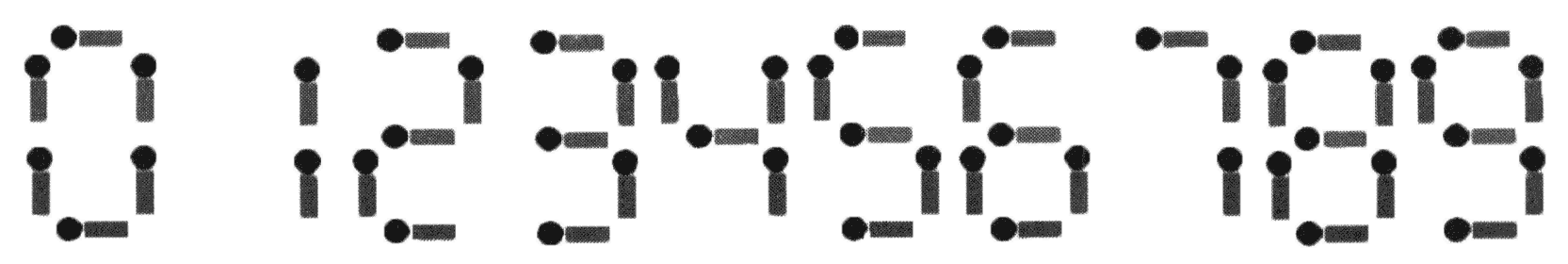
**NOIP2008火柴棒等式(matches.pas/c/cpp)**

【问题描述】

给你n根火柴棍，你可以拼出多少个形如“A+B=C”的等式？等式中的A、B、C是用火柴棍拼出的整数（若该数非零，则最高位不能是0）。用火柴棍拼数字0-9的拼法如图所示：



注意：

1）加号与等号各自需要两根火柴棍。

2）如果A≠B，则A+B=C与B+A=C视为不同的等式（A、B、C≥0）。

3）n根火柴棍必须全部用上。

【输入】

输入文件matches.in共一行，又一个整数n（n≤24）。

【输出】

输出文件matches.out共一行，表示能拼成的不同等式的数目。

【输入输出样例1】

|  |  |
| --- | --- |
| matches.in | matches.out |
| 14 | 2 |

【输入输出样例1解释】

2个等式为0+1=1和1+0=1。

【输入输出样例2】

|  |  |
| --- | --- |
| matches.in | matches.out |
| 18 | 9 |

【输入输出样例2解释】9个等式为：

0+4=4 0+11=11 1+10=11 2+2=4 2+7=9 4+0=4 7+2=9 10+1=11 11+0=11

**NOIP2009 Hankson的趣味题（son.pas/c/cpp）**

【问题描述】

Hanks博士是BT（Bio-Tech，生物技术）领域的知名专家，他的儿子名叫Hankson。现在，刚刚放学回家的Hankson正在思考一个有趣的问题。

今天在课堂上，老师讲解了如何求两个正整数和的最大公约数和最小公倍数。现在Hankson认为自己已经熟练地掌握了这些知识，他开始思考一个“求公约数”和“求公倍数”之类问题的“逆问题”，这个问题是这样的：已知正整数，，，，设某未知正整数x满足：

1）x和的最大公约数是；

2）x和的最小公倍数是。Hankson的“逆问题”就是求出满足条件的正整数x。但稍加思索之后，他发现这样的x并不唯一，甚至可能不存在。因此他转而开始考虑如何求解满足条件的x的个数。请你帮助他编程求解这个问题。

【输入】

输入文件名为son.in。第一行为一个正整数n，表示有n组输入数据。接下来的n行每行一组输入数据，为四个正整数，，，，每两个整数之间用一个空格隔开。输入数据保证能被整除，能被整除。

【输出】

输出文件son.out共n行。每组输入数据的输出结果占一行，为一个整数。对于每组数据：若不存在这样的x，请输出0；

若存在这样的x，请输出满足条件的x的个数；

【输入输出样例】

|  |  |
| --- | --- |
| Son.in | Son.out |
| 2  41 1 96 288  95 1 37 1776 | 6  2 |

