第 1 章 绪论

在 Oier 中，有一句话广为流传： 任何蒟蒻必须经过大量的刷题练习才能成为大牛乃至于神牛。 这就是著名的 lzn 定理。然而，我们这些蒟蒻们，没有经过那么多历练，却要和大牛们同场竞技，我们该怎么以弱胜强呢？答案就是： 骗分 那么，骗分是什么呢？骗分就是用简单的程序（比标准算法简单很多，保证蒟蒻能轻松搞定的程序），尽可能多得骗取分数。 让我们走进这本《新版骗分导论》，来学习骗分的技巧，来挑战神牛吧！

第 2 章 从无解出发

2.1 无解情况

在很多题目中都有这句话：“若无解，请输出-1.”

看到这句话时，骗分的蒟蒻们就欣喜若狂，因为——数据中必定会有无解的情况！那么，只要打出下面这个程序：

```
printf(“-1”);
```

就能得到 10 分，甚至 20 分，30 分！

举个例子：

NOIP2012 第 4 题，文化之旅

题目描述 Description

有一位使者要游历各国，他每到一个国家，都能学到一种文化，但他不愿意学习任何一种文化超过一次（即如果他学习了某种文化，则他就不能到达其他有这种文化的国家）。不同的国家可能有相同的文化。不同文化的国家对其他文化的看法不同，有些文化会排斥外来文化（即如果他学习了某种文化，则他不能到达排斥这种文化的其他国家）。

现给定各个国家间的地理关系，各个国家的文化，每种文化对其他文化的看法，以及这位使者游历的起点和终点（在起点和终点也会学习当地的文化），国家间的道路距离，试求从起点到终点最少需走多少路。

输入描述 Input Description

第一行为五个整数 N，K，M，S，T，每两个整数之间用一个空格隔开，依次代表国家个数（国家编号为 1 到 N），文化种数（文化编号为 1 到 K），道路的条数，以及起点和终点的编号（保证 S 不等于 T）；

第二行为 N 个整数，每两个整数之间用一个空格隔开，其中第 i 个数 Ci，表示国家 i 的文化为 Ci。

接下来的 K 行，每行 K 个整数，每两个整数之间用一个空格隔开，记第 i 行的第 j 个数为 aij，aij= 1 表示文化 i 排斥外来文化 j（i 等于 j 时表示排斥相同文化的外来人），aij= 0 表示不排斥（注意 i 排斥 j 并不保证 j 一定也排斥 i）。

接下来的 M 行，每行三个整数 u，v，d，每两个整数之间用一个空格隔开，表示国家 u 与国家 v 有一条距离为 d 的可双向通行的道路（保证 u 不等于 v，两个国家之间可能有多条道路）。

输出描述 Output Description

输出只有一行，一个整数，表示使者从起点国家到达终点国家最少需要走的距离数（如果无解则输出-1）。

样例输入 Sample Input
输入样例 1
2 2 1 1 2
1 2
0 1
1 0
1 2 10
输入样例 2
2 2 1 1 2
1 2
0 1
0 0
1 2 10
样例输出 Sample Output
输出样例 1
-1
输出样例 2
10
数据范围及提示 Data Size & Hint
【输入输出样例 1 说明】
由于到国家 2 必须要经过国家 1，而国家 2 的文明却排斥国家 1 的文明，所以不可能到达国家 2。
【输入输出样例 2 说明】
路线为 1 -> 2。
【数据范围】
对于 20%的数据，有 2≤N≤8，K≤5；
对于 30%的数据，有 2≤N≤10，K≤5；
对于 50%的数据，有 2≤N≤20，K≤8；
对于 70%的数据，有 2≤N≤100，K≤10；
对于 100%的数据，有 2≤N≤100，1≤K≤100，1≤M≤N2，1≤ki≤K，1≤u,v≤N，1≤d≤1000，S≠T，1 ≤S, T≤N。
这道题看起来很复杂，但其中有振奋人心的一句话“输出-1”，我考试时就高兴坏了（当时我才初一，水平太烂），随手打了个 printf(“-1”);,得 10 分。
2.2 样例——白送的分数
每道题目的后面，都有一组“样例输入”和“样例输出”。它们的价值极大，不仅能初步帮你检验程序的对错（特别坑的样例除外），而且，如果你不会做这道题（这种情况蒟蒻们已经司空见惯了），你就可以直接输出样例！ 例如美国的 USACO，它的题目有一个规则，就是第一组数据必须是样例。那么，只要你输出所有的样例，你就能得到 100 分（满分 1000）！这是相当可观的分数了。
现在，你已经掌握了最基础的骗分技巧。只要你会基本的输入输出语句，你就能实现这些骗分方法。那么，如果你有一定的基础，请看下一章——我将教你怎样用简单方法骗取部分分数。

第 3 章 “艰苦朴素永不忘”

本章的标题来源于《学习雷锋好榜样》的一句歌词，但我不是想教导你们学习雷锋精神，而是学习骗分！ 看到“朴素”两个字了吗？它们代表了一类算法，主要有模拟和 DFS。下面我就来介绍它们在骗分中的应用。

3.1 模拟 所谓模拟，就是用计算机程序来模拟实际的事件。例如 NOIP2012 的“寻宝”，就是写一个程序来模拟小明上藏宝塔的动作。 较繁的模拟就不叫骗分了，我这里也不讨论这个问题。 模拟主要可以应用在骗高级数据结构题上的分，例如线段树。下面举一个例子来说明一下。 排 队(USACO 2007 January Silver)

【问题描述】

每天，农夫约翰的 N（1≤N≤50000）头奶牛总是按同一顺序排好队，有一天，约翰决定让一些牛玩一场飞盘游戏（Ultimate Frisbee），他决定在队列里选择一群位置连续的奶牛进行比赛，为了避免比赛结果过于悬殊，要求挑出的奶牛身高不要相差太大。

约翰准备了 Q（1≤Q≤200000）组奶牛选择，并告诉你所有奶牛的身高 Hi（1≤ Hi ≤106）。他想知道每组里最高的奶牛和最矮的奶牛身高差是多少。

注意：在最大的数据上，输入输出将占据大部分时间。

【输入】

第一行，两个用空格隔开的整数 N 和 Q。

第 2 到第 N+1 行，每行一个整数，第 i+1 行表示第 i 头奶牛的身高 Hi

第 N+2 到第 N+Q+1 行，每行两个用空格隔开的整数 A 和 B，表示选择从 A 到 B 的所有牛（1 ≤ A ≤ B ≤ N）

【输出】

共 Q 行，每行一个整数，代表每个询问的答案。

输入样例 输出样例

6 3
1
7
3
4
2
5
1 5
4 6
2 2 6
3
0

对于这个例子，大牛们可以写个线段树，而我们蒟蒻，就模拟吧。

附模拟程序：

```
for(int i=1;i<=q;i++){
scanf(“%d%d”,&a,&b);
```

```
int min=INTMAX,max=INTMIN;
for(int i=a;i<=b;i++){
if(h[i]<min)min=h[i];
if(h[i]>max)max=h[i];
}
printf(“%d\n”,max-min);
}
```

程序简洁明了，并且能高效骗分。本程序得 50 分。

3.2 万能钥匙——DFS DFS 是图论中的重要算法，但我们看来，图论神马的都是浮云，关键就是如何骗分。下面引出本书的第 2 条定理： DFS 是万能的。 这对于你的骗分是至关重要的。比如说，一些动态规划题，可以 DFS；数学题，可以 DFS；剪枝的题，更能 DFS。下面以一道省选题为例，解释一下 DFS 骗分。 例题：NOIP2003，采药

题目描述 Description

辰辰是个天资聪颖的孩子，他的梦想是成为世界上最伟大的医师。为此，他想拜附近最有威望的医师为师。医师为了判断他的资质，给他出了一个难题。医师把他带到一个到处都是草药的山洞里对他说：“孩子，这个山洞里有一些不同的草药，采每一株都需要一些时间，每一株也有它自身的价值。我会给你一段时间，在这段时间里，你可以采到一些草药。如果你是一个聪明的孩子，你应该可以让采到的草药的总价值最大。”

如果你是辰辰，你能完成这个任务吗？

输入描述 Input Description

输入第一行有两个整数 T（1<=T<=1000）和 M（1<=M<=100），用一个空格隔开，T 代表总共能够用来采药的时间，M 代表山洞里的草药的数目。接下来的 M 行每行包括两个在 1 到 100 之间（包括 1 和 100）的整数，分别表示采摘某株草药的时间和这株草药的价值。

输出描述 Output Description

输出包括一行，这一行只包含一个整数，表示在规定的时间内，可以采到的草药的最大总价值。

样例输入 Sample Input

```
70 3
71 100
69 1
1 2
```

样例输出 Sample Output

```
3
```

数据范围及提示 Data Size & Hint

对于 30%的数据，M<=10；

对于全部的数据，M<=100。

这题的方法很简单。我们瞄准 20%的数据来做，可以用 DFS 枚举方案，然后模拟计算出最优解。附一个大致的代码：

```
void DFS(int d,int c){
```

```
if(d==n){if(c>ans)ans=c; return;}
DFS(d+1,c+w[i]);
DFS(d+1,c);
}
```

第 4 章 骗分的关键——猜想

4.1 听天由命 如果你觉得你的人品很好，可以试试这一招——输出随机数。 先看一下代码：

```
include<stdlib.h>

include<time.h>

//以上两个头文件必须加
srand(time(NULL));
//输出随机数前执行此语句
printf(“%d”,rand()%X);
//输出一个 0~X-1 的随机整数。
```

这种方法适用于输出一个整数（或判断是否）的题目中，答案的范围越小越好。让老天决定你的得分吧。 据说，在 NOIP2013 中，有人最后一题不会，愤然打了个随机数，结果得了 70 分啊!! 4.2 猜测答案 有些时候，问题的答案可能很有特点：对于大多数情况，答案是一样的。这时，骗分就该出手了。你需要做的，就是发掘出这个答案，然后直接输出。 有时，你需要运用第 3 章中学到的知识，先写出朴素算法，然后造一些数据，可能就会发现规律。 例如，本班月赛中有一道题： 炸毁计划

