

{

struct node \* next;

....

} node;

**typedef** **bool** (\* remove\_fn)(node **const** \* v);

// Remove all nodes from the supplied list for which the

// supplied remove function returns true.

// Returns the new head of the list.

node \* remove\_if(node \* head, remove\_fn rm)

{

**for** (node \* prev = NULL, \* curr = head; curr != NULL; )

{

node \* const next = curr->next;

**if** (rm(curr))

{

if (prev)

prev->next = next;

else

head = next;

free(curr);

}

**else**

prev = curr;

curr = next;

}

**return** head;

}

2 利用二级指针删除单链表元素

**void** remove\_if(node \*\* head, remove\_fn rm)

{

**for** (node\*\* curr = head; \*curr; )

{

node \* entry = \*curr;

if (rm(entry))

{

\*curr = entry->next;

free(entry);

}

**else**

curr = &entry->next;

}

}

The key insight is that the links in a linked list are pointers and so **pointers to pointers** are the prime candidates for modifying such a list.