【说明】

第一组输入数据，X可以是9、18、36、72、144、288，共有6个。

第二组输入数据，X可以是48、1776，共有2个。

【数据范围】

对于50%的数据，保证有1≤，，，≤且n≤100。

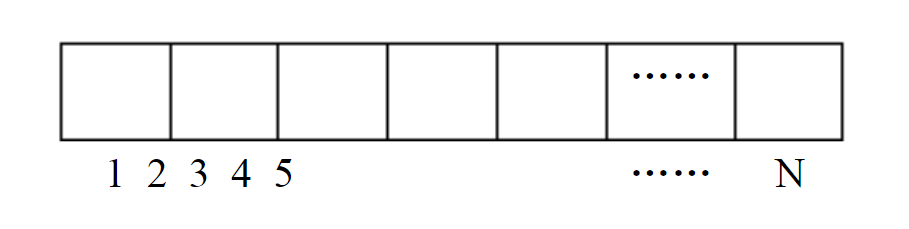
对于100%的数据，保证有1≤，，，≤2000000000且n≤2000。

**NOIP2010乌龟棋 (tortoise.pas/c/cpp)**

【问题描述】

小明过生日的时候，爸爸送给他一副乌龟棋当作礼物。

乌龟棋的棋盘是一行 N个格子，每个格子上一个分数（非负整数） 。棋盘第 1 格是唯一的起点，第 N格是终点，游戏要求玩家控制一个乌龟棋子从起点出发走到终点。



乌龟棋中 M 张爬行卡片，分成 4 种不同的类型（M 张卡片中不一定包含所有 4 种类型的卡片，见样例） ，每种类型的卡片上分别标有 1、2、3、4 四个数字之一，表示使用这种卡片后，乌龟棋子将向前爬行相应的格子数。游戏中，玩家每次需要从所有的爬行卡片中选择一张之前没有使用过的爬行卡片，控制乌龟棋子前进相应的格子数，每张卡片只能使用一次。

游戏中，乌龟棋子自动获得起点格子的分数，并且在后续的爬行中每到达一个格子，就得到该格子相应的分数。 玩家最终游戏得分就是乌龟棋子从起点到终点过程中到过的所有格子的分数总和。

很明显，用不同的爬行卡片使用顺序会使得最终游戏的得分不同，小明想要找到一种卡片使用顺序使得最终游戏得分最多。

现在，告诉你棋盘上每个格子的分数和所有的爬行卡片，你能告诉小明，他最多能得到多少分吗？

【输入】

输入文件名 tortoise.in。输入文件的每行中两个数之间用一个空格隔开。

第 1 行2 个正整数 N和 M，分别表示棋盘格子数和爬行卡片数。

第 2 行 N个非负整数，， ……, ，其中表示棋盘第i个格子上的分数。

第 3 行M 个整数，,, ……, ，表示 M 张爬行卡片上的数字。

输入数据保证到达终点时刚好用光 M 张爬行卡片，即。

【输出】

输出文件名 tortoise.out。

输出只有 1行，1个整数，表示小明最多能得到的分数。

【输入输出样例 1】

|  |  |
| --- | --- |
| tortoise.in | tortoise.out |
| 9 5  6 10 14 2 8 8 18 5 17  1 3 1 2 1 | 73 |

【输入输出样例 1 说明】

小明使用爬行卡片顺序为 1，1，3，1，2，得到的分数为 6+10+14+8+18+17=73。注意，由于起点是 1，所以自动获得第 1 格的分数 6。

【输入输出样例 2】

|  |  |
| --- | --- |
| tortoise.in | tortoise.out |
| 13 8  4 96 10 64 55 13 94 53 5 24 89 8 30  1 1 1 1 1 2 4 1 | 455 |

【数据范围】

对于 30%的数据有 1 ≤ N≤ 30，1 ≤M≤ 12。

对于 50%的数据有 1 ≤ N≤ 120，1 ≤M≤ 50，且 4 种爬行卡片，每种卡片的张数不会超过 20。

对于 100%的数据有 1 ≤ N≤ 350，1 ≤M≤ 120，且 4 种爬行卡片，每种卡片的张数不会超过 40；0 ≤ ≤ 100，1 ≤ i ≤ N；1 ≤ ≤ 4，1 ≤ i ≤M。输入数据保证 。