【问题描述】

皇军侵占了通往招远的黄金要道。为了保护渤海通道的安全，使得黄金能够顺利地运送到敌后战略总指挥地延安，从而购买战需武器，所以我们要通过你的程序确定这条战略走廊是否安全。

已知我们有 N 座小岛，只有使得每一个小岛都能与其他任意一个小岛联通才能保证走廊的安全。每个小岛之间只能通过若干双向联通的桥保持联系，已知有 M 座桥(Ai,Bi)表示第 i 座桥连接了 Ai 与 Bi 这两座城市。

现在，敌人的炸药只能炸毁其中一座桥，请问在仅仅炸毁这一座桥的情况下，能否保证所有岛屿安全，都能联通起来。

现在给出 Q 个询问 Ci，其中 Ci 表示桥梁编号，桥梁的编号按照输入顺序编号。每个询问表示在仅仅炸毁第 Ci 座桥的情况下能否保证所有岛屿安全。如果可以，在输出文件当中，对应输入顺序输出 yes，否则输出 no（输出为半角英文单词，区分大小写，默认为小写，不含任何小写符号，每行输出一个空格，忽略文末空格）。

【输入格式】

第一行 三个整数 N，M，Q，分别表示岛屿的个数，桥梁的个数和询问的个数。

第二行到第 M+1 行 每行两个整数。第 i+1 行有两个整数 Ai Bi 表示这个桥梁的属性。

第 M+2 行 有 Q 个整数 Ci 表示查询。

【输出格式】

Q 行，表示查询结果。
【样例】
destroy.in destroy.out
2 1 1
1 2
1 no
【样例范围】
对于 80%的数据，N≤100。
对于 100%的数据，N≤1000，N,Q≤M≤2000 。
你发现问题了吗？那么多座桥，炸一座就破坏岛屿的联系，可能性微乎其微（除非特别设计数据）。那么，我们的骗分策略就出来了：对于所有询问，输出 yes.果然，此算法效果不错，得 80 分。
现在知道猜测答案的厉害了吧？

4.3 寻找规律 首先声明：本节讲的规律不是正当的算法规律，而是数据的特点。某些题目会给你很多样例，你就可以观察他们的特点了。有时，数据中的某一个（或几个）数，能通过简单的关系直接算出答案。 只要你找到了规律，在很多情况下你都能得到可观的分数。 这样的题目大多出现在 NOI 或更高等级的比赛中，本人蒟蒻一个，就不举例了。传说某人去省选时专门琢磨数据的规律，结果有一题得了 30 分。 4.4 小数据杀手——打表
我认识一个人，他在某老师家上 C 语言家教，老师每讲一题，他都喊一句：“打表行吗？”
他真的是打表的忠实粉丝。表虽然不能乱打，但还是很有用的。
先看一个例子：NOIP2003 栈
题目描述 Description
栈是计算机中经典的数据结构，简单的说，栈就是限制在一端进行插入删除操作的线性表。
栈有两种最重要的操作，即 pop（从栈顶弹出一个元素）和 push（将一个元素进栈）。
栈的重要性不言自明，任何一门数据结构的课程都会介绍栈。宁宁同学在复习栈的基本概念时，想到了一个书上没有讲过的问题，而他自己无法给出答案，所以需要你的帮忙
宁宁考虑的是这样一个问题：一个操作数序列，从 1，2，一直到 n（图示为 1 到 3 的情况），栈 A 的深度大于 n。
现在可以进行两种操作，
1.将一个数，从操作数序列的头端移到栈的头端（对应数据结构栈的 push 操作）
2. 将一个数，从栈的头端移到输出序列的尾端（对应数据结构栈的 pop 操作）
使用这两种操作，由一个操作数序列就可以得到一系列的输出序列，下图所示为由 1 2 3 生成序列 2 3 1 的过程。（原始状态如上图所示） 。
你的程序将对给定的 n，计算并输出由操作数序列 1，2，…，n 经过操作可能得到的输出序列的总数。
输入描述 Input Description
输入文件只含一个整数 n（1≤n≤18）

输出描述 Output Description
输出文件只有一行，即可能输出序列的总数目
样例输入 Sample Input
3
样例输出 Sample Output
5
这题看似复杂，但数据范围太小，N<=18。所以，骗分程序就好写了：
int
a[18]={1,2,5,14,42,132,429,1430,4862,16796,58786,208012,742900,2674440,9694845,35357670,129644790,477638700};
scanf(“%d”,&n):
printf(“%d”,ans[n-1]);
测试结果不言而喻，AC 了。
学完这一章，你已基本掌握了骗分技巧。下面的内容涉及一点算法知识，难度有所增加。蒟蒻中的蒟蒻可以止步于此了。
第 5 章 做贪心的人
5.1 贪心的算法 给你一堆纸币，让你挑一张，相信你一定会挑面值最大的。其实，这就是贪心算法。 贪心算法是个复杂的问题，但你不用管那么多。我们只关心骗分。给你一个问题，让你从一些东西中选出一些，你就可以使用贪心的方法，尽量挑好的。 举个例子：这是我们的市队选拔的一道题。 2. 有趣的问题
【问题描述】
2013 年的 NOIP 结束后， Smart 发现自己又被题目碾压了，心里非常地不爽，于是
暗下决心疯狂地刷数学题目，做到天昏地暗、废寝忘食，准备在今年的中考中大展身手。
有一天，他在做题时发现了一个有趣的问题：
给定 n 个二元组(ai, bi) i)，记函数：
y=100*sigma(ai)/sigma(bi);
将函数 y 的值四舍五入取整。
现将 n 个二元组去掉其中的 k 个计算一个新的 y 值（也四舍五入取整），均能满足：y <= z ，求出最小的 z 值。Smart 想让你帮他一起找出最小的 z 值。
【输入格式】
输入包含多组测试数据。每组测试数据第一行两个整数：n 和 k；第二行为 n 个数：
a1 a2 …… an；第三行为；n 个数： b1 b2 …… bn。
输入数据当 n、k 均为 0 时结束。
【输出格式】
对于每组测试数据输出一行，即找出的最小的尤值。
注意：为避免精度四舍五入出现误差，测试点保证每个函数值与最终结果的差值至
少为 0.001 。
【样例】
math.in
3 1

5 0 1
5 1 6
4 2
1 2 7 9
5 6 7 9
0 0
math. out
83
100
【数据范围】
对于 40% 的数据： n≤20;
对于 70% 的数据： n≤1000;
对于 100% 的数据： n≤10000，ai,bi 都在 int 范围内。
这题让人望而生畏，但我们有贪心的手段。每个二元组的 a 值是乘到答案中的，所以 a 越大越好，那么只要选择出最小的 k 个去掉即可。代码就不写了，因为这个设计到下一章的内容：排序。
此代码得 20 分。
5.2 贪心地得分 我们已经学了很多骗分方法，但他们中的大多效率并不高，一般能骗 10~20 分。这不能满足我们的贪心。 然而，我们可以合成骗分的程序。举个最简单的例子，有些含有无解情况的题目，它们同样有样例。我们可以写这个程序 if(是样例)printf(样例);
else printf(“-1”);
这样也许能变 10 分为 20 分，甚至更多。 当然，合并骗分方法时要注意，不要重复骗同一种情况，或漏考虑一些情况。 大量能骗分的问题都能用此法，大家可以试试用新方法骗 2.1 中的例子“文化之旅”。 第 6 章 C++的福利
（请 P 党们跳过本章，这不是你们的福利）
在 C++中，有一个好东西，名唤 STL，被万千 Oier 们所崇拜，所喜爱。下面让我们走进 STL。
6.1 快速排序 快速排序是一个经典算法，也是 C++党的经典福利。他们有这样的代码： #include<algorithm>//这是必须的 sort(A,A+n);//对一个下标从 0 开始存储，长度为 n 的数组升序排序 就这么简单，完成了 P 党一大堆代码干的事情。 6.2 “如意金箍棒” C++里有一种东西，叫 vector 容器。它好比如意金箍棒，可以随着元素的数量而改变大小。它其实就是数组，却比数组强得多。 下面看看它的几种操作： vector<int> V;//定义 V.pushback(x);//末尾增加一个元素 x
V.popback();//末尾删除一个元素
V.size();//返回容器中的元素个数 它同样可以使用下标访问。（从 0 开始）
第 7 章 “宁为玉碎，不为瓦全”
至此，我已介绍完了我所知的骗分方法。如果上面的方法都不奏效，我也无能为力。但是，我还有最后一招—— 有句古话说：“宁为玉碎，不为瓦全”。我们蒟蒻也应有这样的精神。骗不到分，就报复一下，卡评测以泄愤吧！ 卡评测主要有两种方法：一是死循环，故意超时；二是进入终端，卡住编译器。 先介绍下第一种。代码很简单，请看：
while(1);

就是这短短一句话，就能卡住评测机长达 10s，20s，甚至更多！对于测试点多、时限长的题目，这是个不错的方法。
第二种方法也很简单，但危害性较大，建议不要在重要比赛中使用，否则可能让你追悔莫及。它就是：
include<con> （windows 系统中使用）

