青少年
信息学奥林匹克竞赛
实战辅导丛书

数学与程序设计

丛书主编 李立新 沈军 王晓敏
本册主编 林厚从 王新

东南大学出版社

青少年信息学奥林匹克竞赛实战辅导丛书

数学与程序设计

丛书主编　李立新　沈　军　王晓敏
本册主编　林厚从　王　新

东南大学出版社
·南京·

图书在版编目(CIP)数据

数学与程序设计 / 林厚从，王新主编. —南京：东南大学出版社，2008.12

（青少年信息学奥林匹克竞赛实战辅导丛书 / 李立新，沈军，王晓敏主编）

ISBN 978-7-5641-1439-8

Ⅰ.数… Ⅱ.①林…②王… Ⅲ.①数学—青少年读物②程序设计—青少年读物 Ⅳ.01-49 TP311-49

中国版本图书馆 CIP 数据核字（2009）第 004518 号

数学与程序设计

主　编	林厚从　王　新
责任编辑	张　煦
出 版 人	江　汉
出版发行	东南大学出版社
社　址	南京市四牌楼 2 号(邮编 210096)
经　销	江苏省新华书店
印　刷	南京玉河印刷厂
版　次	2008 年 12 月第 1 版　2008 年 12 月第 1 次印刷
开　本	787 mm×1092 mm　1/16
印　张	22.5
字　数	530 千字
印　数	1—5000 册
书　号	ISBN 978-7-5641-1439-8/TP · 235
定　价	34.00 元

（凡因印装质量问题，请直接向东大出版社读者服务部调换。电话：025－83792328）

丛书序

得益于计算机工具的特殊结构，以计算机技术为核心的信息技术现在已在整个社会发展中起到了极其重要的作用。同时，由于信息技术的本质在于不断创新，因而人们将21世纪称为信息世纪。根据人类生理特征，青少年时期正处于思维活跃、充满各种幻想的黄金年代，孕育着创新的种子和潜能。长期的实践活动告诉我们，青少年信息学奥林匹克竞赛可以让广大的青少年淋漓尽致地展现其思维的火花，享受创新带来的美感。因此，该项活动得到了全国各地广大青少年朋友的喜爱，越来越多的青少年朋友怀着浓厚的兴趣加入到这项活动中来。

从本质上看，计算机学科是一种思维学科，正确地思维训练可以播种持续创新的优良种子。相对于其他学科的竞赛，信息学竞赛覆盖知识面更为宽广，涉及了数学、数据结构、算法、计算几何、人工智能等相关的专业知识，如何在短时间内有效地掌握这些知识的主体，并灵活地应用其解决实际问题，显然是一个值得认真思考的问题。

知识学习与知识应用基于两种不同的思维策略，且这两种策略的统一本质上依赖于选手自身的领悟，但是如何建立两种策略之间的桥梁、快速地促进选手自身的领悟，显然是教材以及由其延伸的教学设计与实施过程所应考虑的因素。竞赛训练有别于常规的教学，要在一定的时间内得到良好的效果，需要有一定的技术方法，而不应拘泥于规范。从学习的本质看，各种显性知识的学习是相对容易的，或者说，只要时间允许，总是可以消化和理解的；然而，隐性知识的学习和掌握却是较难的。由于隐性知识的学习对竞赛和能力的提高起到决定性的作用，因此，仅仅依靠选手自身的感悟，而不从隐性知识的层面重新组织知识体系，有目的地辅助选手自身主动建构，显然是不能提高竞赛能力的。基于上述认识，结合多年来开展青少年信息学竞赛活动的经验，我们组织了一批有长期一线教学经验的教练员和专家、教授编写出版了这套《青少年信息学奥林匹克竞赛实战辅导丛书》。

丛书的主要特点如下：

1. 兼顾广大青少年课外学习时间的短暂与知识内容较多的矛盾，考虑我国青少年信息学竞赛的特点和安排，丛书分成四个层次，分别面向日常常规训练、数据结构与数学知识强化(包括基本数据结构与数学知识及应用、高级数据结构及应用)、重点专题解析和典型试题解析，既考虑知识体系的系统性及连续训

