全国青少年信息学奥林匹克
中国国家队训练 2010-2011

杨天

竞赛时间：待定

题目名称	等差子序列	最短路	排斥反应
目录	sequence	path	react
可执行文件名	sequence	path	react
输入文件名	标准输入	标准输入	标准输入
输出文件名	标准输出	标准输出	标准输出
每个测试点时限	0.3 秒	0.1 秒	2 秒
测试点数目	20	20	20
每个测试点分值	5	5	5
是否有部分分	否	否	否
题目类型	传统型	传统型	传统型

提交源程序须加后缀

对于 Pascal 语言	sequence.pas	path.pas	react.pas
对于 C 语言	sequence.c	path.c	react.c
对于 C++ 语言	sequence.cpp	path.cpp	react.cpp

注意：最终测试时，所有编译命令均不打开任何优化开关。

等差子序列

【问题描述】

给一个 1 到 N 的排列{Ai}，询问是否存在 1<=p1<p2<p3<p4<p5<…<pLen<=N（Len>=3），使得 $A_{p1},A_{p2},A_{p3},\ldots A_{pLen}$ 是一个等差序列。

【输入格式】

输入的第一行包含一个整数 T，表示组数。

下接 T 组数据，每组第一行一个整数 N，每组第二行为一个 1 到 N 的排列，数字两两之间用空格隔开。

【输出格式】

对于每组数据，如果存在一个等差子序列，则输出一行“Y”，否则输出一行“N”。

【样例输入】

```
2
3
1 3 2
3
3 2 1
```

【样例输出】

```
N
Y
```

【数据说明】

对于 5%的数据，N<=100

对于 30%的数据，N<=1000

对于 100%的数据，N<=10000，T<=7

【解题报告】

首先我们看到此题，容易发现如果存在 Len>3 的等差子序列，那么一定存在 Len=3 的等差子序列，所以问题可以转化为判断序列中是否存在 p<q<r，使得 Ap-Aq=Aq-Ar。

首先很容易地可以想到一个 $O(N^3)$的算法，枚举 p，q，r，然后强行判断，这样可以得到 5%的分数。

然后我们可以发现，如果我们先枚举一个 q，然后由于 Ap-Aq=Aq-Ar，我们

可以利用差相等的性质来优化，枚举完之后，我们在<q 的所有 Ap-Aq 的差与>q 的所有 Aq-Ar 用 Hash 表记录下来，然后看是否重复，这样是一个 $O(N^2)$的算法，可以拿到 30%的分数。

要想通过所有的数据，我们必须进一步优化，得到更优的算法。

首先我们研究上面两个算法，发现他们都没有好好利用一些本题独有的性质，那么什么是本题独有的性质呢？排列！

排列有什么性质呢？

1 到 N 的数每个数都会出现，那么就是说，如果对于一个 1 到 N 的排列，如果前 K 个数里面没有 X（1<=X<=N），那么后面的 N-K 个数里面一定会出现 X。

但是光只是这么发掘这个性质对于解决本题是远远不够的，我们仍然需要对本题的一些独有性质作进一步挖掘。

前面那个等式是通过对数的位置的描述得出的，而我们发现可以通过从数本身的角度来描述上面那个等式。

定义 B_i 表示 i 这个数在原序列中出现的位置，那么对于一个数 Y，需要判断的就变成了对于所有满足 2Y=X+Z 的 X，Z，判断 X，Z 的出现的位置是否满足要求。

对于这个问题，我们要把它分解成两个小的问题，第一个就是找出所有满足要求的 X，Z，第二个就是位置的出现有什么要求。

我们先来看第一个问题，因为 2Y=X+Z，那么如果我们把 1 到 N 的数全部写在一个数轴上。

1 2 3 4 5 6 7 ……

假设 Y=4，那么容易发现 X=3，Z=5；X=2，Z=6；X=1，Z=7 都是满足条件的对子

即满足条件的对子关于 Y 在数轴上的点轴对称。

现在我们再来看看第二个问题，位置有什么要求。

假设 X，Z 是关于 Y 的合法对子，那么 B_x，B_z 一定一个大于 B_y，一个小于 B_y。

现在我们尝试从另一个角度来描述这个问题。

从这个排列的第一个位置开始往后扫描，设每次扫到的数字为 Aq，那么就判断所有满足条件的对子中是否存在一个在 1 到 q 中出现，一个在 1 到 q 中没有出现。

我们发现这个算法由于要判断这些对子，而这些对子的总数也是高达 $O(N^2)$的。

那么我们有什么方式可以快速判断多个对子呢？

前面的分析中用到的数轴，我们继续来用数轴来表示问题。一开始把数轴上 1 到 N 的位置都标记为 0，表示暂时没有出现，然后每次向后扫描进入一个位置 q，就将 Aq 上标记为 1，并且同时判断对子是否存在一个出现，一个没有出现的。