**NOIP2011选择客栈(hotel.pas/c/cpp)**

【问题描述】

丽江河边有 n 家很有特色的客栈，客栈按照其位置顺序从1 到n 编号。每家客栈都按照某一种色调进行装饰（总共k 种，用整数0 ~ k-1 表示），且每家客栈都设有一家咖啡店，每家咖啡店均有各自的最低消费。

两位游客一起去丽江旅游，他们喜欢相同的色调，又想尝试两个不同的客栈，因此决定分别住在色调相同的两家客栈中。晚上，他们打算选择一家咖啡店喝咖啡，要求咖啡店位于两人住的两家客栈之间（包括他们住的客栈），且咖啡店的最低消费不超过p。

他们想知道总共有多少种选择住宿的方案，保证晚上可以找到一家最低消费不超过p元的咖啡店小聚。

【输入】

输入文件 hotel.in，共n+1 行。

第一行三个整数 n，k，p，每两个整数之间用一个空格隔开，分别表示客栈的个数，色调的数目和能接受的最低消费的最高值；接下来的 n 行，第i+1 行两个整数，之间用一个空格隔开，分别表示i 号客栈的装饰色调和i 号客栈的咖啡店的最低消费。

【输出】

输出文件名为 hotel.out。

输出只有一行，一个整数，表示可选的住宿方案的总数。

【输入输出样例 1】

|  |  |
| --- | --- |
| hotel.in | hotel.out |
| 5 2 3  0 5  1 3  0 2  1 4  1 5 | 3 |

【输入输出样例说明】

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 客栈编号 | ① | ② | ③ | ④ | ⑤ |
| 色调 | 0 | 1 | 0 | 1 | 1 |
| 最低消费 | 5 | 3 | 2 | 4 | 5 |

2 人要住同样色调的客栈，所有可选的住宿方案包括：住客栈①③，②④，②⑤，④⑤，但是若选择住4、5 号客栈的话，4、5 号客栈之间的咖啡店的最低消费是4，而两人能承受的最低消费是3 元，所以不满足要求。因此只有前3 种方案可选。

【数据范围】

对于 30%的数据，有n≤100；

对于 50%的数据，有n≤1,000；

对于 100%的数据，有2≤n≤200,000，0<k≤50，0≤p≤100， 0≤最低消费≤100。

**NOIP2012国王游戏(game.cpp/c/pas)**

【问题描述】

恰逢 H 国国庆，国王邀请 n 位大臣来玩一个有奖游戏。首先，他让每个大臣在左、右手上面分别写下一个整数，国王自己也在左、 右手上各写一个整数。然后， 让这 n 位大臣排成一排，国王站在队伍的最前面。 排好队后，所有的大臣都会获得国王奖赏的若干金币， 每位大臣获得的金币数分别是：排在该大臣前面的所有人的左手上的数的乘积除以他自己右手上的数，然后向下取整得到的结果。

