1.态度：  
 不卑不亢，有自信

2. 重复题目，确认要求：  
 确认sample input/output  
 edge case:  
 输入：有序/无序，为否为空，重复数字，无解，只有1个数，数字是正/负/零/#,取值范围  
 LinkedList：改变node数值  
 二叉树：值是什么类型的  
 输出：为空是否可以作为输出，重复的怎么计算  
 本段代码: Class叫什么？ 注重time complexity or space complexity

3. 思考过程的交流

4. 写出数据结构和算法，解释  
 写出idea及method(如有必要，请画图)  
 idea：  
 方法名 brute force / two pointer / change next pointer  
 method：  
 包含定义好的变量名的，traversal/recursion等过程名的过程表述。e.g. traversal the linklist to save node val in node\_list  
 用case1 case2 case3表示各个条件情况  
 要简单一些。eg.只要说traverse/scan index即可，不用说from i to j,  
 给出space/time complexity  
 对于dic等space comp，要具体分析。由于题目条件，可能存在只有有限个(2个等)，故此时sO(2) = sO(1),而不是通常的O(N)  
 暴力+优化的层次分析  
 tips：  
 为了保险，用for i in range(len(x)), 而不是 for elem in list  
  
5. code之前：  
 'should I implement it in code? or you want me to continue to optimize it?'  
 'we have ... minutes. It seems that I can finish it in time.'  
  
6. right coding  
 写代码时不要自言自语，想什么就跟考官交流，说我要做什么什么了  
 写出框架，再擦掉即可：  
 def func(self,input): #inp：XXX， outp：numb of string...  
 res = 0  
 方框  
 return res  
 开始就要对变量进行定义，不要后补: temp/ans/res  
 comments:  
 最需要写注释的是代码中那些技巧性的部分. 如果你在下次 代码审查 的时候必须解释一下, 那么你应该现在就给它写注释.  
 对于复杂的操作, 应该在其操作开始前写上若干行注释.  
 对于不是一目了然的代码, 应在其行尾添加注释.  
 绝不要描述代码. 假设阅读代码的人比你更懂Python, 他只是不知道你的代码要做什么  
 针对‘method’中列出的case，针对性的注释case1/case2/case3即可  
 code比method多内容时，进行解释和补充

7. 跑test case  
 用面试官的case先跑  
 先针对helper function 进行test，后针对主函数 进行test  
 test case：  
 链表：可以画图指示箭头表示  
 array: 分为两个部分，总结果表示各个关键变量的值，用table表示；小结果写在code的右边，即写即擦，展现具体的计算过程

8. 发音和书写  
 发音：慢而清晰  
 书写：书写行距适中，缩进4个空格，行距！！！  
  
9. 考官交流  
 问了必答   
 说啥是啥，大部分是想给我提示的  
 following up没时间写时，可以只讲思路，怎么做就行。等面试结束后，可以查面试官邮箱，把具体方法发到他邮箱里。