或 #include</dev/console> （Linux 系统中使用）
它非常强大，可以卡住评测系统，使其永远停止不了编译你的程序。唯一的解除方法是，工作人员强行关机，重启，重测。当然，我不保证他们不会气愤地把你的成绩变成 0 分。请慎用此方法。
第 8 章 实战演练
下面我们来做一些习题，练习骗分技巧。 我们来一起分析一下 NOIP2013 普及组的试题吧。
记数问题（NOIP 普及组 2013 第一题）
(count.cpp/c/pas)
描述
试计算在区间 1 到 n 的所有整数中,数字 x(0 ≤ x ≤ 9)共出现了多少次?例如,在 1 到 11 中,即在 1、2、3、4、5、6、7、8、9、10、11 中,数字 1 出现了 4 次。
【输入】
输入文件名为 count.in。
输入共 1 行，包含 2 个整数 n、x，之间用一个空格隔开
【输出】
输出文件名为 count.out。
输出共 1 行，包含一个整数，表示 x 出现的次数。
【输入输出样例】
count.in count.out
11 1 4
限制
每个测试点 1s。
【数据说明】
对于 100%的数据，1≤ n ≤ 1,000,000，0 ≤ x ≤ 9。
表达式求值（noip2013 普及组第二题）
(expr.cpp/c/pas)
描述
给定一个只包含加法和乘法的算术表达式,请你编程计算表达式的值。
【输入】
输入文件为 expr.in。
输入仅有一行,为需要你计算的表达式,表达式中只包含数字、加法运算符“+”和乘 ，且没有括号，所有参与运算的数字均为 0 到 231-1 之间的整数。输入数据保 法运算符“”
证这一行只有 0~ 9、+、这 12 种字符。
【输出】
输出文件名为 expr.out。

输出只有一行，包含一个整数，表示这个表达式的值。注意：当答案长度多于 4 位时，
请只输出最后 4 位，前导 0 不输出。
【输入输出样例 1】
expr.in expr.out
1+13+4 8
【输入输出样例 2】
expr.in expr.out
1+12345678901 7891
【输入输出样例 3】
expr.in expr.out
1+1000000003*1 4
【输入输出样例说明】
样例 1 计算的结果为 8，直接输出 8。
样例 2 计算的结果为 1234567891，输出后 4 位，即 7891。
样例 3 计算的结果为 1000000004，输出后 4 位，即 4。
【数据范围】
对于 30%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100；
对于 80%的数据，0≤表达式中加法运算符和乘法运算符的总数≤1000；
对于 100%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100000。
小朋友的数字（noip2013 普及组第三题）(number.cpp/c/pas)
描述
有 n 个小朋友排成一列。每个小朋友手上都有一个数字,这个数字可正可负。规定每个小朋友的特征值等于排在他前面(包括他本人)的小朋友中连续若干个(最少有一个)小朋友手上的数字之和的最大值。 作为这些小朋友的老师,你需要给每个小朋友一个分数,分数是这样规定的:第一个小朋友的分数是他的特征值,其它小朋友的分数为排在他前面的所有小朋友中(不包括他本人),小朋友分数加上其特征值的最大值。
请计算所有小朋友分数的最大值,输出时保持最大值的符号,将其绝对值对 p 取模后输出。
格式
【输入】
输入文件为 number.in。
第一行包含两个正整数 n、p，之间用一个空格隔开。
第二行包含 n 个数，每两个整数之间用一个空格隔开，表示每个小朋友手上的数字。
【输出】
输出文件名为 number.out。
输出只有一行，包含一个整数，表示最大分数对 p 取模的结果。
【输入输出样例 1】
number.in number.out
5 997 21
1 2 3 4 5
【输入输出样例说明】

小朋友的特征值分别为 1、3、6、10、15，分数分别为 1、2、5、11、21，最大值 21
对 997 的模是 21。
【输入输出样例 2】

number.in number.out
5 7 -1
-1 -1 -1 -1 -1
【输入输出样例说明】
小朋友的特征值分别为-1、-1、-1、-1、-1，分数分别为-1、-2、-2、-2、-2，最大值 -1 对 7 的模为-1，输出-1。
【数据范围】
对于 50%的数据，1 ≤ n ≤ 1,000，1 ≤ p ≤ 1,000 所有数字的绝对值不超过 1000；
99 对于 100%的数据，1 ≤ n ≤ 1,000,000， 1≤ p ≤ 10， 其他数字的绝对值均不超过 10。
车站分级（NOIP 普及组 2013 第四题）(level.cpp/c/pas)
描述
一条单向的铁路线上,依次有编号为 1, 2, ..., n 的 n 个火车站。每个火车站都有一个级别,最低为 1 级。现有若干趟车次在这条线路上行驶,每一趟都满足如下要求:如果这趟车次停靠了火车站 x,则始发站、终点站之间所有级别大于等于火车站 x 的都必须停靠。
(注意:起始站和终点站自然也算作事先已知需要停靠的站点)
例如,下表是 5 趟车次的运行情况。其中,前 4 趟车次均满足要求,而第 5 趟车次由于停靠了 3 号火车站(2 级)却未停靠途经的 6 号火车站(亦为 2 级)而不满足要求。
现有 m 趟车次的运行情况（全部满足要求），试推算这 n 个火车站至少分为几个不同的 级别。
【输入】
输入文件为 level.in。
第一行包含 2 个正整数 n, m，用一个空格隔开。
第 i + 1 行（1 ≤ i ≤ m）中，首先是一个正整数 si（2 ≤ si ≤ n），表示第 i 趟车次有 si 个停
靠站；接下来有 si 个正整数，表示所有停靠站的编号，从小到大排列。每两个数之间用一个 空格隔开。输入保证所有的车次都满足要求。
【输出】
输出文件为 level.out。
输出只有一行，包含一个正整数，即 n 个火车站最少划分的级别数。

【输入输出样例】
【数据范围】
对于 20%的数据，1 ≤ n, m ≤ 10； 对于 50%的数据，1 ≤ n, m ≤ 100； 对于 100%的数据，1 ≤ n, m ≤ 1000。
第 1 题，太弱了，不用骗，得 100 分。

第 2 题，数据很大，但是可以直接输入一个数，输出它 mod 10000 的值。得 10 分。 第 3 题，是一道非常基础的 DP，但对于不知 DP 为何物的蒟蒻来说，就使用暴力算法（即 DFS）。得 20 分。 第 4 题，我们可以寻找一下数据的规律，你会发现，在所有样例中，M 值即为答案。所以直接输出 M，得 10 分。 这样下来，一共得 140 分，比一等分数线还高 20 分！你的信心一定会得到鼓舞的。这就是骗分的神奇。 第 9 章 结语
骗分是蒟蒻的有力武器，可以在比赛中骗得大量分数。相信大家在这本书中收获了很多，希望本书能帮助你多得一些分。
但是，最后我还是要说一句：
不骗分，是骗分的最高境界。
作者：大神 cyd QQ：724612372 新 版 骗 分 导 论 THE NEW GUIDE OF CHEATING IN INFORMATICS OLYMPIAD 蒟 蒻 的 宝 书 目录 第 1 章 绪论 第 2 章 从无解出发 2.1 无解情况 2.2 样例——白送的分数 第 3 章 “艰苦朴素永不忘” 3.1 模拟 3.2 万能钥匙——DFS 第 4 章 骗分的关键——猜想 4.1 听天由命 4.2 猜测答案 4.3 寻找规律 4.4 小数据杀手——打表 第 5 章 做贪心的人 5.1 贪心的算法 5.2 贪心地得分 第 6 章 C++的福利 6.1 快速排序 6.2 “如意金箍棒” 第 7 章 “宁为玉碎，不为瓦全” 第 8 章 实战演练 第 9 章 结语
第 1 章 绪论
在 Oier 中，有一句话广为流传：
任何蒟蒻必须经过大量的刷题练习才能成为大牛乃至于神牛。
这就是著名的 lzn 定理。然而，我们这些蒟蒻们，没有经过那么多历练，却要和大牛们同场竞技，我们该怎么以弱胜强呢？答案就是：
骗分
那么，骗分是什么呢？骗分就是用简单的程序（比标准算法简单很多，保证蒟蒻能轻松搞定的程序），尽可能多得骗取分数。
让我们走进这本《新版骗分导论》，来学习骗分的技巧，来挑战神牛吧！

第 2 章 从无解出发
2.1 无解情况
在很多题目中都有这句话：“若无解，请输出-1.”
看到这句话时，骗分的蒟蒻们就欣喜若狂，因为——数据中必定会有无解的情况！那么，只要打出下面这个程序：
printf(“-1”);
就能得到 10 分，甚至 20 分，30 分！

举个例子：
NOIP2012 第 4 题，文化之旅
题目描述 Description
有一位使者要游历各国，他每到一个国家，都能学到一种文化，但他不愿意学习任何一种文化超过一次（即如果他学习了某种文化，则他就不能到达其他有这种文化的国家）。不同的国家可能有相同的文化。不同文化的国家对其他文化的看法不同，有些文化会排斥外来文化（即如果他学习了某种文化，则他不能到达排斥这种文化的其他国家）。
现给定各个国家间的地理关系，各个国家的文化，每种文化对其他文化的看法，

以及这位使者游历的起点和终点（在起点和终点也会学习当地的文化），国家间的道路距离，试求从起点到终点最少需走多少路。
输入描述 Input Description
第一行为五个整数 N，K，M，S，T，每两个整数之间用一个空格隔开，依次代表国家个数（国家编号为 1 到 N），文化种数（文化编号为 1 到 K），道路的条数，以及起点和终点的编号（保证 S 不等于 T）；
第二行为 N 个整数，每两个整数之间用一个空格隔开，其中第 i 个数 Ci，表示国家 i 的文化为 Ci。
接下来的 K 行，每行 K 个整数，每两个整数之间用一个空格隔开，记第 i 行的第 j 个数为 aij，aij= 1 表示文化 i 排斥外来文化 j（i 等于 j 时表示排斥相同文化的外来人），aij= 0 表示不排斥（注意 i 排斥 j 并不保证 j 一定也排斥 i）。
接下来的 M 行，每行三个整数 u，v，d，每两个整数之间用一个空格隔开，表示国家 u 与国家 v 有一条距离为 d 的可双向通行的道路（保证 u 不等于 v，两个国家之间可能有多条道路）。
输出描述 Output Description
输出只有一行，一个整数，表示使者从起点国家到达终点国家最少需要走的距离数（如果无解则输出-1）。
样例输入 Sample Input
输入样例 1
2 2 1 1 2
1 2
0 1
1 0
1 2 10
输入样例 2
2 2 1 1 2
1 2
0 1
0 0
1 2 10
样例输出 Sample Output
输出样例 1
-1
输出样例 2
10
数据范围及提示 Data Size & Hint
【输入输出样例 1 说明】
由于到国家 2 必须要经过国家 1，而国家 2 的文明却排斥国家 1 的文明，所以不可能到达国家 2。
【输入输出样例 2 说明】
路线为 1 -> 2。
【数据范围】
对于 20%的数据，有 2≤N≤8，K≤5；
对于 30%的数据，有 2≤N≤10，K≤5；