国王不希望某一个大臣获得特别多的奖赏，所以他想请你帮他重新安排一下队伍的顺序，使得获得奖赏最多的大臣，所获奖赏尽可能的少。注意，国王的位置始终在队伍的最前面。

【输入】

输入文件为 game.in。

第一行包含一个整数 n，表示大臣的人数。

第二行包含两个整数 a 和 b，之间用一个空格隔开，分别表示国王左手和右手上的整数。

接下来 n 行，每行包含两个整数 a 和 b，之间用一个空格隔开，分别表示每个大臣左手和右手上的整数。

【输出】

输出文件名为 game.out。

输出只有一行，包含一个整数，表示重新排列后的队伍中获奖赏最多的大臣所获得的金币数。

【输入输出样例】

|  |  |
| --- | --- |
| game.in | game.out |
| 3   1. 1 2. 3   7 4  4 6 | 2 |

【输入输出样例说明】

按 1、2、3 号大臣这样排列队伍，获得奖赏最多的大臣所获得金币数为 2；

按 1、3、2 这样排列队伍，获得奖赏最多的大臣所获得金币数为 2；

按 2、1、3 这样排列队伍，获得奖赏最多的大臣所获得金币数为 2；

按 2、3、1 这样排列队伍，获得奖赏最多的大臣所获得金币数为 9；

按 3、1、2 这样排列队伍，获得奖赏最多的大臣所获得金币数为 2；

按 3、2、1 这样排列队伍，获得奖赏最多的大臣所获得金币数为 9。

因此，奖赏最多的大臣最少获得 2 个金币，答案输出 2。

【数据范围】

对于 20%的数据，有 1≤ n≤ 10，0 < a、b < 8；

对于 40%的数据，有 1≤ n≤20，0 < a、b < 8；

对于 60%的数据，有 1≤ n≤100；

对于 60%的数据，保证答案不超过 ；

对于 100%的数据，有 1 ≤ n ≤1,000，0 < a、b < 10000。

**NOIP2013火柴排队(match.cpp/c/pas)**

【问题描述】

涵涵有两盒火柴，每盒装有 n 根火柴，每根火柴都有一个高度。 现在将每盒中的火柴各自排成一列， 同一列火柴的高度互不相同， 两列火柴之间的距离定义为： 其中 ai 表示第一列火柴中第 i 个火柴的高度，bi 表示第二列火柴中第 i 个火柴的高度。

每列火柴中相邻两根火柴的位置都可以交换，请你通过交换使得两列火柴之间的距离最小。请问得到这个最小的距离，最少需要交换多少次？如果这个数字太大，请输出这个最小交换次数对 99,999,997 取模的结果。

【输入】

输入文件为 match.in。

共三行，第一行包含一个整数 n，表示每盒中火柴的数目。

第二行有 n 个整数，每两个整数之间用一个空格隔开，表示第一列火柴的高度。

第三行有 n 个整数，每两个整数之间用一个空格隔开，表示第二列火柴的高度。

【输出】

输出文件为 match.out。

输出共一行，包含一个整数，表示最少交换次数对 99,999,997 取模的结果。

【输入输出样例 1】

|  |  |
| --- | --- |
| match.in | match.out |
| 4  2 3 1 4  3 2 1 4 | 1 |

【输入输出样例说明】

最小距离是 0，最少需要交换 1 次，比如：交换第 1 列的前 2 根火柴或者交换第 2 列的前 2 根火柴。

【输入输出样例 2】

|  |  |
| --- | --- |
| match.in | match.out |
| 4  1 3 4 2  1 7 2 4 | 2 |

【输入输出样例说明】

最小距离是 10，最少需要交换 2 次，比如：交换第 1 列的中间 2 根火柴的位置，再交换第 2 列中后 2 根火柴的位置。

【数据范围】

对于 10%的数据， 1 ≤ n ≤ 10；

对于 30%的数据，1 ≤ n ≤ 100；

对于 60%的数据，1 ≤ n ≤ 1,000；

对于 100%的数据，1 ≤ n ≤ 100,000，0 ≤火柴高度≤ -1。

**NOIP2014联合权值 (link.cpp/c/pas)**

【问题描述】

无向连通图 G 有 n 个点，n-1 条边。点从 1 到 n 依次编号，编号为 i 的点的权值为 Wi，每条边的长度均为 1。图上两点(u, v)的距离定义为 u 点到 v 点的最短距离。对于图 G 上的点对(u, v)，若它们的距离为 2，则它们之间会产生 Wu×Wv 的联合权值。

请问图 G 上所有可产生联合权值的有序点对中，联合权值最大的是多少？所有联合权值 之和是多少？

【输入】

输入文件名为 link.in。 第一行包含 1 个整数 n。

接下来 n-1 行，每行包含 2 个用空格隔开的正整数 u、v，表示编号为 u 和编号为 v 的 点之间有边相连。

最后 1 行，包含 n 个正整数，每两个正整数之间用一个空格隔开，其中第 i 个整数表示 图 G 上编号为 i 的点的权值为 Wi。

【输出】

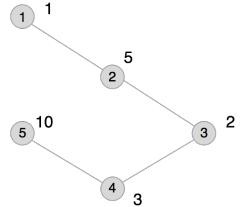
输出文件名为 link.out。

输出共 1 行，包含 2 个整数，之间用一个空格隔开，依次为图 G 上联合权值的最大值和所有联合权值之和。由于所有联合权值之和可能很大，输出它时要对 10007 取余。

