GeekBand 极客班

互联网人才 十 油站!

链表

www.geekband.com

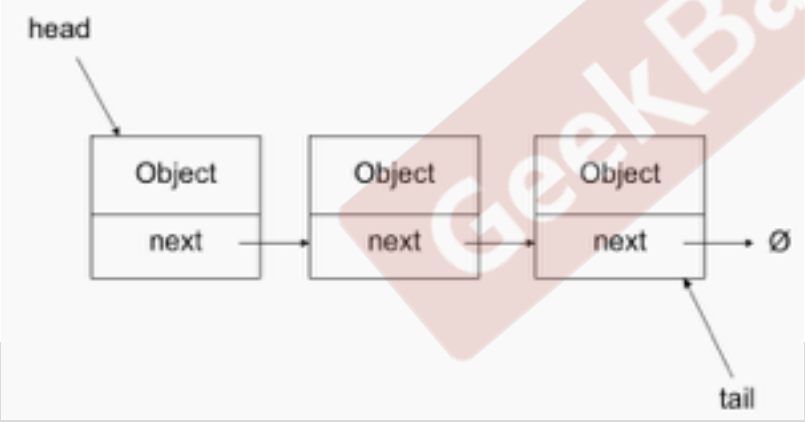
2 链表

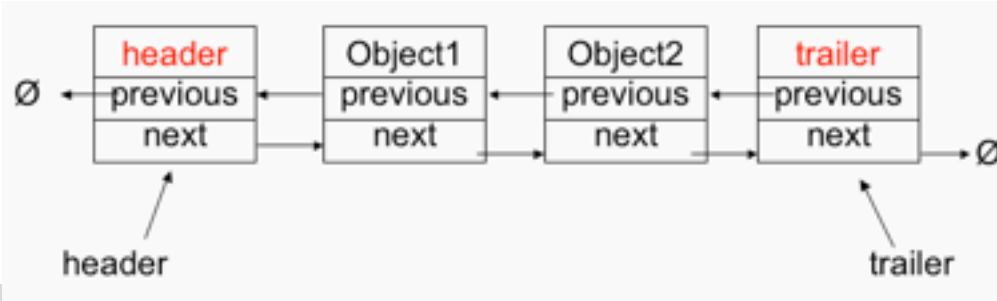
Outline

1. Introduce Dummy Node in Linked List
2. Basic skills in Linked List you should know
3. Two pointers
4. Frequent Questions

链表介绍

对于单向链表(singly linked list)，每个节点有一个next指针指向后一个节点，还有一个成员变量用以储存数值；对于双向链表(Doubly Linked List)，还有一个prev指针指向前一个节点。与数组类似，搜索链表需要O(n)的时间复杂度，但是链表不能通过常数时间读取第k个数据。链表的优势在于能够以较高的效率在任意位置插入或删除一个节点。





注意

a. 哪个节点的next指针会受到影响，则需要修正该指针
b. 如果待删除节点是动态开辟的内存空间，则需要释放这部分空间(C/C++)

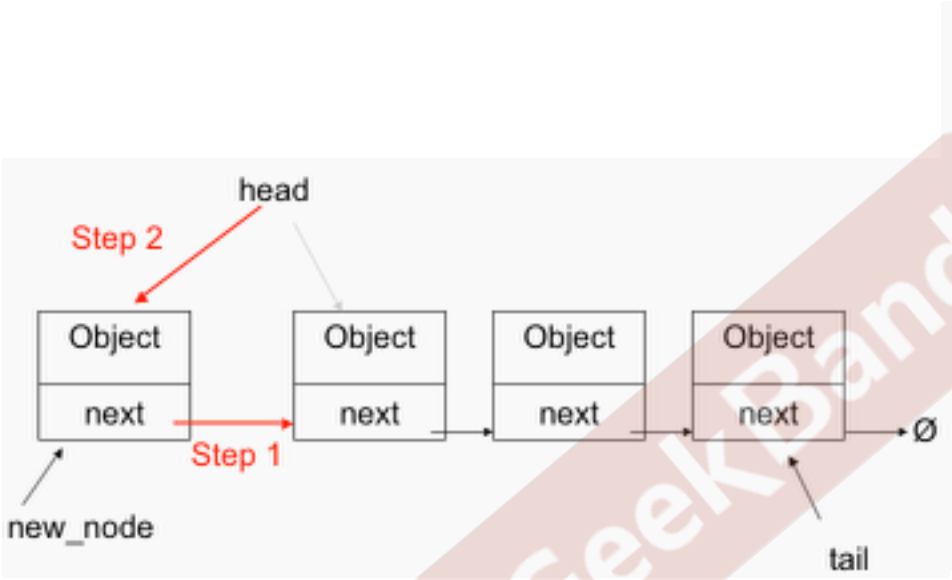
链表操作时利用dummy node是一个非常好用的trick：只要涉及操作head节点，不妨创建dummy node:

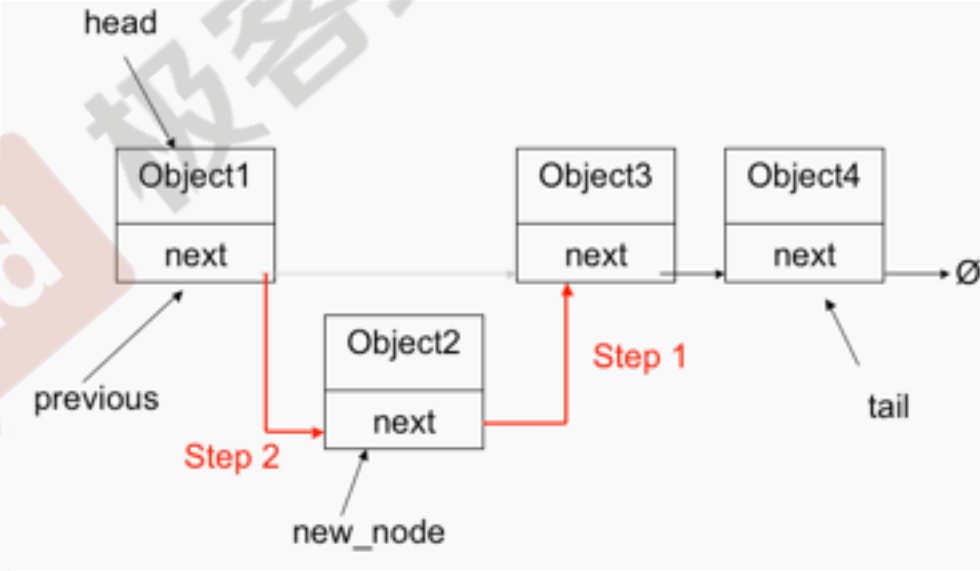
```
ListNode *dummy = new ListNode(0);
dummy->next = head;
```

查找

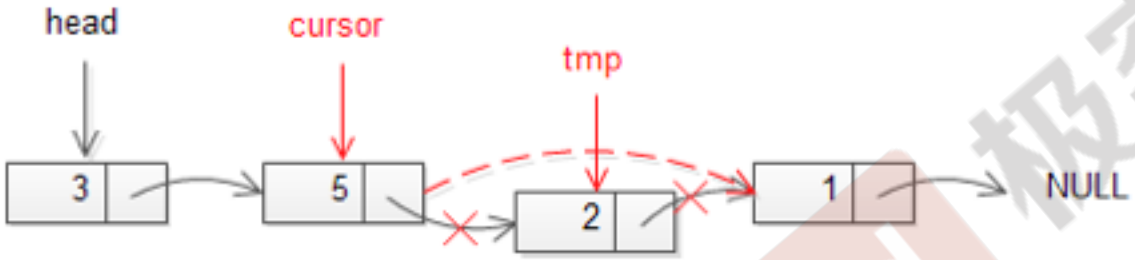
```
current = head
//for each item in the list
while(current != null)
        //if the data matches the target
        if(target.equals(current.getData())
                return true
        //advance current
        current = current.getNext()
return false
```