对于 50%的数据，有 2≤N≤20，K≤8；
对于 70%的数据，有 2≤N≤100，K≤10；
对于 100%的数据，有 2≤N≤100，1≤K≤100，1≤M≤N2，1≤ki≤K，1≤u,v≤N，1≤d≤1000，S≠T，1 ≤S, T≤N。

这道题看起来很复杂，但其中有振奋人心的一句话“输出-1”，我考试时就高兴坏了（当时我才初一，水平太烂），随手打了个 printf(“-1”);,得 10 分。

2.2 样例——白送的分数
每道题目的后面，都有一组“样例输入”和“样例输出”。它们的价值极大，不仅能初步帮你检验程序的对错（特别坑的样例除外），而且，如果你不会做这道题（这种情况蒟蒻们已经司空见惯了），你就可以直接输出样例！
例如美国的 USACO，它的题目有一个规则，就是第一组数据必须是样例。那么，只要你输出所有的样例，你就能得到 100 分（满分 1000）！这是相当可观的分数了。

现在，你已经掌握了最基础的骗分技巧。只要你会基本的输入输出语句，你就能实现这些骗分方法。那么，如果你有一定的基础，请看下一章——我将教你怎样用简单方法骗取部分分数。

第 3 章 “艰苦朴素永不忘”
本章的标题来源于《学习雷锋好榜样》的一句歌词，但我不是想教导你们学习雷锋精神，而是学习骗分！
看到“朴素”两个字了吗？它们代表了一类算法，主要有模拟和 DFS。下面我就来介绍它们在骗分中的应用。
3.1 模拟
所谓模拟，就是用计算机程序来模拟实际的事件。例如 NOIP2012 的“寻宝”，就是写一个程序来模拟小明上藏宝塔的动作。
较繁的模拟就不叫骗分了，我这里也不讨论这个问题。
模拟主要可以应用在骗高级数据结构题上的分，例如线段树。下面举一个例子来说明一下。
排 队(USACO 2007 January Silver)
【问题描述】
每天，农夫约翰的 N（1≤N≤50000）头奶牛总是按同一顺序排好队，有一天，约翰决定让一些牛玩一场飞盘游戏（Ultimate Frisbee），他决定在队列里选择一群位置连续的奶牛进行比赛，为了避免比赛结果过于悬殊，要求挑出的奶牛身高不要相差太大。
约翰准备了 Q（1≤Q≤200000）组奶牛选择，并告诉你所有奶牛的身高 Hi（1≤ Hi ≤106）。他想知道每组里最高的奶牛和最矮的奶牛身高差是多少。
注意：在最大的数据上，输入输出将占据大部分时间。
【输入】
第一行，两个用空格隔开的整数 N 和 Q。
第 2 到第 N+1 行，每行一个整数，第 i+1 行表示第 i 头奶牛的身高 Hi
第 N+2 到第 N+Q+1 行，每行两个用空格隔开的整数 A 和 B，表示选择从 A 到 B 的所有牛（1 ≤ A ≤ B ≤ N）

【输出】
共 Q 行，每行一个整数，代表每个询问的答案。
输入样例 输出样例
6 3
1
7
3
4
2
5
1 5
4 6
2 2 6
3
0

对于这个例子，大牛们可以写个线段树，而我们蒟蒻，就模拟吧。
附模拟程序：
```
for(int i=1;i<=q;i++){
scanf(“%d%d”,&a,&b);
int min=INT_MAX,max=INT_MIN;
for(int i=a;i<=b;i++){
if(h[i]<min)min=h[i];
if(h[i]>max)max=h[i];
}
printf(“%d\n”,max-min);
}
```
程序简洁明了，并且能高效骗分。本程序得 50 分。
3.2 万能钥匙——DFS
DFS 是图论中的重要算法，但我们看来，图论神马的都是浮云，关键就是如何骗分。下面引出本书的第 2 条定理：
DFS 是万能的。
这对于你的骗分是至关重要的。比如说，一些动态规划题，可以 DFS；数学题，可以 DFS；剪枝的题，更能 DFS。下面以一道省选题为例，解释一下 DFS 骗分。
例题：NOIP2003，采药
题目描述 Description
辰辰是个天资聪颖的孩子，他的梦想是成为世界上最伟大的医师。为此，他想拜附近最有威望的医师为师。医师为了判断他的资质，给他出了一个难题。医师把他带到一个到处都是草药的山洞里对他说：“孩子，这个山洞里有一些不同的草药，采每一株都需要一些时间，每一株也有它自身的价值。我会给你一段时间，在这段时间里，你可以采到一些草药。如果你是一个聪明的孩子，你应该可以让采到的草药的总价值最大。”
如果你是辰辰，你能完成这个任务吗？
输入描述 Input Description

输入第一行有两个整数 T（1<=T<=1000）和 M（1<=M<=100），用一个空格隔开，T 代表总共能够用来采药的时间，M 代表山洞里的草药的数目。接下来的 M 行每行包括两个在 1 到 100 之间（包括 1 和 100）的整数，分别表示采摘某株草药的时间和这株草药的价值。
输出描述 Output Description
输出包括一行，这一行只包含一个整数，表示在规定的时间内，可以采到的草药的最大总价值。
样例输入 Sample Input
70 3
71 100
69 1
1 2
样例输出 Sample Output
3
数据范围及提示 Data Size & Hint
对于 30%的数据，M<=10；
对于全部的数据，M<=100。
这题的方法很简单。我们瞄准 20%的数据来做，可以用 DFS 枚举方案，然后模拟计算出最优解。附一个大致的代码：

```
void DFS(int d,int c){
if(d==n){if(c>ans)ans=c; return;}
DFS(d+1,c+w[i]);
DFS(d+1,c);
}
```

第 4 章 骗分的关键——猜想

4.1 听天由命

如果你觉得你的人品很好，可以试试这一招——输出随机数。
先看一下代码：

```
#include<stdlib.h>
#include<time.h>
//以上两个头文件必须加
srand(time(NULL));
//输出随机数前执行此语句
printf(“%d”,rand()%X);
//输出一个 0~X-1 的随机整数。
```

这种方法适用于输出一个整数（或判断是否）的题目中，答案的范围越小越好。让老天决定你的得分吧。
据说，在 NOIP2013 中，有人最后一题不会，愤然打了个随机数，结果得了 70 分啊!!

4.2 猜测答案

有些时候，问题的答案可能很有特点：对于大多数情况，答案是一样的。这时，骗分就该出手了。你需要做的，就是发掘出这个答案，然后直接输出。
有时，你需要运用第 3 章中学到的知识，先写出朴素算法，然后造一些数据，可

能就会发现规律。
例如，本班月赛中有一道题：
炸毁计划
【问题描述】
皇军侵占了通往招远的黄金要道。为了保护渤海通道的安全，使得黄金能够顺利地运送到敌后战略总指挥地延安，从而购买战需武器，所以我们要通过你的程序确定这条战略走廊是否安全。
已知我们有 N 座小岛，只有使得每一个小岛都能与其他任意一个小岛联通才能保证走廊的安全。每个小岛之间只能通过若干双向联通的桥保持联系，已知有 M 座桥(Ai,Bi)表示第 i 座桥连接了 Ai 与 Bi 这两座城市。
现在，敌人的炸药只能炸毁其中一座桥，请问在仅仅炸毁这一座桥的情况下，能否保证所有岛屿安全，都能联通起来。
现在给出 Q 个询问 Ci，其中 Ci 表示桥梁编号，桥梁的编号按照输入顺序编号。每个询问表示在仅仅炸毁第 Ci 座桥的情况下能否保证所有岛屿安全。如果可以，在输出文件当中，对应输入顺序输出 yes，否则输出 no（输出为半角英文单词，区分大小写，默认为小写，不含任何小写符号，每行输出一个空格，忽略文末空格）。
【输入格式】
第一行 三个整数 N，M，Q，分别表示岛屿的个数，桥梁的个数和询问的个数。
第二行到第 M+1 行 每行两个整数。第 i+1 行有两个整数 Ai Bi 表示这个桥梁的属性。
第 M+2 行 有 Q 个整数 Ci 表示查询。
【输出格式】
Q 行，表示查询结果。
【样例】
destroy.in destroy.out
2 1 1
1 2
1 no
【样例范围】
对于 80%的数据，N≤100。
对于 100%的数据，N≤1000，N,Q≤M≤2000 。

你发现问题了吗？那么多座桥，炸一座就破坏岛屿的联系，可能性微乎其微（除非特别设计数据）。那么，我们的骗分策略就出来了：对于所有询问，输出 yes.果然，此算法效果不错，得 80 分。
现在知道猜测答案的厉害了吧？
4.3 寻找规律
首先声明：本节讲的规律不是正当的算法规律，而是数据的特点。
某些题目会给你很多样例，你就可以观察他们的特点了。有时，数据中的某一个（或几个）数，能通过简单的关系直接算出答案。
只要你找到了规律，在很多情况下你都能得到可观的分数。
这样的题目大多出现在 NOI 或更高等级的比赛中，本人蒟蒻一个，就不举例了。
传说某人去省选时专门琢磨数据的规律，结果有一题得了 30 分。