【输入输出样例】

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| link.in | | | | | link.out |
| 5 | 5 |  |  | 5  1 2  2 3  3 4  4 5  1 5 2 3 10 | 20 74 |
| 1 | 2 |  |  | 1 2 |
| 2 | 3 |  |  | 2 3 |
| 3 | 4 |  |  | 3 4 |
| 4 | 5 |  |  | 4 5 |
| 1 | 5 |  |  | 1 5 2 3 10 |

【样例说明】



本例输入的图如上所示，距离为 2 的有序点对有(1,3)、(2,4)、(3,1)、(3,5)、(4,2)、 (5,3)。其联合权值分别为 2、15、2、20、15、20。其中最大的是 20，总和为 74。

【数据说明】

对于 30%的数据，1<≤100； 对于 60%的数据，1<≤2000；

对于 100%的数据，1<≤200,000，0<Wi ≤10,000。

**NOIP2015信息传递(message.cpp/c/pas)**

【问题描述】

有 𝑛 个同学（编号为 1 到 𝑛）正在玩一个信息传递的游戏。在游戏里每人都有一个固定的信息传递对象，其中，编号为 𝑖 的同学的信息传递对象是编号为𝑇𝑖的同学。

游戏开始时，每人都只知道自己的生日。之后每一轮中，所有人会同时将自己当前所知的生日信息告诉各自的信息传递对象（注意：可能有人可以从若干人那里获取信息， 但是每人只会把信息告诉一个人，即自己的信息传递对象）。当有人从别人口中得知自 己的生日时，游戏结束。请问该游戏一共可以进行几轮？

【输入格式】

输入文件名为 message.in。 输入共 2 行。

第 1 行包含 1 个正整数 𝑛，表示 𝑛 个人。

第 2 行包含 𝑛 个用空格隔开的正整数 𝑇1, 𝑇2, … … , 𝑇𝑛，其中第𝑖 个整数𝑇𝑖表示编号为 𝑖

的同学的信息传递对象是编号为 Ti 的同学， 𝑇𝑖 ≤ 𝑛 且 𝑇𝑖 ≠ 𝑖。

数据保证游戏一定会结束。

【输出格式】

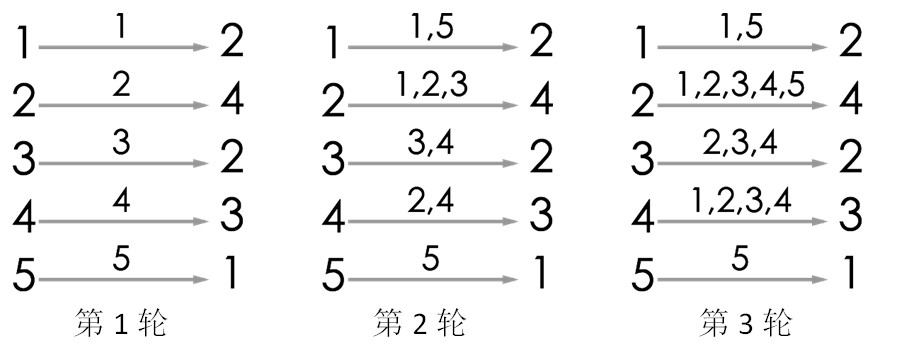
输出文件名为 message.out。

输出共 1 行，包含 1 个整数，表示游戏一共可以进行多少轮。

【输入输出样例】

|  |  |
| --- | --- |
| message.in | message.out |
| 5  2 4 2 3 1 | 3 |

【输入输出样例说明】



游戏的流程如图所示。当进行完第 3 轮游戏后，4 号玩家会听到 2 号玩家告诉他自 己的生日，所以答案为 3。当然，第 3 轮游戏后，2 号玩家、3 号玩家都能从自己的消息 来源得知自己的生日，同样符合游戏结束的条件。