练的特点，又考虑各个层次选手独立训练的需要。

2. 区别于常规的教学模式，每册丛书的体系设计以实战需要为核心主线，突出重点，整个体系从逻辑上构成符合某种知识体系学习规律的系统化结构。

3. 围绕实战辅导需求，在解析知识和知识应用关系所蕴涵的递归思维策略的基础上，重构知识点关系，采用抛锚式和支架式并重教学思路，突出并强化知识和知识应用两者之间的联系。

4. 在显性知识及其关系基础上，强调知识应用模式及其建构的学习方法的教学，注重学习思维和能力的训练，实现知识应用能力和竞赛能力的提高，强化从程序设计及应用的角度来进行训练的特点。

5. 整套丛书的设计，不仅注重竞赛实战的需要，还考虑选手未来的发展，强调计算机程序设计正确思维的训练和培养，以不断建立持续创新的源泉。

恰逢邓小平同志“计算机的普及要从娃娃抓起”重要讲话发表 25 周年之际，我们期望以此奉献给广大读者朋友一套立意新、选材精、内容丰富的青少年信息学奥赛读本。

本套丛书的编写与出版，得到了东南大学出版社的大力支持，在此表示致感谢！

李立新　沈军　王晓敏

2008 年 12 月

前　言

N. Wirth 给出了程序设计的一个重要公式：程序＝算法＋数据结构。

作为一个程序员来说，不应该只把程序设计作为一门技术，更应该看成是一种艺术！其实，算法本身也是一门艺术！数据结构本身也是一门艺术！程序也好，算法也好，数据结构也好，其中都蕴涵了很多的数学，而数学更是一门艺术！如果把数学与程序设计完美地结合在一起，则是艺术的巅峰！

从某种意义上来说，计算机源于数学。而作为计算机科学核心技术的程序设计与数学之间的关系更是密不可分，可以这样说，数学是计算机程序设计的灵魂！利用数学方面的知识、数学分析的方法以及数学题解的技巧，可以使得程序设计变得轻松、美观、高效，而且往往能反映出问题的本质。

在国际、国内的各项信息学奥林匹克竞赛中，越来越多地出现了数学的影子，也越来越多地用到了数学知识，对选手的数学修养要求也越来越高。本书的目的就在于给广大中学、大学信息学参赛者和爱好者，介绍和总结一些信息学奥赛中常用的数学知识和数学方法，希望能起到抛砖引玉的作用。编写本书的动机之一是想把我的恩师——南京航空航天大学李立新教授的部分讲座讲义整理，把自己这几年的一些授课讲义和自编的一些题目进行归纳。本书吸收了很多 OIer 的灵感和智慧，复旦大学的王晨、卞晶同学也为本书的编写做了大量调试和校对工作，在此一并表示感谢！

由于水平有限，书中难免有不当之处，恳请谅解，也欢迎广大读者批评指正，不胜感激！

本书中所有例题的参考程序均采用 Free Pascal 实现。如果需要本书中的所有测试数据，请发 E-mail 给 hc. lin@163. com 联系。

作　者

2008 年 8 月

目　　录

第一章　初等数论

1.1　概述

数论被誉为数学的皇后，它的研究对象是我们经常接触的整数及其一些性质，例如求两个整数的最大公约数和最小公倍数、求一个正整数的素因子分解、求一个方程的整数解等等。

1.1.1　数论的起源

古希腊人把对于数的抽象关系的研究与用数进行计算的实际技能区别开来，前者称为算术(arithmetic)，后者称为算术计算术(logistic)。这种分类法，从中世纪延续下来一直用到 15 世纪末，教科书中开始用单一的名称——“算术”来论述数的理论方面和实用方面。有趣的是，在今天的欧洲大陆，arithmetic 有其原始意义；而在英国和美国，一般所说的 arithmetic 是古代 logistic 的同义词，并且，在这两个国家，使用描述性的术语 number theory(数论)来表示对数的研究的抽象方面。

