```
#define NULL 0
typedef struct node{
	node *next;
	int value;
};

//反向一遍判断最后是 null 还是 head，在反向回来
bool IsExistLoop(node* head)
{
	node *prev , *now , *next;
	bool re;
	prev = NULL;
	now = head;
	next = (*head) . next;
	while( next != head && next != NULL){
		(*now) . next = prev;
		prev = now;
		now = next;
		next = (*now) . next;
	}
	re = next == head;
	node *t;
	t = now;
	now = prev;
	prev = t;
	next = (*head) . next;
	while( now != NULL){
		(*now) . next = prev;
		prev = now;
		now = next;
		next = (*now) . next;
	}
	return re;
}

//把每个节点的 next 指向他的前一个节点
node * reverse(node* head)
{
	node *prev , *now , *next;
	prev = NULL;
	now = head;
	next = (*head) . next;
	while( now != NULL){
		(*now) . next = prev;
```

```
            prev = now;
            now = next;
            next = (*now) . next;
        }
        return prev;
}

//找到环的开头，然后将绕一圈后指向他的点指向 NULL
node * unloop(node * head)
{
        if(!IsExistLoop(head))return head;
        node *prev , *now , *next;
        prev = NULL;
        now = head;
        next = (*head) . next;
        (*now) . next = NULL;
        while(IsExistLoop(next)){
            (*now) . next = next;
            prev = now;
            now = next;
            next = (*now) . next;
            (*now) . next = NULL;
        }
        (*now) . next = next;
        node *t;
        t = now;
        while((*t) . next != now)t = (*t) . next;
        (*t) . next = NULL;
        return head;
}

//判断该节点是否在环内，如果是则删除，如果不是则返回 false
bool delete_internode(node * target_node)
{
        if(!IsExistLoop(target_node))return false;
        node *prev , *now , *next;
        prev = NULL;
        now = target_node;
        next = (*target_node) . next;
        (*now) . next = NULL;
        bool re=true;
        if(IsExistLoop(next))re = false;
        (*now) . next = next;
        if(!re)return re;
```

```
        node *t;
        while((*t) . next != now)t = (*t) . next;
        (*t) . next = (*now) . next;
        delete now;
        return true;
}

//若有环，则找到环的开头，然后判断是否在同一个环里；若无环则看看末尾是否相同
bool detect_internode(node * first_link , node * second_link)
{
        bool l1 = IsExistLoop(first_link) , l2 = IsExistLoop(second_link);
        if(l1 ^ l2) return false;
        if(l1){
                node *t1 , *t2;
                node *prev , *now , *next;
                prev = NULL;
                now = first_link;
                next = (*first_link) . next;
                (*now) . next = NULL;
                while(IsExistLoop(next)){
                        (*now) . next = next;
                        prev = now;
                        now = next;
                        next = (*now) . next;
                        (*now) . next = NULL;
                }
                (*now) . next = next;
                t1 = now;
                prev = NULL;
                now = second_link;
                next = (*second_link) . next;
                (*now) . next = NULL;
                while(IsExistLoop(next)){
                        (*now) . next = next;
                        prev = now;
                        now = next;
                        next = (*now) . next;
                        (*now) . next = NULL;
                }
                (*now) . next = next;
                t2 = now;
                while((*now) . next != t2 && (*now) . next != t1)now = (*now) . next;
                if((*now) . next == t1)return true;
                else return false;
```

```
        }
        else{
                node *t1 , *t2;
                t1 = first_link;
                t2 = second_link;
                while((*t1) . next != NULL)t1 = (*t1) . next;
                while((*t2) . next != NULL)t2 = (*t2) . next;
                if(t1 == t2)return true;
                else return false;
        }
}

//当当前节点与下一个节点断开后仍然成立的节点中第一个节点
node* first_internode(node* first_link , node* second_link)
{
        if(!detect_internode(first_link , second_link))return first_link;
        node *prev , *now , *next;
        prev = NULL;
        now = first_link;
        next = (*first_link) . next;
        (*now) . next = NULL;
        while(!detect_internode(now , second_link)){
                (*now) . next = next;
                prev = now;
                now = next;
                next = (*now) . next;
                (*now) . next = NULL;
        }
        (*now) . next = next;
        return now;
}

//调试用的主程序
int main()
{
        return 0;
}
```