二分

**int binsearch(int \*t,int k,int n)**

**{//t为数组，k是要查找的数，n为长度，此为升序**

**int low = 1,high = n,mid;**

**while(low<=high)**

**{**

**mid = (low+high)/2;**

**if(k == t[mid])**

**return mid;**

**else if(k<t[mid])**

**high = mid-1;**

**else**

**low = mid+1;**

**}**

**return -1;**

**}**

**快速幂取模**

**long long quickmod(long long a,long long b,long long m)**

**{**

**long long ans = 1;**

**while(b)//用一个循环从右到左便利b的所有二进制位**

**{**

**if(b&1)//判断此时b[i]的二进制位是否为1**

**{**

**ans = (ans\*a)%m;//乘到结果上，这里a是a^(2^i)%m**

**b--;//把该为变0**

**}**

**b/=2;**

**a = a\*a%m;**

**}**

**return ans;**

**}**

**Kmp**

**int next[N];**

**char str1[M],str2[N];**

**//str1 长，str2 短**

**//len1,len2,对应str1,str2的长**

**void get\_next(int len2)**

**{**

**int i = 0,j = -1;**

**next[0] = -1;**

**while(i<len2)**

**{**

**if(j == -1 || str2[i] == str2[j])**

**{**

**i++;**

**j++;**

**if(str2[i] != str2[j])**

**next[i] = j;**

**else**

**next[i] = next[j];**

**}**

**else**

**j = next[j];**

**}**

**//计算某字符串的周期，如aaaa是4，abcd是1**

**/\***

**int i = 0;j = -1;**

**next[0] = -1;**

**while(str2[i])**

**{**

**if(j == -1 || str2[i] == str2[j])**

**{**

**i++;j++;**

**next[i] = j;**

**}**

**else**

**j = next[j];**

**}**

**len = strlen(str);**

**i = len-j;**

**if(len%i==0)**

**return len/i;**

**else**

**return 1;**

**\*/**

**}**

**int kmp(int len1,int len2)**

**{**

**int i = 0,j = 0;**

**get\_next(len2);**

**while(i<len1)**

**{**

**if(j == -1 || str1[i] == str2[j])**

**{**

**i++;**

**j++**

**}**

**else**

**j = next[j];**

**/\***

**if(j == len2)//计算str2在str1中出现多少次**

**{**

**cnt++;**

**j= next[j];**

**}**

**\*/**

**}**

**//return j; //j为匹配的长度**

**if(j>len2)**

**return 1;//这里也可以返回i-len2来获得匹配在主串中开始的位置**

**else**

**return 0;**

**}**

**//数字KMP**

**int a[1000005],b[10005];**

**int next[10005],n,m;**

**void getnext()**

**{**

**int i = 0,j = -1;**

**next[0] = -1;**

**while(i<m)**

**{**

**if(j == -1 || b[i] == b[j])**

**{**

**i++;**

**j++;**

**if(b[i] == b[j])**

**next[i] = next[j];**

**else**

**next[i] = j;**

**}**

**else**

**j = next[j];**

**}**

**}**

**int kmp()//返回匹配位置**

**{**

**int i = 0,j = 0;**

**while(i<n)**

**{**

**if(a[i] == b[j])**

**{**

**if(j == m-1)**

**return i-j+1;**

**i++;**

**j++;**

**}**

**else**

**{**

**j = next[j];**

**if(j == -1)**

**{**

**i++;**

**j = 0;**

**}**

**}**

**}**

**return -1;**

**}**

**拓展欧几里得**

**int gcd(int n,int m)//n>m**

**{**

**//最大公约数**

**int r;**

**while(m)**

**{**

**r = n%m;**

**n = m;**

**m = r;**

**}**

**return n;**

**}**

**int kgcd(int a,int b)**

**{**

**if(!a) return b;**

**if(!b) return a;**

**if(!(a&1) && !(b&1))**

**return kgcd(a>>1,b>>1)<<1;**

**else if(!(b&1)) return kgcd(a,b>>1);**

**else if(!(a&1)) return kgcd(a>>1,b);**

**else return kgcd(abs(a-b),min(a,b));**

**}**

**//扩展欧几里得**

**//求方程ax+by+c = 0有多少整数解**

**int extgcd(int a,int b,int &x,int &y)**

**{**

**if(!b)**

**{**

**x=1;**

**y=0;**

**return a;**

**}**

**int d = extgcd(b,a%b,x,y);**

**int t = x;**

**x=y;**

**y=t-a/b\*y;**

**return d;**

**}**

**最长公共子序列**

**#include <stdio.h>**

**#include <string.h>**

**#include <algorithm>**

**using namespace std;**

**char s1[1000],s2[1000];**

**int len1,len2,dp[1000][1000],mark[1000][1000];//如果数据太大，dp数组可以考虑滚动数组**

**void LCS()**

**{**

**int i,j;**

**memset(dp,0,sizeof(dp));**

**for(i = 0;i<=len1;i++)**

**mark[i][0] = 1;**

**for(i = 0;i<=len2;i++)**

**mark[0][i] = -1;**

**for(i = 1; i<=len1; i++)**

**{**

**for(j = 1; j<=len2; j++)**

**{**

**if(s1[i-1]==s2[j-1])**

**{**

**dp[i][j] = dp[i-1][j-1]+1;**

**mark[i][j] = 0;**

**}**

**else if(dp[i-1][j]>=dp[i][j-1])**

**{**

**dp[i][j] = dp[i-1][j];**

**mark[i][j] = 1;**

**}**

**else**

**{**

**dp[i][j] = dp[i][j-1];**

**mark[i][j] = -1;**

**}**

**}**

**}**

**}**

**void PrintLCS(int i,int j)**

**{**

**if(!i && !j)**

**return ;**

**if(mark[i][j]==0)//公共的**

**{**

**PrintLCS(i-1,j-1);**

**printf("%c",s1[i-1]);**

**}**

**else if(mark[i][j]==1)**

**{**

**PrintLCS(i-1,j);**

**printf("%c",s1[i-1]);**

**}**

**else**

**{**

**PrintLCS(i,j-1);**

**printf("%c",s2[j-1]);**

**}**

**}**

**大数**

**package cn.renly.ACMtest;**

**import java.math.BigInteger;**

**import java.lang.String;**

**import java.util.Scanner;**

**public class Main {**

**public static void main(String[] args) {**

**Scanner cin=new Scanner(System.in);**

**int cases = cin.nextInt();**

**for(int i=1;i<=cases;i++){**

**BigInteger a = cin.nextBigInteger();**

**BigInteger b = cin.nextBigInteger();**

**if(i!=1)System.out.println();**

**System.out.println("Case "+i+":");**

**System.out.print(a+" + "+b+" = ");**

**System.out.println(a.add(b));**

**}**

**}**

**}**