增加节点



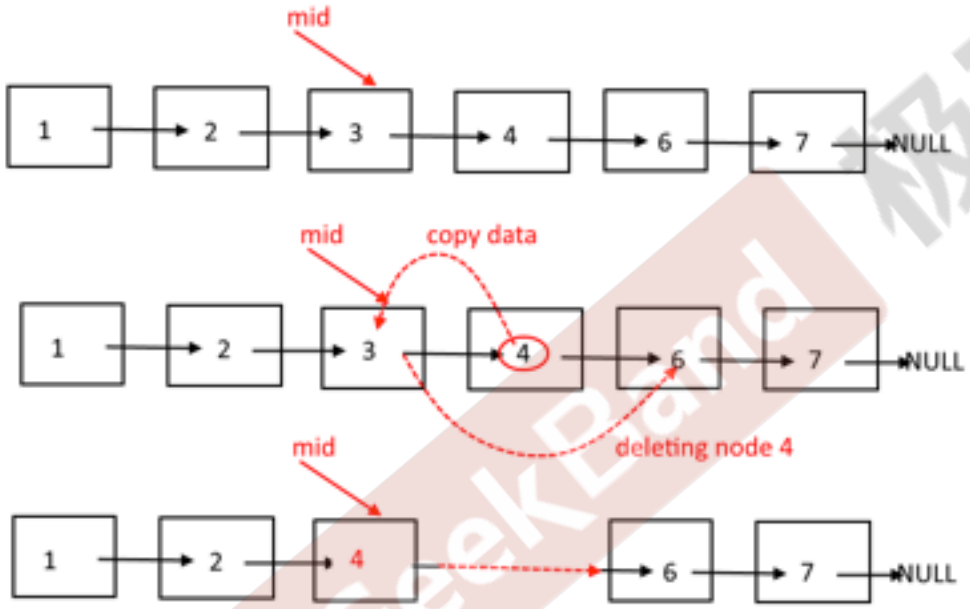


删除




```
void delNode(ListNode *prev) {
  ListNode *curr = prev->next;
  prev->next = curr->next;    // 删除curr节点只会使prev节点的next
受到影响
  delete curr;    // 清理trash指针
}
```

注：操作Linked List时务必注意边界条件：curr == head, curr == tail 或者 curr == NULL


mid
1
2
3
4
6
7
NULL
mid
copy data
1
2
3
4
6
7
NULL
deleting node 4
mid
1
2
4
6
7
NULL

Dummy Node

Scenario: When the head is not determinated

1. Remove Duplicates from Sorted List I, II
2. Merge Two Sorted Lists
3. Partition List
4. Reverse Linked List I,II

Remove Duplicates from Sorted List

Given a sorted linked list, delete all duplicates such that each element appear only once. For example, Given 1->1->2, return 1->2. Given 1->1->2->3->3, return 1->2->3.

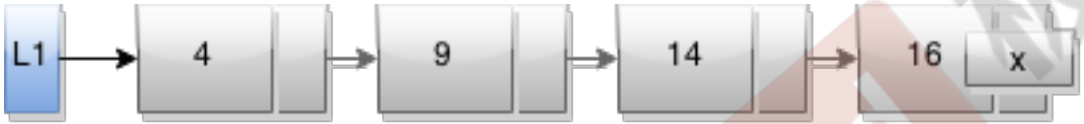
Remove Duplicates from Sorted List II

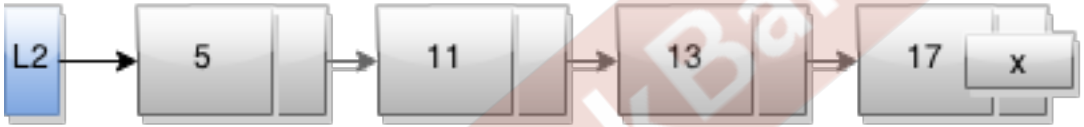
Given a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list.

For example,
Given 1->2->3->3->4->4->5, return 1->2->5.
Given 1->1->1->2->3, return 2->3.

Merge Two Sorted List

Merge two sorted linked lists and return it as a new list. The new list should be made by splicing together the nodes of the first two lists.





Reorder List

Given a linked list and a value x, write a function to reorder this list such that all nodes less than x come before the nodes greater than or equal to x.

Basic Skills

1. Insert a Node in Sorted List
2. Remove a Node from Linked List
3. Reverse a Linked List
4. Merge Two Linked Lists
5. Find the Middle of a Linked List

Sort List

Sort a linked list in O(n log n) time using constant space complexity.
Hint: merge sort

MergeSort(arr[], l, r)
If r > l

1. *Find the middle point to divide the array into two halves:*
   middle m = (l+r)/2
2. *Call mergeSort for first half:*
   Call mergeSort(arr, l, m)
3. *Call mergeSort for second half:*
   Call mergeSort(arr, m+1, r)
4. *Merge the two halves sorted in step 2 and 3:*
   Call merge(arr, l, m, r)

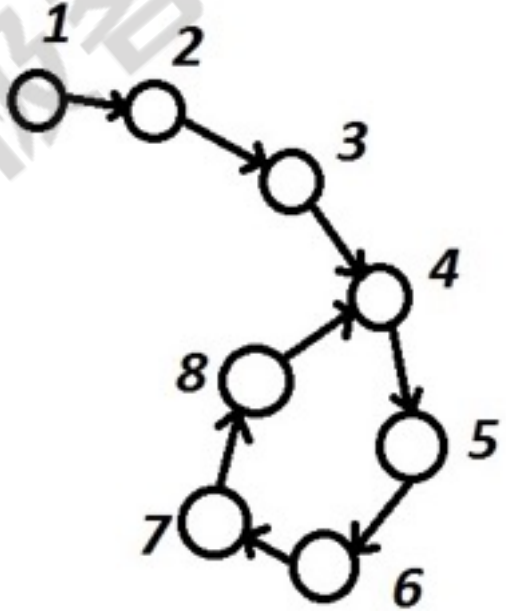
Two Pointers

1. Linked List Cycle I, II
2. Remove/Find Nth Node From End of List
3. Find the Middle of Linked List

Linked List Cycle

Given a linked list, determine if it has a cycle in it.