4.4 小数据杀手——打表
我认识一个人，他在某老师家上 C 语言家教，老师每讲一题，他都喊一句：“打表行吗？”
他真的是打表的忠实粉丝。表虽然不能乱打，但还是很有用的。
先看一个例子：NOIP2003 栈
题目描述 Description
栈是计算机中经典的数据结构，简单的说，栈就是限制在一端进行插入删除操作的线性表。
栈有两种最重要的操作，即 pop（从栈顶弹出一个元素）和 push（将一个元素进栈）。
栈的重要性不言自明，任何一门数据结构的课程都会介绍栈。宁宁同学在复习栈的基本概念时，想到了一个书上没有讲过的问题，而他自己无法给出答案，所以需要你的帮忙
宁宁考虑的是这样一个问题：一个操作数序列，从 1，2，一直到 n（图示为 1 到 3 的情况），栈 A 的深度大于 n。
现在可以进行两种操作，
1.将一个数，从操作数序列的头端移到栈的头端（对应数据结构栈的 push 操作）
2. 将一个数，从栈的头端移到输出序列的尾端（对应数据结构栈的 pop 操作）
使用这两种操作，由一个操作数序列就可以得到一系列的输出序列，下图所示为由 1 2 3 生成序列 2 3 1 的过程。（原始状态如上图所示） 。
你的程序将对给定的 n，计算并输出由操作数序列 1，2，…，n 经过操作可能得到的输出序列的总数。
输入描述 Input Description
输入文件只含一个整数 n（1≤n≤18）
输出描述 Output Description
输出文件只有一行，即可能输出序列的总数目
样例输入 Sample Input
3
样例输出 Sample Output
5

这题看似复杂，但数据范围太小，N<=18。所以，骗分程序就好写了：

```
int
a[18]={1,2,5,14,42,132,429,1430,4862,16796,58786,208012,742900,2674440,9694845,35357670,129644790,477638700};

scanf(“%d”,&n):
printf(“%d”,ans[n-1]);
```

测试结果不言而喻，AC 了。

学完这一章，你已基本掌握了骗分技巧。下面的内容涉及一点算法知识，难度有所增加。蒟蒻中的蒟蒻可以止步于此了。

第 5 章 做贪心的人
5.1 贪心的算法

给你一堆纸币，让你挑一张，相信你一定会挑面值最大的。其实，这就是贪心算法。
贪心算法是个复杂的问题，但你不用管那么多。我们只关心骗分。给你一个问题，让你从一些东西中选出一些，你就可以使用贪心的方法，尽量挑好的。
举个例子：这是我们的市队选拔的一道题。
2. 有趣的问题
【问题描述】
2013 年的 NOIP 结束后， Smart 发现自己又被题目碾压了，心里非常地不爽，于是
暗下决心疯狂地刷数学题目，做到天昏地暗、废寝忘食，准备在今年的中考中大展身手。
有一天，他在做题时发现了一个有趣的问题：
给定 n 个二元组(ai, bi) i)，记函数：
y=100*sigma(ai)/sigma(bi);
将函数 y 的值四舍五入取整。
现将 n 个二元组去掉其中的 k 个计算一个新的 y 值（也四舍五入取整），均能满足：y <= z ，求出最小的 z 值。Smart 想让你帮他一起找出最小的 z 值。
【输入格式】
输入包含多组测试数据。每组测试数据第一行两个整数：n 和 k；第二行为 n 个数：
a1 a2 …… an；第三行为；n 个数： b1 b2 …… bn。
输入数据当 n、k 均为 0 时结束。
【输出格式】
对于每组测试数据输出一行，即找出的最小的尤值。
注意：为避免精度四舍五入出现误差，测试点保证每个函数值与最终结果的差值至
少为 0.001 。
【样例】
math.in
3 1
5 0 1
5 1 6
4 2
1 2 7 9
5 6 7 9
0 0
math. out
83
100
【数据范围】
对于 40% 的数据： n≤20;
对于 70% 的数据： n≤1000;
对于 100% 的数据： n≤10000，ai,bi 都在 int 范围内。

这题让人望而生畏，但我们有贪心的手段。每个二元组的 a 值是乘到答案中的，所以 a 越大越好，那么只要选择出最小的 k 个去掉即可。代码就不写了，因为这个设计到下一章的内容：排序。
此代码得 20 分。
5.2 贪心地得分
我们已经学了很多骗分方法，但他们中的大多效率并不高，一般能骗 10~20 分。这不能满足我们的贪心。
然而，我们可以合成骗分的程序。举个最简单的例子，有些含有无解情况的题目，它们同样有样例。我们可以写这个程序
```
if(是样例)printf(样例);
else printf(“-1”);
```
这样也许能变 10 分为 20 分，甚至更多。
当然，合并骗分方法时要注意，不要重复骗同一种情况，或漏考虑一些情况。
大量能骗分的问题都能用此法，大家可以试试用新方法骗 2.1 中的例子“文化之旅”。
第 6 章 C++的福利
（请 P 党们跳过本章，这不是你们的福利）
在 C++中，有一个好东西，名唤 STL，被万千 Oier 们所崇拜，所喜爱。下面让我们走进 STL。
6.1 快速排序
快速排序是一个经典算法，也是 C++党的经典福利。他们有这样的代码：
```
#include<algorithm>//这是必须的
sort(A,A+n);//对一个下标从 0 开始存储，长度为 n 的数组升序排序
```
就这么简单，完成了 P 党一大堆代码干的事情。
6.2 “如意金箍棒”
C++里有一种东西，叫 vector 容器。它好比如意金箍棒，可以随着元素的数量而改变大小。它其实就是数组，却比数组强得多。
下面看看它的几种操作：
```
vector<int> V;//定义
V.push_back(x);//末尾增加一个元素 x
V.pop_back();//末尾删除一个元素
V.size();//返回容器中的元素个数
```
它同样可以使用下标访问。（从 0 开始）
第 7 章 “宁为玉碎，不为瓦全”
至此，我已介绍完了我所知的骗分方法。如果上面的方法都不奏效，我也无能为力。但是，我还有最后一招——
有句古话说：“宁为玉碎，不为瓦全”。我们蒟蒻也应有这样的精神。骗不到分，就报复一下，卡评测以泄愤吧！
卡评测主要有两种方法：一是死循环，故意超时；二是进入终端，卡住编译器。先介绍下第一种。代码很简单，请看：
```
while(1);
```
就是这短短一句话，就能卡住评测机长达 10s，20s，甚至更多！对于测试点多、时限长的题目，这是个不错的方法。
第二种方法也很简单，但危害性较大，建议不要在重要比赛中使用，否则可能让

你追悔莫及。它就是：
#include<con> （windows 系统中使用）
或 #include</dev/console> （Linux 系统中使用）
它非常强大，可以卡住评测系统，使其永远停止不了编译你的程序。唯一的解除方法是，工作人员强行关机，重启，重测。当然，我不保证他们不会气愤地把你的成绩变成 0 分。请慎用此方法。
第 8 章 实战演练
下面我们来做一些习题，练习骗分技巧。
我们来一起分析一下 NOIP2013 普及组的试题吧。
记数问题（NOIP 普及组 2013 第一题）
(count.cpp/c/pas)
描述
试计算在区间 1 到 n 的所有整数中,数字 x(0 ≤ x ≤ 9)共出现了多少次?例如,在 1 到 11 中,即在 1、2、3、4、5、6、7、8、9、10、11 中,数字 1 出现了 4 次。
【输入】
输入文件名为 count.in。
输入共 1 行，包含 2 个整数 n、x，之间用一个空格隔开
【输出】
输出文件名为 count.out。
输出共 1 行，包含一个整数，表示 x 出现的次数。
【输入输出样例】
count.in count.out
11 1 4
限制
每个测试点 1s。
【数据说明】
对于 100%的数据，1≤ n ≤ 1,000,000，0 ≤ x ≤ 9。
表达式求值（noip2013 普及组第二题）
(expr.cpp/c/pas)
描述
给定一个只包含加法和乘法的算术表达式,请你编程计算表达式的值。
【输入】
输入文件为 expr.in。
输入仅有一行，为需要你计算的表达式，表达式中只包含数字、加法运算符“+”和乘 ，且没有括号，所有参与运算的数字均为 0 到 231-1 之间的整数。输入数据保 法运算符“*”
证这一行只有 0~ 9、+、*这 12 种字符。
【输出】
输出文件名为 expr.out。
输出只有一行，包含一个整数，表示这个表达式的值。注意：当答案长度多于 4 位时，

请只输出最后 4 位，前导 0 不输出。
【输入输出样例 1】
expr.in expr.out
1+1*3+4 8
【输入输出样例 2】
expr.in expr.out
1+1234567890*1 7891
【输入输出样例 3】
expr.in expr.out
1+1000000003*1 4
【输入输出样例说明】
样例 1 计算的结果为 8，直接输出 8。
样例 2 计算的结果为 1234567891，输出后 4 位，即 7891。
样例 3 计算的结果为 1000000004，输出后 4 位，即 4。
【数据范围】
对于 30%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100；
对于 80%的数据，0≤表达式中加法运算符和乘法运算符的总数≤1000；
对于 100%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100000。
小朋友的数字（noip2013 普及组第三题）(number.cpp/c/pas)
描述
有 n 个小朋友排成一列。每个小朋友手上都有一个数字,这个数字可正可负。规定每个小朋友的特征值等于排在他前面(包括他本人)的小朋友中连续若干个(最少有一个)小朋友手上的数字之和的最大值。 作为这些小朋友的老师,你需要给每个小朋友一个分数,分数是这样规定的:第一个小朋友的分数是他的特征值,其它小朋友的分数为排在他前面的所有小朋友中(不包括他本人),小朋友分数加上其特征值的最大值。
请计算所有小朋友分数的最大值,输出时保持最大值的符号,将其绝对值对 p 取模后输出。
格式
【输入】
输入文件为 number.in。
第一行包含两个正整数 n、p，之间用一个空格隔开。
第二行包含 n 个数，每两个整数之间用一个空格隔开，表示每个小朋友手上的数字。
【输出】
输出文件名为 number.out。
输出只有一行，包含一个整数，表示最大分数对 p 取模的结果。
【输入输出样例 1】
number.in number.out
5 997 21
1 2 3 4 5
【输入输出样例说明】
小朋友的特征值分别为 1、3、6、10、15，分数分别为 1、2、5、11、21，最大值 21