【数据规模与约定】

对于 30%的数据，𝑛 ≤ 200；

对于 60%的数据，𝑛 ≤ 2500；

对于 100%的数据，𝑛 ≤ 200000。

**NOIP2016天天爱跑步（running.cpp/c/pas)**

【问题描述】

小 C 同学认为跑步非常有趣，于是决定制作一款叫做《天天爱跑步》的游戏。

《天天爱跑步》是一个养成类游戏，需要玩家每天按时上线，完成打卡任务。这个游戏的地图可以看作一棵包含n个结点和 n−1条边的树，每条边连接两个结点，且任意两个结点存在一条路径互相可达。树上结点编号为从1到n的连续正整数。

现在有 m 个玩家，第i个玩家的起点为，终点为。每天打卡任务开始时，所有玩家在第 0 秒同时从自己的起点出发，以每秒跑一条边的速度，不间断地沿着最短路径向着自己的终点跑去，跑到终点后该玩家就算完成了打卡任务。（由于地图是一棵树，所以每个人的路径是唯一的），小 C 想知道游戏的活跃度，所以在每个结点上都放置了一个观察员。在结点j的观察员会选择在第秒观察玩家，一个玩家能被这个观察员观察到当且仅当该玩家 在第 秒也正好到达了结点j。 小 C 想知道每个观察员会观察到多少人？

注意：我们认为一个玩家到达自己的终点后该玩家就会结束游戏，他不能等待一段时间后再被观察员观察到。即对于把结点 j 作为终点的玩家：若他在第 秒前到达终点，则在结点 j 的观察员不能观察到该玩家；若他正好在第秒到达终点，则在结点j的观察员可以观察到这个玩家。

【输入格式】

从文件 running.in 中读入数据。

第一行有两个整数 n 和 m 。其中 n 代表树的结点数量，同时也是观察员的数量，m 代表玩家的数量。

接下来 n−1行每行两个整数u和v，表示结点u到结点v有一条边。接下来一行n个整数，其中第j个整数为，表示结点j出现观察员的时间。接下来m行，每行两个整数和，表示一个玩家的起点和终点。对于所有的数据，保证 1≤ ,≤n ，0≤≤n。

【输出格式】

输出到文件 running.out 中。

输出 1 行 n 个整数，第 j 个整数表示结点 j 的观察员可以观察到多少人。

【样例 1 输入】

|  |  |
| --- | --- |
| running.in | running.out |
| 6 3  2 3  1 2  1 4  4 5  4 6  0 2 5 1 2 3  1 5  1 3  2 6 | 2 0 0 1 1 1 |

对于1号点， = 0 ，故只有起点为1号点的玩家才会被观察到，所以玩家1和玩家2被观察到，共2人被观察到。

对于2号点，没有玩家在第2秒时在此结点，共0人被观察到。对于3号点没有玩家在第5秒时在此结点，共0人被观察到。对于4号点，玩家1被观察到，共1人被观察到。

对于5号点，玩家2被观察到，共1人被观察到。对于6号点，玩家3被观察到，共1人被观察到。

|  |  |
| --- | --- |
| running.in | running.out |
| 5 3  1 2  2 3  2 4  1 5  0 1 0 3 0  3 1  1 4  5 5 | 1 2 1 0 1 |

【子任务】

每个测试点的数据规模及特点如下表所示。提示：数据范围的个位上的数字可以帮助判断 是哪一种数据类型。

|  |  |  |  |
| --- | --- | --- | --- |
| 测试点编号 | n | m | 约定 |
| 1 | = 991 | = 991 | 所有人的起点等于自己的终点，  即 = |
| 2 |
| 3 | = 992 | = 992 | = 0 |
| 4 |
| 5 | = 993 | = 993 | 无 |
| 6 | = 99994 | = 99994 | 树退化成一条链，其中 1 与 2 有边，  2 与 3 有边，. . . ，n − 1 与 n 有边 |
| 7 |
| 8 |
| 9 | = 99995 | = 99995 | 所有的 = 1 |
| 10 |
| 11 |
| 12 |
| 13 | = 99996 | = 99996 | 所有的 = 1 |
| 14 |
| 15 |
| 16 |
| 17 | = 99997 | = 99997 | 无 |
| 18 |
| 19 |
| 20 | = 299998 | = 299998 |

