Hello World ! --other

二分

题意：给出N个凹凸曼的座标，N个小怪兽的座标，问最近的凹凸曼和小怪兽的距离

//http://acm.csu.edu.cn/OnlineJudge/problem.php?cid=2036&pid=2

思路：凹凸曼的点按x座标排好序

对于小怪兽i，二分查找找到(bi.x - ans, bi.x + ans) 这个区间，然后就只要枚举这个区间更新最优值就行了

int main()

{

int A, B;

while(scanf("%d%d%d", &n, &A, &B) != EOF) {

for(int i = 0; i < n; i++) scanf("%d%d", &a[i].x, &a[i].y);

for(int i = 0; i < n; i++) scanf("%d%d", &b[i].x, &b[i].y);

sort(a, a+n);

sort(b, b+n);

double ans = 99999999999;

for(int i = 0; i < n; i++) {

for(int j = Find\_Min(max(0, (int)(a[i].x-ans)+1)); j <= Find\_Max((int)(a[i].x+ans)-1); j++)

ans = min(ans, \_dis(a[i], b[j]));

}

printf("%.3lf\n", ans/(A+B));

}

return 0;

}

指数循环节 欧拉函数 A^B mod C

题意：计算A^B mod C , (1<=A,C<=1000000000,1<=B<=10^1000000).

思路：指数循环节: A^x mod C = A^(x mod φ(C) + φ(C)) mod C

判断n!是否能够被m整除

计算方法是把m进行质因数分解，看下m的每一个质因数是否能够在n!中找到；

n!中间包含了多少个x(x是任意的一个数，不过一般情况下我们都只讨论x为质数)，这种问题的答案是：n/x+n/(x^2)+n/(x^3).....[一直加到x的乘方不超过n]，这个定理的证明也非常的简单，这里就不再赘述了.根据以上观点，就可以分别计算m的每一个质因数是否被完全包含，如果有一个没有被包含，那么就不能被整除！

算n!包含多少个x 可以用优美的递归 : F(n,x) = n/x + F(n/x, x)

ask:

为什么素数表只要开到46340? sqrt( 2^31 )

从因子大到小检查能加速? a little

problem link : <http://acm.hust.edu.cn/vjudge/problem/viewProblem.action?id=21255>

void divide(ll n)

{

fac\_num = 0;

for(int i = 0; p[i] <= n && i < prime\_num; i++)

if(n % p[i] == 0) {

fac[fac\_num].p = p[i], fac[fac\_num].n = 0;

while(n % p[i] == 0) {

fac[fac\_num].n ++;

n /= p[i];

}

fac\_num++;

}

if(n > 1) fac[fac\_num].p = n, fac[fac\_num++].n = 1;

}

int calc(ll n, int x)

{

if(n == 0) return 0;

return n/x + calc(n/x, x);

}

bool judge(ll n)

{

// for(int i = 0; i < fac\_num; i++) printf("!%d\n", fac[i].p);

for(int i = fac\_num-1; i >= 0; i--) { //从大到小能加速?

// int cnt = 0; ll j = fac[i].p;

// while(n >= j) {

// cnt += n/j;

// j \*= fac[i].p;

// }

// if(cnt < fac[i].n) return false;

if(calc(n, fac[i].p) < fac[i].n) return false; //递归算，慢一点点，但是优美

}

return true;

}

/\*

凑1-n最少要多少个数字

\*/

//http://acm.csu.edu.cn/OnlineJudge/problem.php?cid=2037&pid=0

#include <math.h>

#include <stdio.h>

double LOG2(int x)

{

return log(x\*1.0)/log(2.0);

}

int main()

{

int n;

while(scanf("%d", &n) == 1) {

printf("%d\n", (int)LOG2(n)+1);

}

return 0;

}