举一个例子吧，比如{5，1，3，7，2，6，4}我们现在扫描到 3 了，那么我们来数轴上的情况。

1 2 3 4 5 6 7

1 0 1 0 1 0 0

我们要判断的对子就是（1，5），（2，4），也就是如果将这个东西看成 01 串，判断 S(1->2)，与 S(5->4)是否相同。

判断连续的 01 字符串是否相等可以用 Hash 表来做，关于如何构造 Hash 值，参考杨弋同学 07 年的论文。

但是求 Hash 仍然是个 $O(N^2)$，然后我们可以构造一个线段树来快速维护 Hash，这样我们可以通过一次扫描来判断，时间复杂度 O(NlogN)

最短路

【问题描述】

给一个 N 个点 M 条边的连通无向图，满足每条边最多属于一个环，有 Q 组询问，每次询问两点之间的最短路径。

【输入格式】

输入的第一行包含三个整数，分别表示 N 和 M 和 Q
下接 M 行，每行三个整数 v，u，w 表示一条无向边 v-u，长度为 w
最后 Q 行，每行两个整数 v，u 表示一组询问

【输出格式】

输出 Q 行，每行一个整数表示询问的答案

【样例输入】

```
9 10 2
1 2 1
1 4 1
3 4 1
2 3 1
3 7 1
7 8 2
7 9 2
1 5 3
1 6 4
5 6 1
1 9
5 7
```

【样例输出】

```
5
6
```

【数据说明】

对于 5%的数据，N<=100
对于 20%的数据，N<=1000
对于 100%的数据，N<=10000，Q<=10000

【解题报告】

这个题目是关于最短路的题，对于一般最短路的题目，一般来说，dijkstra 算法和 floyd 算法是我们的首选，但是这个题目如果用这两个算法来做，绝对会超时，所以我们需要挖掘题目的性质找到题目的更优算法。

首先由于是求两两之间的最短路，显然 floyd 算法是首选，但是这个算法的时间复杂度会达到 $O(N^3)$，能通过 5%的数据。

关于这个题目的性质，我们首先要来推测一下边的总数，我们可以用下面的方法来推测边的总数。

首先用 dfs 求出一棵 dfs 树，然后把非树边两端点经过的边做标记，发现每条边最多被标记一次，显然边数 M<=2*N。

有了这个性质，我们就可以套用 dijkstra 算法对每一个点求一次单源最短路时间复杂度 O(NMlogN)，可以通过 20%的数据。

前面我们通过对题目边数的分析找到了一定的性质，现在让我们尝试找出更多特别的性质。

我们可以尝试着拿几个图出来进行分析(比如样例)，首先我们可以对这个图求块，把块求出来之后我们看看容易发现，块分为两种，一种就是简单的一条边，另一种就是简单的一个环。

比如样例其中的块为{1-2，2-3，3-4，1-4}，{1-5，5-6，6-1}，{3-7}，{7-8}，{7-9}，貌似我们又更进了一步。

接下来我们来看看另外一个问题，假设我们原题不是图而是一棵树，我们怎么来完成这样的询问？我们可以任选一个点，然后建树，然后由于路径是唯一的所以比较简单的可以完成，利用求 lca 的方法。而为什么一般图不能套用树的方法呢？第一，一般图有多条路径；第二，一般图无法分层。

而回到原题，发现本题的图是可以分层的。我们把 1 号点定为根，然后可以发现，一号点就可以拓展出若干个块，这些块会拓展出一些点，把 1 号点叫做这些块的拓展点，把拓展出一个点的块叫做这个点的原始块，然后这些被拓展的点又可以拓展出一些新的拓展块，重复下去。

在这样的过程中，我们可以发现以下一些性质。

1.一个点只会被拓展一次：因为如果一个点 X 被拓展了 2 次，那么说明它被两个不同的块拓展了，设这两个块为 G1，G2，那么由于 1 到 G1 到 X 有路径，且 1 到 G2 到 X 有路径，所以 G1，G2，1，X 应该属于一个块，所以矛盾。

2.一个点只有一个原始块，由性质 1 易证。

同理一个块也只有一个原始点，又根据性质 2，一个点只有一个原始块，那么我们可以把 1 个点的原始块的原始点叫做这个点的父亲节点，把这个点到父亲节点的最短路计为这个点到其父亲节点的边的权，用这样的一些边来构造一个新的图。显然每个点的父亲节点只有一个且 1 没有父亲节点，显然 N 点 N-1 条边，这是一棵树。