【提示】

如果你的程序需要用到较大的栈空间（这通常意味着需要较深层数的递归），请务必仔细阅读选手目录下的文档 running/stack.pdf，以了解在最终评测时栈空间的限制与在当前工作环境下调整栈空间限制的方法。

**NOIP2017时间复杂度(complexity.cpp/c/pas)**

【问题描述】

小明正在学习一种新的编程语言 A++，刚学会循环语句的他激动地写了好多程序并 给出了他自己算出的时间复杂度，可他的编程老师实在不想一个一个检查小明的程序，于是你的机会来啦！下面请你编写程序来判断小明对他的每个程序给出的时间复杂度是否正确。

A++语言的循环结构如下：

F i x y

循环体

E

其中“**F i x y**”表示新建变量 （i 变 量 i 不可与未被销毁的变量重名）并初始化为 x， 然后判断 i 和 y 的大小关系，若 i 小于等于 y 则进入循环，否则不进入。每次循环结束 后 i 都会被修改成 i +1，一旦 i 大于 y 终止循环。

x 和 y 可以是正整数（x 和 y 的大小关系不定）或变量 n。n 是一个表示数据规模的 变量，在时间复杂度计算中需保留该变量而不能将其视为常数，该数远大于100。

“**E**”表示循环体结束。循环体结束时，这个循环体新建的变量也被销毁。

注：本题中为了书写方便，在描述复杂度时，使用大写英文字母“O”表示通常意义下“Θ”的概念。

【输入格式】

输入文件名为 complexity.in。

输入文件第一行一个正整数 t，表示有 t（t ≤ 10）个程序需要计算时间复杂度。 每个程序我们只需抽取其中 “**F i x y**”和“**E**”即可计算时间复杂度。注意：循环结构允许嵌套。

接下来每个程序的第一行包含一个正整数 L 和一个字符串，L 代表程序行数，字符串表示这个程序的复杂度，“O(1)”表示常数复杂度，“O(n^w)”表示复杂度为，其 中 w是一个小于 100 的正整数（输入中不包含引号），输入保证复杂度只有 O(1)和 O(n^w)两种类型。

接下来 L 行代表程序中循环结构中的“F i x y”或者 “E”。 程序行若以“F”开头，表示进入一个循环，之后有空格分离的三个字符（串）i x y，其中 i 是一个小写字母（保证不为“n”），表示新建的变量名，x 和 y 可能是正整数或n ，已知若为正整数则一定小于 100。 程序行若以“**E**”开头，则表示循环体结束。

【输出格式】

输出文件名为 complexity.out。

输出文件共 t 行，对应输入的 t 个程序，每行输出“Yes”或“No”或者“ERR”（输 出中不包含引号），若程序实际复杂度与输入给出的复杂度一致则输出“Yes”，不一致 则输出“No”，若程序有语法错误（其中语法错误只有: ① F 和 E 不匹配 ②新建的变 量与已经存在但未被销毁的变量重复两种情况），则输出“ERR”。

注意：即使在程序不会执行的循环体中出现了语法错误也会编译错误， 要输出 “ERR”。

【输入输出样例 1】

|  |  |
| --- | --- |
| complexity.in | complexity.out |
| 8  2 O(1)  F i 1 1  E  2 O(n^1)  F x 1 n  E  1 O(1)  F x 1 n  4 O(n^2)  F x 5 n  F y 10 n  E  E  4 O(n^2)  F x 9 n  E  F y 2 n  E  4 O(n^1)  F x 9 n  F y n 4  E  E  4 O(1)  F y n 4  F x 9 n  E  E  4 O(n^2)  F x 1 n | Yes Yes ERR  Yes No Yes Yes ERR |
| F x 1 10  E  E |  |