毕达哥拉斯和他的学派在数学上有很多创造，他们尤其对整数的变化规律感兴趣。例如，把全部因数之和(除其本身以外)等于本身的数称为完全数(perfect number，又称完美数)；而将本身大于其因数之和的数称为盈数；将小于其因数之和的数称为亏数。上帝 6 天创造世界，而 6 就是一个完全数，因为 $6=1+2+3$；阿尔克温说整个人类是诺亚方舟上的神灵下凡，这一创造是不完善的，所以 8 是个亏数，因为 $8>1+2+4$。直到 1952 年，人类才知道 12 个完全数，它们都是偶数，其中前三个是 6、28 和 496。

一般认为，毕达哥拉斯及其后继者，连同这个团体的哲学，是数论发展的先驱，是后来把数论发展为神秘主义的基础。例如，亚姆利库，这位公元 320 年左右有影响的新柏拉图派哲学家，就曾把亲和数(amicable numbers)的发现归功于毕达哥拉斯。两个数是亲和的，即一个数的真因子的和等于另一个数。例如 284 和 220 就是亲和的，因为 284 的真因子是 1、2、4、71、142，其和为 220；而 220 的真因子有 1、2、4、5、10、11、20、22、44、55、110，其和为 284。后来这又增添了神秘的色彩和迷信的意思，即分别写上这两个数的护符会使两数的佩带者保持良好的友谊，这种数在魔术、法术、占星学和占卦上，都起着重要的作用。奇怪的是，好像很长一段时间再没有发现新的亲和数，直到 17 世纪，费马手持一本《算术》，并在其空白处写写画画，竟把数论引上了近代的轨道，并于 1636 年宣布 17296 和 18416 为另一对亲和数。但后来得知这只不过是重新发现，这对亲和数早在 13 世纪末 14 世纪初就曾被斑纳(1256—1321)发现过，并且也许曾被泰比特・伊本柯拉用于其公式。又过了两年，法

国数学家笛卡儿给出了第三对。再后来，瑞士数学家欧拉着手于系统地寻找亲和数，于1747年给出了一个30对亲和数的表，后来又扩展到超过60对。在这种数的研究历史中，还有一件奇怪的事：一个16岁的意大利男孩帕加尼尼在1886年发现了被人们忽视、比较小的一对亲和数：1184和1210。

出现现代计算机后，数论的研究得到了迅猛发展，因为很多数论的证明和猜想都可以通过高速计算机来解决和验证，甚至不断地穷举出来。本章就介绍一些简单的数论知识及相应的实际应用。

1.1.2 整除

设 a,b 都是非0整数，如果存在一个整数 q，使得 $b=a\times q$，那么就说 b 可被 a 整除，记作 $a|b$，且称 b 是 a 的倍数，a 是 b 的约数(因子)。例如，$3|12$，$21|63$。下面是有关整除的几个性质：

(1) 如果 $a|b$ 且 $b|c$，那么 $a|c$。

(2) $a|b$ 且 $a|c$ 等价于对任意的整数 x,y，有 $a|(b\times x+c\times y)$。

(3) 设 $m\neq 0$，那么 $a|b$ 等价于 $(m\times a)|(m\times b)$。

(4) 设整数 x,y 满足下式：$a\times x+b\times y=1$，且 $a|n$、$b|n$，那么 $(a\times b)|n$。

证明：因为 $a|n$ 且 $b|n$，由性质(3)可得 $(a\times b)|(b\times n)$ 且 $(a\times b)|(a\times n)$

