首先研究一下我突如其来想的问题，字符串倒置问题。

（1）StringBuffer中的reverse()方法可以实现字符串倒置。但是使用前需要做相应的类型转化。

String s = "abc";

StringBuffer sb = new StringBuffer(s);

System.out.print(sb.reverse());

（2）干倒置。就化成char数组然后把数组倒置呗。值得注意的是char数组转化成string，

New string（char[]）构造string的方法包含了char数组的构造方法。

String s = "abc";

char[] c = s.toCharArray();

char temp;

for(int i=0;i<s.length()/2;i++){

temp=c[i];

c[i]=c[s.length()-1-i];

c[s.length()-1-i]=temp;

}

String ss = new String(c);

System.out.print(ss);

然后在研究一下的char[]转化为string

（1）string的构造方法中含有以char[]数组作为参数的构造方法

String s = new String（char[] c）;

（2）String.valueOf(char[] c)方法

String s = String.valueOf(char[] c);

题目：

Given an array *S* of *n* integers, find three integers in *S* such that the sum is closest to a given number, target. Return the sum of the three integers. You may assume that each input would have exactly one solution.

For example, given array S = {-1 2 1 -4}, and target = 1.

The sum that is closest to the target is 2. (-1 + 2 + 1 = 2).

这个是以上一个为基础，当给出target没有时，向上寻找输出有的最近的target。因为昨天做的就是根据target来查找哪三个数字然后返回List。以上次为基础，自己在用递归做个判断，如果给出target，查找出的list为空就说明没有匹配数字，那么target+=1；然后再次判断，直到list有内容位置返回target。

public static int threeSumClosest(int[] nums, int target) {

int n=target;

ThreeSum ts = new ThreeSum();

List<List<Integer>> reslist = new ArrayList<>();

reslist = ts.threeSum(nums,target);

if(reslist.isEmpty()){

reslist=ts.threeSum(nums,n+=1);

}

System.out.println(reslist);

return n;

}

其实没有什么可写的，但是leetcode官网也没有给答案。不知道我想的对不对。