对 997 的模是 21。
【输入输出样例 2】

number.in number.out
5 7 -1
-1 -1 -1 -1 -1
【输入输出样例说明】
小朋友的特征值分别为-1、-1、-1、-1、-1，分数分别为-1、-2、-2、-2、-2，最大值 -1 对 7 的模为-1，输出-1。
【数据范围】
对于 50%的数据，1 ≤ n ≤ 1,000，1 ≤ p ≤ 1,000 所有数字的绝对值不超过 1000；
99 对于 100%的数据，1 ≤ n ≤ 1,000,000， 1≤ p ≤ 10， 其他数字的绝对值均不超过 10。
车站分级（NOIP 普及组 2013 第四题）(level.cpp/c/pas)
描述
一条单向的铁路线上,依次有编号为 1, 2, ..., n 的 n 个火车站。每个火车站都有一个级别,最低为 1 级。现有若干趟车次在这条线路上行驶,每一趟都满足如下要求:如果这趟车次停靠了火车站 x,则始发站、终点站之间所有级别大于等于火车站 x 的都必须停靠。
(注意:起始站和终点站自然也算作事先已知需要停靠的站点)
例如,下表是 5 趟车次的运行情况。其中,前 4 趟车次均满足要求,而第 5 趟车次由于停靠了 3 号火车站(2 级)却未停靠途经的 6 号火车站(亦为 2 级)而不满足要求。
现有 m 趟车次的运行情况（全部满足要求），试推算这 n 个火车站至少分为几个不同的 级别。
【输入】
输入文件为 level.in。
第一行包含 2 个正整数 n, m，用一个空格隔开。
第 i + 1 行（1 ≤ i ≤ m）中，首先是一个正整数 si（2 ≤ si ≤ n），表示第 i 趟车次有 si 个停
靠站；接下来有 si 个正整数，表示所有停靠站的编号，从小到大排列。每两个数之间用一个 空格隔开。输入保证所有的车次都满足要求。
【输出】
输出文件为 level.out。
输出只有一行，包含一个正整数，即 n 个火车站最少划分的级别数。

【输入输出样例】

【数据范围】
对于 20%的数据，1 ≤ n, m ≤ 10； 对于 50%的数据，1 ≤ n, m ≤ 100； 对于 100%的数据，1 ≤ n, m ≤ 1000。

第 1 题，太弱了，不用骗，得 100 分。
第 2 题，数据很大，但是可以直接输入一个数，输出它 mod 10000 的值。得 10 分。
第 3 题，是一道非常基础的 DP，但对于不知 DP 为何物的蒟蒻来说，就使用暴力算法（即 DFS）。得 20 分。
第 4 题，我们可以寻找一下数据的规律，你会发现，在所有样例中，M 值即为答案。所以直接输出 M，得 10 分。
这样下来，一共得 140 分，比一等分数线还高 20 分！你的信心一定会得到鼓舞的。这就是骗分的神奇。
第 9 章 结语
骗分是蒟蒻的有力武器，可以在比赛中骗得大量分数。相信大家在这本书中收获了很多，希望本书能帮助你多得一些分。
但是，最后我还是要说一句：
不骗分，是骗分的最高境界。
作者：大神 cyd
QQ：724612372
新 版 骗 分 导 论
THE NEW GUIDE OF CHEATING IN INFORMATICS OLYMPIAD
蒟 蒻 的 宝 书
目录

第 1 章 绪论
在 Oier 中，有一句话广为流传： 任何蒟蒻必须经过大量的刷题练习才能成为大牛乃至于神牛。 这就是著名的 lzn 定理。然而，我们这些蒟蒻们，没有经过那么多历练，却要和大牛们同场竞技，我们该怎么以弱胜强呢？答案就是： 骗分 那么，骗分是什么呢？骗分就是用简单的程序（比标准算法简单很多，保证蒟蒻能轻松搞定的程序），尽可能多得骗取分数。 让我们走进这本《新版骗分导论》，来学习骗分的技巧，来挑战神牛吧！
第 2 章 从无解出发
2.1 无解情况
在很多题目中都有这句话：“若无解，请输出-1.”
看到这句话时，骗分的蒟蒻们就欣喜若狂，因为——数据中必定会有无解的情况！那么，只要打出下面这个程序：
```
printf(“-1”);
```
就能得到 10 分，甚至 20 分，30 分！

举个例子：
NOIP2012 第 4 题，文化之旅
题目描述 Description
有一位使者要游历各国，他每到一个国家，都能学到一种文化，但他不愿意学习任何一种文化超过一次（即如果他学习了某种文化，则他就不能到达其他有这种文化的国家）。不同的国家可能有相同的文化。不同文化的国家对其他文化的看法不同，有些文化会排斥外来文化（即如果他学习了某种文化，则他不能到达排斥这种文化的其他国家）。
现给定各个国家间的地理关系，各个国家的文化，每种文化对其他文化的看法，以及这位使者游历的起点和终点（在起点和终点也会学习当地的文化），国家间的道路距离，试求从起点到终点最少需走多少路。
输入描述 Input Description
第一行为五个整数 N，K，M，S，T，每两个整数之间用一个空格隔开，依次代表国家个数（国家编号为 1 到 N），文化种数（文化编号为 1 到 K），道路的条数，以及起点和终点的编号（保证 S 不等于 T）；
第二行为 N 个整数，每两个整数之间用一个空格隔开，其中第 i 个数 Ci，表示国家 i 的文化为 Ci。
接下来的 K 行，每行 K 个整数，每两个整数之间用一个空格隔开，记第 i 行的第 j 个数为 aij，aij= 1 表示文化 i 排斥外来文化 j（i 等于 j 时表示排斥相同文化的外来人），aij= 0 表示不排斥（注意 i 排斥 j 并不保证 j 一定也排斥 i）。
接下来的 M 行，每行三个整数 u，v，d，每两个整数之间用一个空格隔开，表示国家 u 与国家 v 有一条距离为 d 的可双向通行的道路（保证 u 不等于 v，两个国家之间可能有多条道路）。
输出描述 Output Description
输出只有一行，一个整数，表示使者从起点国家到达终点国家最少需要走的距离数（如果无解则输出-1）。
样例输入 Sample Input
输入样例 1
2 2 1 1 2
1 2
0 1
1 0
1 2 10
输入样例 2
2 2 1 1 2
1 2
0 1
0 0
1 2 10
样例输出 Sample Output
输出样例 1
-1
输出样例 2
10

数据范围及提示 Data Size & Hint
【输入输出样例 1 说明】
由于到国家 2 必须要经过国家 1，而国家 2 的文明却排斥国家 1 的文明，所以不可能到达国家 2。
【输入输出样例 2 说明】
路线为 1 -> 2。
【数据范围】
对于 20%的数据，有 2≤N≤8，K≤5；
对于 30%的数据，有 2≤N≤10，K≤5；
对于 50%的数据，有 2≤N≤20，K≤8；
对于 70%的数据，有 2≤N≤100，K≤10；
对于 100%的数据，有 2≤N≤100，1≤K≤100，1≤M≤N2，1≤ki≤K，1≤u,v≤N，1≤d≤1000，S≠T，1 ≤S, T≤N。
这道题看起来很复杂，但其中有振奋人心的一句话“输出-1”，我考试时就高兴坏了（当时我才初一，水平太烂），随手打了个 printf(“-1”);,得 10 分。
2.2 样例——白送的分数
每道题目的后面，都有一组“样例输入”和“样例输出”。它们的价值极大，不仅能初步帮你检验程序的对错（特别坑的样例除外），而且，如果你不会做这道题（这种情况蒟蒻们已经司空见惯了），你就可以直接输出样例！ 例如美国的 USACO，它的题目有一个规则，就是第一组数据必须是样例。那么，只要你输出所有的样例，你就能得到 100 分（满分 1000）！这是相当可观的分数了。
现在，你已经掌握了最基础的骗分技巧。只要你会基本的输入输出语句，你就能实现这些骗分方法。那么，如果你有一定的基础，请看下一章——我将教你怎样用简单方法骗取部分分数。
第 3 章 “艰苦朴素永不忘”
本章的标题来源于《学习雷锋好榜样》的一句歌词，但我不是想教导你们学习雷锋精神，而是学习骗分！ 看到“朴素”两个字了吗？它们代表了一类算法，主要有模拟和 DFS。下面我就来介绍它们在骗分中的应用。 3.1 模拟 所谓模拟，就是用计算机程序来模拟实际的事件。例如 NOIP2012 的“寻宝”，就是写一个程序来模拟小明上藏宝塔的动作。 较繁的模拟就不叫骗分了，我这里也不讨论这个问题。 模拟主要可以应用在骗高级数据结构题上的分，例如线段树。下面举一个例子来说明一下。 排 队(USACO 2007 January Silver)
【问题描述】
每天，农夫约翰的 N（1≤N≤50000）头奶牛总是按同一顺序排好队，有一天，约翰决定让一些牛玩一场飞盘游戏（Ultimate Frisbee），他决定在队列里选择一群位置连续的奶牛进行比赛，为了避免比赛结果过于悬殊，要求挑出的奶牛身高不要相差太大。
约翰准备了 Q（1≤Q≤200000）组奶牛选择，并告诉你所有奶牛的身高 Hi（1≤ Hi ≤106）。他想知道每组里最高的奶牛和最矮的奶牛身高差是多少。
注意：在最大的数据上，输入输出将占据大部分时间。
【输入】
第一行，两个用空格隔开的整数 N 和 Q。
第 2 到第 N+1 行，每行一个整数，第 i+1 行表示第 i 头奶牛的身高 Hi
第 N+2 到第 N+Q+1 行，每行两个用空格隔开的整数 A 和 B，表示选择从 A 到 B 的