如何判断一个单链表中有环?

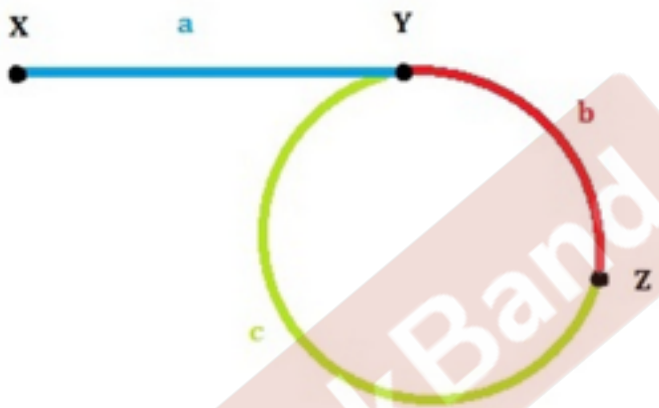


Linked List Cycle 2

Follow up: Can you solve it without using extra space?

Given a linked list, return the node where the cycle begins. If there is no cycle, return null.

如何找到环的第一个节点?



```
public static ListNode detectCycle(ListNode head) {
    ListNode slow = head;
    ListNode fast = head;

    while (true) {
        if (fast == null || fast.next == null) {
            return null;    //遇到null了，说明不存在环
        }
        slow = slow.next;
        fast = fast.next.next;
        if (fast == slow) {
            break;    //第一次相遇在Z点
        }
    }
    slow = head;    //slow从头开始走，
    while (slow != fast) {    //二者相遇在Y点，则退出
        slow = slow.next;
        fast = fast.next;
    }
    return slow;
}
```

4. 如何判断两个单链表是否有交点？先判断两个链表是否有环，如果一个有环一个没环，肯定不相交；如果两个都没有环，判断两个列表的尾部是否相等；如果两个都有环，判断一个链表上的Z点是否在另一个链表上。

如何找到第一个相交的节点？求出两个链表的长度L1,L2（如果有环，则将Y点当做尾节点来算），假设L1<L2，用两个指针分别从两个链表的头部开始走，长度为L2的链表先走(L2-L1，然后两个一起走，直到二者相遇。

Frequent Questions

Linked List

Reverse Linked List

Reverse the linked list and return the new head.

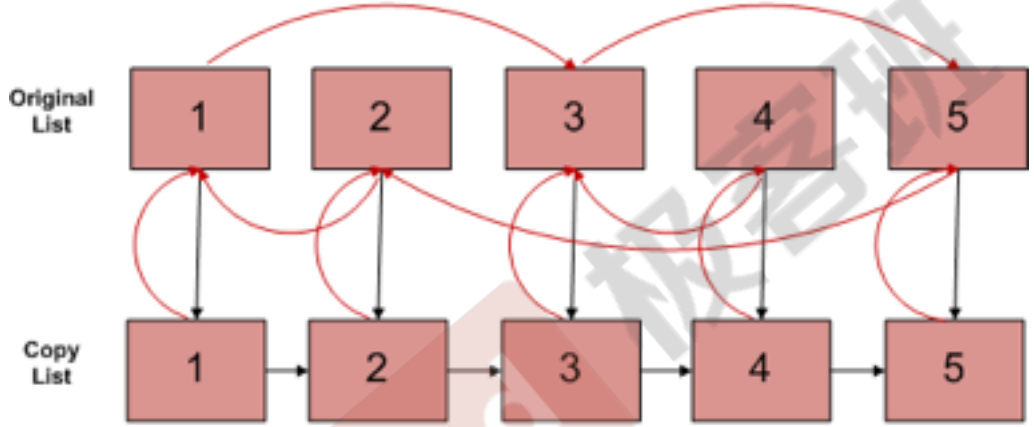
Merge k Sorted Lists

Merge k sorted linked lists and return it as one sorted list.

Copy List with Random Pointer

A linked list is given such that each node contains an additional random pointer which could point to any node in the list or null.

Return a deep copy of the list.



Homework

Given two linked lists, each element of the lists is a integer. Write a function to return a new list, which is the "sum" of the given two lists.
Part a. Given input (7->1->6) + (5->9->2), output 2->1->9.
Part b. Given input (6->1->7) + (2->9->5), output 9->1->2.