再由性质(2)得 $(a\times b)|(a\times n\times x+b\times n\times y)$

而 $a\times n\times x+b\times n\times y=n\times(a\times x+b\times y)=n\times 1=n$

所以 $(a\times b)|n$

(5) 若 $b=q\times d+c$，那么 $d|b$ 的充要条件是 $d|c$。

另外还有一些有用的例子，比如：若2能整除 a 的最末位(约定0可以被任何数整除)，则 $2|a$，若4能整除 a 的最后两位，则 $4|a$，若8能整除 a 的最后三位，则 $8|a$，…；若3能整除 a 的各位数字之和，则 $3|a$；若9能整除 a 的各位数字之和，则 $9|a$；若11能整除 a 的偶数位数字之和与奇数位数字之和的差，则 $11|a$。同时，能被7、11、13整除的数的特征是：这个数的末三位数与末三位以前的数字所组成的数之差能被7、11、13整除。

1.1.3 最大公约数与最小公倍数

一般地，设 $a_1,a_2,a_3,\cdots,a_k$ 是 k 个非0整数，如果存在一个非0整数 d，使得 $d|a_1$，$d|a_2$，…，$d|a_k$，那么 d 就称为 $a_1,a_2,a_3,\cdots,a_k$ 的公约数。公约数中最大的一个称为最大公约数，记为 $\mathrm{GCD}(a_1,a_2,a_3,\cdots,a_k)$，显然它是存在的，至少为1。当GCD=1时，称这 n 个数是互质的或既约的。公约数一定是最大公约数的约数。

一般地，设 $a_1,a_2,a_3,\cdots,a_k$ 是 k 个非0整数，如果存在一个非0整数 d，使得 $a_1|d$，$a_2|d$，…，$a_k|d$，那么 d 就称为 $a_1,a_2,a_3,\cdots,a_k$ 的公倍数。公倍数中最小的一个称为最小公倍数，记为 $\mathrm{LCM}(a_1,a_2,a_3,\cdots,a_k)$，显然它也是存在的。公倍数一定是最小公倍数的倍数。

数学上，求两个数的最大公约数常用辗转相除法(或辗转相减法)；而求两个数的最小公倍数的方法可以用穷举法或逐步倍增法，如求3和8的最小公倍数，可以让 n 从1开始逐步加1，不断检查 $8\times n$ 是不是3的倍数。

定理：a、b 两个数的最大公约数乘以它们的最小公倍数等于 a 和 b 本身的乘积。

比如，要求 3 和 8 的最小公倍数，则 LCM(3,8)= 3×8 div GCD(3,8)= 24。

1.1.4　勾股数

对于正整数 X,Y,Z，如果满足下列不定方程：$X^2+Y^2=Z^2$，则称这组正整数(X,Y,Z)为勾股数它们有着不同的几何含义，如作为一个三角形的三个边。一般限定 X 和 Y 互质，且 X 为偶数。认为(4,3,5)和(6,8,10)为本质相同的勾股数，(X,Y,Z)，(X,Z,Y)，(Y,X,Z)，(Y,Z,X)，(Z,X,Y)，(Z,Y,X)是一组相同的勾股数。

勾股数有如下几个性质：

(1) X,Y,Z 一定两两互质。

(2) X,Y 一定一奇一偶。

(3) $X+Z$ 一定是一个完全平方数。

(4) $(Y+Z)/2$ 也是一个完全平方数。

(5) $X\times Y\times Z$ 一定能被 60 整除。

1.1.5　应用举例

例 1-1　求 n 个数的最大公约数与最小公倍数

[问题描述]

编程求 n 个($n\leqslant 100$)正整数 A_i($A_i\leqslant 30\,000$，$1\leqslant i\leqslant n$)的最大公约数和最小公倍数。假设解一定在长整数范围之内。

[输入样例]

```
3        {3 个数}
3  4  5  {3 个数的值分别为 3,4,5}
```

[输出样例]

```
1       {最大公约数}
60      {最小公倍数}
```

[问题分析]

方法 1　对于最大公约数，我们可以先找出 $M=\min\{A_i\}$，然后让 j 从 M 到 1 逐步递减穷举，判断所有 A_i 除以 j 的余数是否都等于 0，第 1 个满足的 j 值即为解。对于最小公倍数，设 $M=\max\{A_i\}$，然后让 j 从 1 开始逐步递增穷举，判断 $M\times j$ 除以所有 A_i 的余数是否都等于 0，第 1 个满足的 $M\times j$ 值即为解。

