**面试大总结之链表**

一、OverView:  
链表是面试中常考的，本文参考了其它一些文章，加上小编的自己总结，基本每个算法都测试并优化过。  
[算法大全（1）单链表](http://www.cnblogs.com/jax/archive/2009/12/11/1621504.html) 中还有一些链表题目，将来也会整理进来。这些题目虽然简单，但如果能毫无BUG地写出，定能让面试官司对您印象分大增。  
  
小亮点是：主页君用Recursion 和 Iterator 各写了一次所有题目，这样就算遇到不熟悉的写法，我们也都可以运用自如。  
  
**二、代码**  
以下是GitHub的代码链接，欢迎大家指正：

[**ListNodeDemo.java**](https://github.com/yuzhangcmu/LeetCode_algorithm/blob/master/ListNodeDemo.java)

**三、代码目录：**  
  1. 求单链表中结点的个数:  
      getListLength  
  2. 将单链表反转:  
      reverseList（遍历），reverseListRec（递归）  
  3. 查找单链表中的倒数第K个节点（k > 0）:  
      reGetKthNode  
  4. 查找单链表的中间结点:  
      getMiddleNode  
  5. 从尾到头打印单链表:  
      reversePrintListStack，reversePrintListRec（递归）  
  6. 已知两个单链表pHead1 和pHead2 各自有序，把它们合并成一个链表依然有序:  
      mergeSortedList, mergeSortedListRec  
  7. 判断一个单链表中是否有环:  
      hasCycle  
  8. 判断两个单链表是否相交:  
      isIntersect  
  9. 求两个单链表相交的第一个节点:  
      getFirstCommonNode  
  10. 已知一个单链表中存在环，求进入环中的第一个节点:  
      detectCycle, getFirstNodeInCycleHashMap  
  11. 给出一单链表头指针pHead和一节点指针pToBeDeleted，O(1)时间复杂度删除节点  
      pToBeDeleted: delete  
   
**四、总结规律：**  
  1. DummyNode的使用。  
做链表题目时，如果我们需要返回头部，一般情况我们可以创建一个虚拟节点，叫DummyNode，把头部挂在它的后面。这样就算头部变化了之后，只要返回DummyNode.next就能轻松得到新头部，而不用纠结新的头部到底 在哪里。  
   2. Merge LinkedList是相当基础的题目，这么多题目是基于它的，必须写熟。  
   3. Reverse linkedList最简单的写法就是创建DummyNode，然后把旧的链表不断插入到DummyNode的后面，就能轻松地返回链表了。  
   4. 操作链表的时候，我们经常会改变某些Node的下一个节点。如果你希望待一下会再用到被改变掉的下一个节点，请一定记得用tmp先把它保存起来。  
   5. 查找链表的中间节点：使用2个快慢指针 ，一个进2步，一个进1步，快指针到达终点时，慢指针就会停留在链表的中间位置了。

Fa