找到这样一个性质，我们就可以放心大胆的使用这个原来求树上最短路的算法来完成了，特别要注意的是，最后求两点最短路的时候，lca 所在的那一层的权值要特殊处理，因为如果是这两点在 lca 相遇时候是在同一个圈上的时候，权值就为 min{TotLenOfCircle-abs(A-B),abs(A-B)}，A，B 分别是两个点在 lca 那一层的边权。时间复杂度为 O(QlogN+NlogN)

排斥反应

【问题描述】

在一个圆上均匀分布 p*q 个点{A_1，A_2，A_3…A_{p*q}}，Ai 与 Aj 的距离为 min{abs(i-j)，p*q-abs(i-j)}，在上面选任意个点(可以选 0 个)，如果选择的点中存在两个点距离为 p 或 q，就会发生排斥反应，求不发生排斥反应的方案总数。

【输入格式】

输入的第一行包含两个整数，分别表示 p 和 q

【输出格式】

输出一个整数，表示方案总数，由于这个题答案可能很大，只要输出答案 mod 19921107

【样例输入】

```
1 6
```

【样例输出】

```
18
```

【数据说明】

对于 5%的数据，p=1，q<=10^6
对于 15%的数据，p=1，q<=10^9
对于 100%的数据，p<=10，q<=10^9，p 和 q 互质

【解题报告】

这是一个有关于组合类型的试题，关于这个题目我们先从最简单的部分开始下手。

对于 5%的数据 p=1，容易发现在这种情况下不可能存在两个被选的点距离为 q，所以我们可以直接用动态规划来完成。首先我们枚举 1 这个点是否被选，然后用 f[i]表示选 i 这个点的长度为 i 的序列(注意，这里不是环)总方案数，g[i]表示不选择 i 这个点的长度为 i 的序列的总方案数。容易得到转移：

F[i]=g[i-1]
G[i]=f[i-1]+g[i-1]
时间复杂度 O(q)

观察上面的转移方程，我们很容易发现每次的转移之和之前的两个数 f[i-1]和 g[i-1]有关，我们想到了矩阵乘法，用矩阵乘法加速动态规划的转移，时间复杂度为 O(2^3logq)，可以通过 15%的数据。

而对于 p 不等于 1 的情况我们就没有什么好的办法了，我们不能直接记录前若干个点来推算下面的状态，需要另辟蹊径。

注意，因为我们每次算的时候都是从 1 开始按顺序计算，这样是无法同时保证 p，q 的不出现的。

如果我们改变一下计算的顺序呢？

现在我们来看一个例子 p=5，q=6，我们列出这样的一张表格，这个表格第一行第一列为 1，然后每向右走一格就+p(带 mod 的)，向下走一格就+q 于是我们可以得到下面这样一张神奇的表格。

1	6	11	16	21	26
7	12	17	22	27	2
13	18	23	28	3	8
19	24	29	4	9	14
25	30	5	10	15	20

容易发现，1-p*q 的每一个数都出现了一遍，为什么会这样呢？

我们试着找寻题目中的条件来证明这个结论，发现 p 和 q 互质，现在我们要证明我们对于每一对互质的 p 和 q 都能够写成这样一张表格。

首先对于我们按照 mod p 的取值讲这 p*q 个点分成 p 个等价类，由于向右走是不改变 mod p 的值的，我们只要证明对于当 x 取 0 到 p-1 时，x*q(mod p)的取值各不相同，这是显然的，如果存在两个 x1，x2 使得 x1*q-x2*q=p*y，由于 gcd(p，q)=1 和不定方程的相关性质，显然 x1-x2=z*p，显然对于 x 取值为 0 到 p-1 时是不会重复的。

同理我们很容易可以证明对于当 x 取 0 到 q-1 时，x*p(mod q)的取值各不相同，于是我们可以得到这个表格中每一行都是 mod p 相同的一类，每一列都是 mod q 相同的一类，1 到 p*q 中的每一个数 x 都可以通过找出 x mod p 所属的这一行，x mod q 的这一列来唯一确定。

通过这样几步变换我们构造了这样一个 p*q 的矩阵，这时候我们惊奇的发现这个题目可以这样来描述，给一个 p*q 的矩阵，从中选出若干个元素，使得元素互不相邻。(注意，这里的矩阵是上下相连，左右相接的)

题目中提到 p<=10，这样容易让我们想到状态压缩动态规划问题。我们枚举第一行的 01 状态表示，然后用矩阵来加速，时间复杂度 $O((2^p)^3 \log q)$，无法通过数据。

注意到由于当 p=10 的时候 1024 种状态中，有大量的状态被荒废，我们通过一轮枚举来减少可行状态，我们发现当 p=10 时，真正有用的状态为 123 种，最后的时间复杂度为 $O(TotState^3 \log q)$