方法 2　先求出两个数的最大公约数(最小公倍数)，再和其他数求最大公约数(最小公倍数)，只需调用函数 $n-1$ 次。

[参考程序]

```
program p1_1a(input,output);  {方法 1}
var i,j,k,n,min,max,gcd,lcm:longint;
```

```
    key:boolean;
    a:array[1..100] of longint;
begin
  readln(n);
  for i:=1 to n do  read(a[i]);
  min:=maxlongint;
  for i:=1 to n do  if min>a[i] then min:=a[i];
  for i:=min downto 1 do
    begin
      key:=true;
      for j:=1 to n do  if a[j] mod i>0 then key:=false;
      if key then begin gcd:=i;break;end;
    end;
  max:=0;
  for i:=1 to n do  if max<a[i] then max:=a[i];
  i:=0;
  repeat
    i:=i+max;
    key:=true;
    for j:=1 to n do  if i mod a[j]>0 then key:=false;
    if key then begin lcm:=i;break;end;
  until key;
  writeln('Gcd=',gcd);
  writeln('Lcm=',lcm);
end.

program p1_1b(input,output);  {方法 2}
var i,j,k,n,gcd,lcm:longint;
   a:array[1..100] of longint;

function gcdnum(p,q:longint):longint;  {递归函数,利用辗转相除法求 p 和 q 的最大公约数}
begin
    if p mod q=0 then gcdnum:=q else gcdnum:=gcdnum(q,p mod q);
end;

begin {main}
  readln(n);
  for i:=1 to n do  read(a[i]);
  gcd:=a[1];lcm:=a[1];
```

```
    for i:=2 to n do
      begin
        gcd:=gcdnum(gcd,a[i]);
        lcm:=lcm * a[i] div gcdnum(lcm,a[i]);
      end;
    writeln('Gcd=',gcd);
    writeln('Lcm=',lcm);
  end.
```

[拓展探究]

(1) 本题也可以采用"二进制算法"求解。二进制算法求两个整数 a 和 b 的最大公约数方法如下：

若 $a=b$，则 $GCD(a,b)=a$，否则：

① 若 a,b 均为偶数，则 $GCD(a,b) = 2GCD(a/2,b/2)$

② 若 a 为偶数，b 为奇数，则 $GCD(a,b) = GCD(a/2,b)$

③ 若 a 为奇数，b 为偶数，则 $GCD(a,b) = GCD(a,b/2)$

④ 若 a,b 均为奇数，则 $GCD(a,b) = GCD(a-b,b)$

(2) 给出两个正整数 $a,b(1\leqslant a,b\leqslant 10^{100})$，求这两个数的最大公约数和最小公倍数。

输入格式：两行，每行一个正整数，分别表示 a 和 b。

输出格式：两行，分别为 a 和 b 的最大公约数 $GCD(a,b)$ 和最小公倍数 $LCM(a,b)$。

输入样例：

```
123
321
```

输出样例：

```
3
13161
```

例 1-2　欧几里德的游戏

[问题描述]

欧几里德的两个后代 Stan 和 Ollie 正在玩一种数字游戏，这个游戏是他们的祖先欧几里德发明的。给定两个正整数 M 和 N，从 Stan 开始，取其中较大的一个数，减去较小的数的正整数倍，当然，得到的数 K 不能小于 0。然后是 Ollie 对刚才得到的数 K，和 M、N 中较小的那个数，再进行同样的操作……直到一个人得到了 0，他就取得了胜利。下面是他们用 (25,7) 两个数游戏的过程：

Start:　25　7

Stan:　11　7 {18　7, 11　7, 4　7 均可能}

Ollie:　4　7

Stan :　4　3

Ollie:　1　3

Stan： 1 0

Stan 赢得了游戏的胜利。

现在，假设他们"完美"地操作，谁会取得胜利呢？

[输入格式]

输入文件(game. in)的第一行为测试数据的组数 C。

下面有 C 行，每行为一组数据，包含两个正整数 M 和 N，M 和 N 的大小不超过长整型数的范围。

[输出格式]