所有牛（1 ≤ A ≤ B ≤ N）
【输出】
共 Q 行，每行一个整数，代表每个询问的答案。
输入样例 输出样例
6 3
1
7
3
4
2
5
1 5
4 6
2 2 6
3
0
对于这个例子，大牛们可以写个线段树，而我们蒟蒻，就模拟吧。
附模拟程序：

```
for(int i=1;i<=q;i++){
scanf(“%d%d”,&a,&b);
int min=INTMAX,max=INTMIN;
for(int i=a;i<=b;i++){
if(h[i]<min)min=h[i];
if(h[i]>max)max=h[i];
}
printf(“%d\n”,max-min);
}
```

程序简洁明了，并且能高效骗分。本程序得 50 分。
3.2 万能钥匙——DFS DFS 是图论中的重要算法，但我们看来，图论神马的都是浮云，关键就是如何骗分。下面引出本书的第 2 条定理： DFS 是万能的。 这对于你的骗分是至关重要的。比如说，一些动态规划题，可以 DFS；数学题，可以 DFS；剪枝的题，更能 DFS。下面以一道省选题为例，解释一下 DFS 骗分。 例题：NOIP2003，采药
题目描述 Description
辰辰是个天资聪颖的孩子，他的梦想是成为世界上最伟大的医师。为此，他想拜附近最有威望的医师为师。医师为了判断他的资质，给他出了一个难题。医师把他带到一个到处都是草药的山洞里对他说：“孩子，这个山洞里有一些不同的草药，采每一株都需要一些时间，每一株也有它自身的价值。我会给你一段时间，在这段时间里，你可以采到一些草药。如果你是一个聪明的孩子，你应该可以让采到的草药的总价值最大。”
如果你是辰辰，你能完成这个任务吗？
输入描述 Input Description
输入第一行有两个整数 T（1<=T<=1000）和 M（1<=M<=100），用一个空格隔开，T 代表总共能够用来采药的时间，M 代表山洞里的草药的数目。接下来的 M 行每

行包括两个在 1 到 100 之间（包括 1 和 100）的整数，分别表示采摘某株草药的时间和这株草药的价值。
输出描述 Output Description
输出包括一行，这一行只包含一个整数，表示在规定的时间内，可以采到的草药的最大总价值。
样例输入 Sample Input
70 3
71 100
69 1
1 2
样例输出 Sample Output
3
数据范围及提示 Data Size & Hint
对于 30%的数据，M<=10；
对于全部的数据，M<=100。
这题的方法很简单。我们瞄准 20%的数据来做，可以用 DFS 枚举方案，然后模拟计算出最优解。附一个大致的代码：

```
void DFS(int d,int c){
if(d==n){if(c>ans)ans=c; return;}
DFS(d+1,c+w[i]);
DFS(d+1,c);
}
```

第 4 章 骗分的关键——猜想
4.1 听天由命 如果你觉得你的人品很好，可以试试这一招——输出随机数。 先看一下代码：

```
include<stdlib.h>

include<time.h>

//以上两个头文件必须加
srand(time(NULL));
//输出随机数前执行此语句
printf(“%d”,rand()%X);
//输出一个 0~X-1 的随机整数。
```

这种方法适用于输出一个整数（或判断是否）的题目中，答案的范围越小越好。让老天决定你的得分吧。 据说，在 NOIP2013 中，有人最后一题不会，愤然打了个随机数，结果得了 70 分啊!! 4.2 猜测答案 有些时候，问题的答案可能很有特点：对于大多数情况，答案是一样的。这时，骗分就该出手了。你需要做的，就是发掘出这个答案，然后直接输出。 有时，你需要运用第 3 章中学到的知识，先写出朴素算法，然后造一些数据，可能就会发现规律。 例如，本班月赛中有一道题： 炸毁计划
【问题描述】
皇军侵占了通往招远的黄金要道。为了保护渤海通道的安全，使得黄金能够顺利地运送到敌后战略总指挥地延安，从而购买战需武器，所以我们要通过你的程序

确定这条战略走廊是否安全。
已知我们有 N 座小岛，只有使得每一个小岛都能与其他任意一个小岛联通才能保证走廊的安全。每个小岛之间只能通过若干双向联通的桥保持联系，已知有 M 座桥(Ai,Bi)表示第 i 座桥连接了 Ai 与 Bi 这两座城市。
现在，敌人的炸药只能炸毁其中一座桥，请问在仅仅炸毁这一座桥的情况下，能否保证所有岛屿安全，都能联通起来。
现在给出 Q 个询问 Ci，其中 Ci 表示桥梁编号，桥梁的编号按照输入顺序编号。每个询问表示在仅仅炸毁第 Ci 座桥的情况下能否保证所有岛屿安全。如果可以，在输出文件当中，对应输入顺序输出 yes，否则输出 no（输出为半角英文单词，区分大小写，默认为小写，不含任何小写符号，每行输出一个空格，忽略文末空格）。
【输入格式】
第一行 三个整数 N，M，Q，分别表示岛屿的个数，桥梁的个数和询问的个数。
第二行到第 M+1 行 每行两个整数。第 i+1 行有两个整数 Ai Bi 表示这个桥梁的属性。
第 M+2 行 有 Q 个整数 Ci 表示查询。
【输出格式】
Q 行，表示查询结果。
【样例】
destroy.in destroy.out
2 1 1
1 2
1 no
【样例范围】
对于 80%的数据，N≤100。
对于 100%的数据，N≤1000，N,Q≤M≤2000 。
你发现问题了吗？那么多座桥，炸一座就破坏岛屿的联系，可能性微乎其微（除非特别设计数据）。那么，我们的骗分策略就出来了：对于所有询问，输出 yes.果然，此算法效果不错，得 80 分。
现在知道猜测答案的厉害了吧？
4.3 寻找规律 首先声明：本节讲的规律不是正当的算法规律，而是数据的特点。某些题目会给你很多样例，你就可以观察他们的特点了。有时，数据中的某一个（或几个）数，能通过简单的关系直接算出答案。 只要你找到了规律，在很多情况下你都能得到可观的分数。 这样的题目大多出现在 NOI 或更高等级的比赛中，本人蒟蒻一个，就不举例了。传说某人去省选时专门琢磨数据的规律，结果有一题得了 30 分。 4.4 小数据杀手——打表
我认识一个人，他在某老师家上 C 语言家教，老师每讲一题，他都喊一句：“打表行吗？”
他真的是打表的忠实粉丝。表虽然不能乱打，但还是很有用的。
先看一个例子：NOIP2003 栈
题目描述 Description
栈是计算机中经典的数据结构，简单的说，栈就是限制在一端进行插入删除操作的线性表。
栈有两种最重要的操作，即 pop（从栈顶弹出一个元素）和 push（将一个元素进

栈）。
栈的重要性不言自明，任何一门数据结构的课程都会介绍栈。宁宁同学在复习栈的基本概念时，想到了一个书上没有讲过的问题，而他自己无法给出答案，所以需要你的帮忙
宁宁考虑的是这样一个问题：一个操作数序列，从 1，2，一直到 n（图示为 1 到 3 的情况），栈 A 的深度大于 n。
现在可以进行两种操作，
1.将一个数，从操作数序列的头端移到栈的头端（对应数据结构栈的 push 操作）
2. 将一个数，从栈的头端移到输出序列的尾端（对应数据结构栈的 pop 操作）
使用这两种操作，由一个操作数序列就可以得到一系列的输出序列，下图所示为由 1 2 3 生成序列 2 3 1 的过程。（原始状态如上图所示） 。
你的程序将对给定的 n，计算并输出由操作数序列 1，2，…，n 经过操作可能得到的输出序列的总数。
输入描述 Input Description
输入文件只含一个整数 n（1≤n≤18）
输出描述 Output Description
输出文件只有一行，即可能输出序列的总数目
样例输入 Sample Input
3
样例输出 Sample Output
5
这题看似复杂，但数据范围太小，N<=18。所以，骗分程序就好写了：
```
int
a[18]={1,2,5,14,42,132,429,1430,4862,16796,58786,208012,742900,2674440,9694845,35357670,129644790,477638700};
scanf(“%d”,&n):
printf(“%d”,ans[n-1]);
```
测试结果不言而喻，AC 了。
学完这一章，你已基本掌握了骗分技巧。下面的内容涉及一点算法知识，难度有所增加。蒟蒻中的蒟蒻可以止步于此了。
第 5 章 做贪心的人
5.1 贪心的算法 给你一堆纸币，让你挑一张，相信你一定会挑面值最大的。其实，这就是贪心算法。 贪心算法是个复杂的问题，但你不用管那么多。我们只关心骗分。给你一个问题，让你从一些东西中选出一些，你就可以使用贪心的方法，尽量挑好的。 举个例子：这是我们的市队选拔的一道题。 2. 有趣的问题
【问题描述】
2013 年的 NOIP 结束后， Smart 发现自己又被题目碾压了，心里非常地不爽，于是
暗下决心疯狂地刷数学题目，做到天昏地暗、废寝忘食，准备在今年的中考中大展身手。
有一天，他在做题时发现了一个有趣的问题：
给定 n 个二元组(ai, bi) i)，记函数：
y=100*sigma(ai)/sigma(bi);
将函数 y 的值四舍五入取整。

