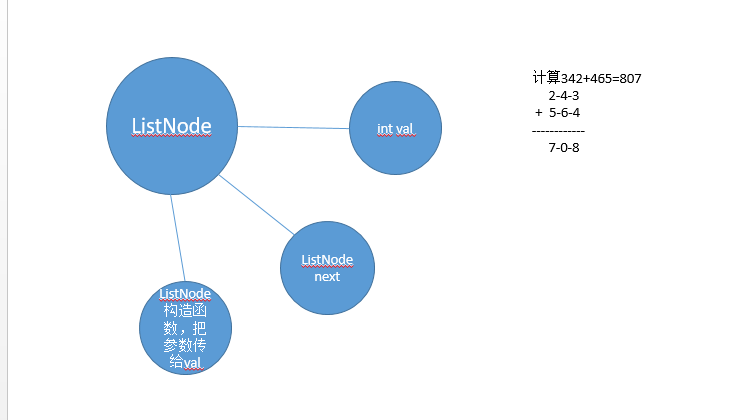
You are given two non-empty liked lists representing two non-negative integers. The digits are stored in reverse order and each of their nodes contain a single digit. Add the two numbers and return it as a linked list.

You may assume the two numbers do not contain any leading zero, except the number 0 itself.

Input:（243）+（564）

Output：708

Explanation：342+465=807.

解析：

下面给的代码是网上一个人写的：有点问题，问题就是只能处理三个位数的相加，四位数相加就不能处理了，应该在那个地方加一个while循环来解决。但是作者竟然通过了leetcode的检验，就很奇怪，没有很好的处理所有情况啊。

代码：//很垃圾的代码，可重用性几乎为零。

class ListNode{

int val;

ListNode next;

ListNode(int x){val = x;}

public String toString(){

return val+"";

}

}

class Solution{

public ListNode addTwoNumbers(ListNode l1;ListNode l2){

if(l1==null&&l2=null)

return null;

if(l1==null&&l2!=null)

return l2;

if(l1!=null&&l2==null)

return l1;

int tmp;

ListNode root = new ListNode(0);

root.next = l2;

while(l1!=null&&l2!=null){//l1 and l2 both unempty.

tmp = l1.val+l2.val;//the first number of the linked lists are ex.

if(tmp>=10&&l1.next!=null){

tmp=tmp-10;

++l1.next.val;

if(l1.next.val>=10&&l1.next.next!=null){

l1.next.val=l1.next.val-10;

++l1.next.next.val;

}

else if(l1.next.val>=10&&l1.next.next==null){

l1.next.val=l1.next.val-10;

l1.next.next=new ListNode(1);

}

}

l2.val=tmp;

if(l2.next==null){

l2.next=l1.next;

l1=l1.next;

break;

}else{

l1=l1.next;

l2=l2.next;

}

}

return root.next;

}

}

public class Add\_twonum\_2{

public static void main(String[] args){

ListNode list11 = new ListNode(8);

ListNode listl2 = new ListNode(9);

ListNode listl3 = new ListNode(9);

ListNode list21 = new ListNode(2);

list11.next=list12;

list12.next=list13;

list13.next=null;

list21.next=null;

Solution ex = new Solution();

ListNode result = ex.addTwoNumbers(listl1;listl21);

while(result!=null){

System.out,print(result+"->");

result = result.next;

}

}

}

知识点：

1. 链表：对于C/C++而言链表就是指针那些东西，但是对于java而言，没有指针，需要自己定义一个数据结构，这个数据结构要能够实现指针实现的那种功能。
2. toString：当创建一个类的时候没有定义toString方法，输出对象时，会输出对象的哈希值。当程序员创建一个类时定义了toString方法，那么就这样用

class solution{

public String toString(){

return "……";

}

}

main函数里就{

solution exp = new solution();

System.out.println(exp);//这种输出方法就是默认的调用了exp的toString()

System.out.println(exp.toString);//这种就是显示的调用exp的toString(),结果都一样

}

1. 官方解法流程图：处理了所有的情况，也就是说，链表长度不固定的数字也能计算，而不是只能计算三位数以内的加法，同时可以联想减法怎么做。