输出文件(game. out)要求对每组输入数据输出一行。

如果 Stan 胜利，则输出"Stan wins"；否则输出"Ollie wins"。

[输入样例]

```
2
25 7
24 15
```

[输出样例]

```
Stan wins
Ollie wins
```

[问题分析]

这道题目很容易让人想起辗转相除法。事实上，辗转相除的除数和余数所形成的各种状态都会在游戏过程中出现。例如下面的游戏过程：

游戏过程的各个状态	辗转相除的过程	属于哪一局	所属情况
50 18	50 / 18 = 2…14	第 1 局的初状态	第一种
32 18		第 1 局的末状态	
14 18	18 / 14 = 1…4	第 2 局的初状态（也是末状态）	第二种
14 4	14 / 4 = 3…2	第 3 局的初状态	第一种
10 4		第 3 局的中间状态	
6 4		第 3 局的末状态	
2 4	4 / 2 = 2…0	第 4 局	

我们可以把一步辗转相除以及它与下一步辗转相除之间的游戏状态称作"一局"，辗转相除的被除数和除数实际上对应了每一局的初状态。显然，每一局的初状态是游戏中必然出现的，而且最后一局只有一个状态，面临最后一局初状态的人就赢得了胜利。因此，本题

的大致想法是：尽量让自己能够取每一局的初状态，让对手取每一局（除最后一局）的末状态。下面就分两种情况来具体说说这个想法的实现。

第一种情况，先举个例子 $A=32$，$B=14$，A 和 B 的辗转相除过程如下：

32/14＝2…4

14/4 ＝3…2

4 /2 ＝2…0

这种情况的特点就是：每一步相除所得的商都大于1，即对于每一局的初状态 (A,B)，$A>B$，都满足：$A/B>1$。对于这种情况，我们的策略是取走 $(A/B-1)\times B$，这样就得到新状态（A mod B + B，B），接下来，对手就没有选择，只能取走 B，剩下（A mod B，B），而这就是下一局的初状态，这样就保证下一局的初状态肯定由自己取。这种情况下先取者（Stan）必胜。

第二种情况是某一局的初状态满足 $A/B=1$，即这一局只有一个状态，自己取完之后下一局的初状态必然落在对方手里。以下是这种情况的一般性表述：存在连续的若干局 $P_i(A_i,B_i)$，…，$P_j(A_j,B_j)$，$i+1<j$，$A_i/B_i>1$，且对于任意一局 $P_k(A_k,B_k)$，$i<k<j$，均满足 $A_k/B_k=1$。如果 $j-i$ 是奇数，则 P_j 和 P_i 的初状态的持有人相同，如果 $j-i$ 是偶数，则不同。因此，如果出现 $j-i$ 为偶数，就有必要在 P_i 局中把当局的状态一次性全部取完，即把下一局的初状态“拱手相让”，这样就可以保证在 P_j 局中再一次拿到初状态。

总之，首先拿到 $A/B>1$ 状态的人，总能根据两种情况作出正确的选择，从而在最后一局拿到初状态。因此，首先拿到 $A/B>1$ 状态的人是必胜的，但如果出现从初状态开始，每局都是 $A/B=1$，如 $A=24$，$B=15$，则要根据这样的局数多少来判断输赢。

［**参考程序**］

```
program p1_2(input,output);
var m,n,t,f,c,i:longint;
begin
  assign(input,'game.in');
  reset(input);
  assign(output,'game.out');
  rewrite(output);
  readln(c);
  for i:=1 to c do
  begin
    readln(m,n);
    if m<n then   begin t:=m; m:=n; n:=t; end; {保证 m>n}
    f:=1;                                 {f=1,表示 Stan 赢,否则 Ollie 赢}
    while (m div n=1) and (m mod n<>0) do
    begin
      t:=m mod n; m:=n; n:=t; f:=-f;
    end;
    if f=1   then writeln('Stan wins') else writeln('Ollie wins');
```