现将 n 个二元组去掉其中的 k 个计算一个新的 y 值（也四舍五入取整），均能满足：y <= z ，求出最小的 z 值。Smart 想让你帮他一起找出最小的 z 值。
【输入格式】
输入包含多组测试数据。每组测试数据第一行两个整数：n 和 k；第二行为 n 个数：
a1 a2 …… an；第三行为；n 个数： b1 b2 …… bn。
输入数据当 n、k 均为 0 时结束。
【输出格式】
对于每组测试数据输出一行，即找出的最小的尤值。
注意：为避免精度四舍五入出现误差，测试点保证每个函数值与最终结果的差值至
少为 0.001 。
【样例】
math.in
3 1
5 0 1
5 1 6
4 2
1 2 7 9
5 6 7 9
0 0
math. out
83
100
【数据范围】
对于 40% 的数据： n≤20;
对于 70% 的数据： n≤1000;
对于 100% 的数据： n≤10000，ai,bi 都在 int 范围内。
这题让人望而生畏，但我们有贪心的手段。每个二元组的 a 值是乘到答案中的，所以 a 越大越好，那么只要选择出最小的 k 个去掉即可。代码就不写了，因为这个设计到下一章的内容：排序。
此代码得 20 分。
5.2 贪心地得分 我们已经学了很多骗分方法，但他们中的大多效率并不高，一般能骗 10~20 分。这不能满足我们的贪心。 然而，我们可以合成骗分的程序。举个最简单的例子，有些含有无解情况的题目，它们同样有样例。我们可以写这个程序 if(是样例)printf(样例);
else printf(“-1”);
这样也许能变 10 分为 20 分，甚至更多。 当然，合并骗分方法时要注意，不要重复骗同一种情况，或漏考虑一些情况。 大量能骗分的问题都能用此法，大家可以试试用新方法骗 2.1 中的例子“文化之旅”。 第 6 章 C++的福利
（请 P 党们跳过本章，这不是你们的福利）
在 C++中，有一个好东西，名唤 STL，被万千 Oier 们所崇拜，所喜爱。下面让我们走进 STL。
6.1 快速排序 快速排序是一个经典算法，也是 C++党的经典福利。他们有这样

的代码： #include<algorithm>//这是必须的 sort(A,A+n);//对一个下标从 0 开始存储，长度为 n 的数组升序排序 就这么简单，完成了 P 党一大堆代码干的事情。 6.2 “如意金箍棒” C++里有一种东西，叫 vector 容器。它好比如意金箍棒，可以随着元素的数量而改变大小。它其实就是数组，却比数组强得多。 下面看看它的几种操作： vector<int> V;//定义 V.pushback(x);//末尾增加一个元素 x
V.popback();//末尾删除一个元素
V.size();//返回容器中的元素个数 它同样可以使用下标访问。（从 0 开始）
第 7 章 “宁为玉碎，不为瓦全”
至此，我已介绍完了我所知的骗分方法。如果上面的方法都不奏效，我也无能为力。但是，我还有最后一招—— 有句古话说：“宁为玉碎，不为瓦全”。我们蒟蒻也应有这样的精神。骗不到分，就报复一下，卡评测以泄愤吧！ 卡评测主要有两种方法：一是死循环，故意超时；二是进入终端，卡住编译器。 先介绍下第一种。代码很简单，请看：
while(1);
就是这短短一句话，就能卡住评测机长达 10s，20s，甚至更多！对于测试点多、时限长的题目，这是个不错的方法。
第二种方法也很简单，但危害性较大，建议不要在重要比赛中使用，否则可能让你追悔莫及。它就是：
include<con> （windows 系统中使用）

或 #include</dev/console> （Linux 系统中使用）
它非常强大，可以卡住评测系统，使其永远停止不了编译你的程序。唯一的解除方法是，工作人员强行关机，重启，重测。当然，我不保证他们不会气愤地把你的成绩变成 0 分。请慎用此方法。
第 8 章 实战演练
下面我们来做一些习题，练习骗分技巧。 我们来一起分析一下 NOIP2013 普及组的试题吧。
记数问题（NOIP 普及组 2013 第一题）
(count.cpp/c/pas)
描述
试计算在区间 1 到 n 的所有整数中,数字 x(0 ≤ x ≤ 9)共出现了多少次?例如,在 1 到 11 中,即在 1、2、3、4、5、6、7、8、9、10、11 中,数字 1 出现了 4 次。
【输入】
输入文件名为 count.in。
输入共 1 行，包含 2 个整数 n、x，之间用一个空格隔开
【输出】
输出文件名为 count.out。
输出共 1 行，包含一个整数，表示 x 出现的次数。
【输入输出样例】
count.in count.out
11 1 4
限制

每个测试点 1s。
【数据说明】
对于 100%的数据，1≤ n ≤ 1,000,000，0 ≤ x ≤ 9。
表达式求值（noip2013 普及组第二题）
(expr.cpp/c/pas)
描述
给定一个只包含加法和乘法的算术表达式,请你编程计算表达式的值。
【输入】
输入文件为 expr.in。
输入仅有一行，为需要你计算的表达式，表达式中只包含数字、加法运算符“+”和乘 ，且没有括号，所有参与运算的数字均为 0 到 231-1 之间的整数。输入数据保 法运算符“”
证这一行只有 0~ 9、+、这 12 种字符。
【输出】
输出文件名为 expr.out。
输出只有一行，包含一个整数，表示这个表达式的值。注意：当答案长度多于 4 位时，
请只输出最后 4 位，前导 0 不输出。
【输入输出样例 1】
expr.in expr.out
1+13+4 8
【输入输出样例 2】
expr.in expr.out
1+12345678901 7891
【输入输出样例 3】
expr.in expr.out
1+1000000003*1 4
【输入输出样例说明】
样例 1 计算的结果为 8，直接输出 8。
样例 2 计算的结果为 1234567891，输出后 4 位，即 7891。
样例 3 计算的结果为 1000000004，输出后 4 位，即 4。
【数据范围】
对于 30%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100；
对于 80%的数据，0≤表达式中加法运算符和乘法运算符的总数≤1000；
对于 100%的数据，0≤表达式中加法运算符和乘法运算符的总数≤100000。
小朋友的数字（noip2013 普及组第三题）(number.cpp/c/pas)
描述
有 n 个小朋友排成一列。每个小朋友手上都有一个数字,这个数字可正可负。规定每个小朋友的特征值等于排在他前面(包括他本人)的小朋友中连续若干个(最少有一个)小朋友手上的数字之和的最大值。 作为这些小朋友的老师,你需要给每个小朋友一个分数,分数是这样规定的:第一个小朋友的分数是他的特征值,其它小朋友的分数为排在他前面的所有小朋友中(不包括他本人),小朋友分数加上其特征值的最大值。
请计算所有小朋友分数的最大值,输出时保持最大值的符号,将其绝对值对 p 取

模后输出。
格式
【输入】
输入文件为 number.in。
第一行包含两个正整数 n、p，之间用一个空格隔开。
第二行包含 n 个数，每两个整数之间用一个空格隔开，表示每个小朋友手上的数字。
【输出】
输出文件名为 number.out。
输出只有一行，包含一个整数，表示最大分数对 p 取模的结果。
【输入输出样例 1】
number.in number.out
5 997 21
1 2 3 4 5
【输入输出样例说明】
小朋友的特征值分别为 1、3、6、10、15，分数分别为 1、2、5、11、21，最大值 21
对 997 的模是 21。
【输入输出样例 2】

number.in number.out
5 7 -1
-1 -1 -1 -1 -1
【输入输出样例说明】
小朋友的特征值分别为-1、-1、-1、-1、-1，分数分别为-1、-2、-2、-2、-2，最大值 -1 对 7 的模为-1，输出-1。
【数据范围】
对于 50%的数据，1 ≤ n ≤ 1,000，1 ≤ p ≤ 1,000 所有数字的绝对值不超过 1000；
99 对于 100%的数据，1 ≤ n ≤ 1,000,000， 1≤ p ≤ 10， 其他数字的绝对值均不超过 10。
车站分级（NOIP 普及组 2013 第四题）(level.cpp/c/pas)
描述
一条单向的铁路线上,依次有编号为 1, 2, ..., n 的 n 个火车站。每个火车站都有一个级别,最低为 1 级。现有若干趟车次在这条线路上行驶,每一趟都满足如下要求:如果这趟车次停靠了火车站 x,则始发站、终点站之间所有级别大于等于火车站 x 的都必须停靠。
(注意:起始站和终点站自然也算作事先已知需要停靠的站点)
例如,下表是 5 趟车次的运行情况。其中,前 4 趟车次均满足要求,而第 5 趟车次由于停靠了 3 号火车站(2 级)却未停靠途经的 6 号火车站(亦为 2 级)而不满足要求。
现有 m 趟车次的运行情况（全部满足要求），试推算这 n 个火车站至少分为几个不同的 级别。
【输入】

输入文件为 level.in。
第一行包含 2 个正整数 n, m，用一个空格隔开。
第 i + 1 行（1 ≤ i ≤ m）中，首先是一个正整数 si（2 ≤ si ≤ n），表示第 i 趟车次有 si 个停
靠站；接下来有 si 个正整数，表示所有停靠站的编号，从小到大排列。每两个数之间用一个 空格隔开。输入保证所有的车次都满足要求。
【输出】
输出文件为 level.out。
输出只有一行，包含一个正整数，即 n 个火车站最少划分的级别数。
第 3/4 页
【输入输出样例】
【数据范围】
对于 20%的数据，1 ≤ n, m ≤ 10； 对于 50%的数据，1 ≤ n, m ≤ 100； 对于 100%的数据，1 ≤ n, m ≤ 1000。
第 1 题，太弱了，不用骗，得 100 分。
第 2 题，数据很大，但是可以直接输入一个数，输出它 mod 10000 的值。得 10 分。 第 3 题，是一道非常基础的 DP，但对于不知 DP 为何物的蒟蒻来说，就使用暴力算法（即 DFS）。得 20 分。 第 4 题，我们可以寻找一下数据的规律，你会发现，在所有样例中，M 值即为答案。所以直接输出 M，得 10 分。 这样下来，一共得 140 分，比一等分数线还高 20 分！你的信心一定会得到鼓舞的。这就是骗分的神奇。 第 9 章 结语
骗分是蒟蒻的有力武器，可以在比赛中骗得大量分数。相信大家在这本书中收获了很多，希望本书能帮助你多得一些分。
但是，最后我还是要说一句：
不骗分，是骗分的最高境界。
作者：大神大麦茶 QQ：5416DMT