给定两个二进制字符串，返回它们的和（也是一个二进制字符串）。

例如，  
a = "11"  
b = "1"  
返回"100"。

首先我们考虑的是如何相加

我们把对应位置上的数字加起来再加上上一个位置的进位，即该位的加和sum

Sum%进制 = 该位的数字

Sum/进制 = 该位的进位

然后我们处理的是当一个数字完毕后，另一个较长数字

当两个数字都还有，我们对应位置和是进位加上两个数字对应位置的数字

当一个数字没有了，我们对应位置和是进位加上还有的那个数字对应位置的数字。

我们用i，j两个指针表示再两个数字的位置，每次加完一个就—

只要i或者j位置有效即>=0，我们就加入该位置的数字，无效则不加入

直到两个数字都遍历完毕为止

最后还有一个进位，加入进位就完成

时间复杂度O(max(a.length,b.length)),空间复杂度O(max(a.length,b.length))

这个方法可以适用于任何进制的两个数字的加法

代码

public static String addBinary(String a,String b){

int i = a.length()-1;

int j = b.length()-1;

int carry = 0;//进位

StringBuilder sb = new StringBuilder();

while(i>=0||j>=0){

int sum = carry;

if(i>=0)

sum += a.charAt(i--)-'0';

if(j>=0)

sum += b.charAt(j--)-'0';

//我们把对应位置上的数字和进位加起来就是该位的总的加和sum

//sum % 进制 = 该位的进位后的数字

//sum / 进制 = 该位的进位

sb.append(sum%2);

carry = sum/2;

}

if(carry!=0)//最后再处理下进位

sb.append(carry);

return sb.reverse().toString();

}