【输入输出样例 1 说明】

第一个程序 i 从 1 到 1 是常数复杂度。

第二个程序 x 从 1 到 n 是 n 的一次方的复杂度。

第三个程序有一个 F 开启循环却没有 E 结束，语法错误。

第四个程序二重循环，n 的平方的复杂度。

第五个程序两个一重循环，n 的一次方的复杂度。

第六个程序第一重循环正常，但第二重循环开始即终止（因为 n 远大于 100，100 大于 4）。

第七个程序第一重循环无法进入，故为常数复杂度。

第八个程序第二重循环中的变量 x 与第一重循环中的变量重复，出现语法错误②，输出ERR。

【数据规模与约定】

对于 30%的数据：不存在语法错误，数据保证小明给出的每个程序的前 L/2 行一定 为以 F 开头的语句，第 L/2+1 行至第 L 行一定为以 E 开头的语句，L≤10，若 x、y 均 为整数，x 一定小于 y，且只有 y 有可能为 n。

对于 50%的数据：不存在语法错误，L≤100，且若 x、y 均为整数，x 一定小于 y， 且只有 y 有可能为 n。

对于 70%的数据：不存在语法错误，L≤100。

对于 100%的数据：L≤100。

**NOIP2018货币系统(money.cpp/c/pas)**

【问题描述】

在网友的国度中共有 n 种不同面额的货币，第i种货币的面额为 a[i]，你可以假设每一种货币都有无穷多张。为了方便，我们把货币种数为 n、面额数组为a[1..n]的货币系统记作 (n,a)。

在一个完善的货币系统中，每一个非负整数的金额 x 都应该可以被表示出，即对每一个非负整数 x，都存在 n 个非负整数 t[i] 满足 a[i]× t[i] 的和为 x。然而，在网友的国度中，货币系统可能是不完善的，即可能存在金额 x 不能被该货币系统表示出。例如在货币系统 n=3, a=[2,5,9] 中，金额 1,3 就无法被表示出来。

两个货币系统 (n,a) 和 (m,b) 是等价的，当且仅当对于任意非负整数 x它要么均可以被两个货币系统表出，要么不能被其中任何一个表出。

现在网友们打算简化一下货币系统。他们希望找到一个货币系统 (m,b)，满足 (m,b) 与原来的货币系统(n,a)等价，且 m 尽可能的小。他们希望你来协助完成这个艰巨的任务：找到最小的 m。

【输入格式】

输入文件名为 money.in。

输入文件的第一行包含一个整数 T，表示数据的组数。接下来按照如下格式分别给 出 T 组数据。

每组数据的第一行包含一个正整数 n。接下来一行包含 n 个由空格隔开的正整数

a[i]。

【输出格式】

输出文件名为 money.out。

输出文件共有 T 行，对于每组数据，输出一行一个正整数，表示所有与 (n,a) 等 价的货币系统 (m,b) 中，最小的 m。

【输入输出样例】

|  |  |
| --- | --- |
| money.in | money.out |
| 2  4  3 19 10 6  5  11 29 13 19 17 | 2  5 |

【输入输出样例说明】

在第一组数据中，货币系统 (2, [3,10]) 和给出的货币系统 (n, a) 等价，并可以验证不存在 m < 2 的等价的货币系统，因此答案为 2。

在第二组数据中，可以验证不存在 m < n 的等价的货币系统，因此答案为 5。

【数据规模与约定】

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试点 | n |  | 测试点 | n |  |
| 1 | = 2 | ≤ 1000 | 11 | ≤ 13 | ≤ 16 |
| 2 | 12 |
| 3 | 13 |
| 4 | = 3 | 14 | ≤ 25 | ≤ 40 |
| 5 | 15 |
| 6 | 16 |
| 7 | = 4 | 17 | ≤ 100 | ≤ 25000 |
| 8 | 18 |
| 9 | = 5 | 19 |
| 10 | 20 |

对于 100% 的数据，满足 1 ≤ T ≤ 20, n,a[i] ≥ 1。