CCF 全国信息学奥林匹克联赛（NOIP2012）复赛

提高组 day1

（请选手务必仔细阅读本页内容）

一．题目概况

中文题目名称	Vigenère 密码	国王游戏	开车旅行
英文题目与子目录名	vigenere	game	drive
可执行文件名	vigenere	game	drive
输入文件名	vigenere.in	game.in	drive.in
输出文件名	vigenere.out	game.out	drive.out
每个测试点时限	1 秒	1 秒	1 秒
测试点数目	10	10	20
每个测试点分值	10	10	5
附加样例文件	有	有	有
结果比较方式	全文比较（过滤行末空格及文末回车）		
题目类型	传统	传统	传统

二．提交源程序文件名

对于 C++语言	vigenere.cpp	game.cpp	drive.cpp
对于 C 语言	vigenere.c	game.c	drive.c
对于 pascal 语言	vigenere.pas	game.pas	drive.pas

三．编译命令（不包含任何优化开关）

对于 C++语言	g++ -o vigenere vigenere.cpp -lm	g++ -o game game.cpp -lm	g++ -o drive drive.cpp -lm
对于 C 语言	gcc -o vigenere vigenere.c -lm	gcc -o game game.c -lm	gcc -o drive drive.c -lm
对于 pascal 语言	fpc vigenere.pas	fpc game.pas	fpc drive.pas

四．运行内存限制

内存上限	128M	128M	128M

注意事项：

1、文件名（程序名和输入输出文件名）必须使用英文小写。

2、C/C++中函数 main()的返回值类型必须是 int，程序正常结束时的返回值必须是 0。

3、全国统一评测时采用的机器配置为：CPU Intel Core2 Quad Q8200 2.33GHz，内存 2G，上述时限以此配置为准。

4、特别提醒：评测在 NOI Linux 下进行。

1. Vigenère 密码

(vigenere.cpp/c/pas)

【问题描述】

16 世纪法国外交家 Blaise de Vigenère 设计了一种多表密码加密算法——Vigenère 密码。Vigenère 密码的加密解密算法简单易用，且破译难度比较高，曾在美国南北战争中为南军所广泛使用。

在密码学中，我们称需要加密的信息为明文，用 M 表示；称加密后的信息为密文，用 C 表示；而密钥是一种参数，是将明文转换为密文或将密文转换为明文的算法中输入的数据，记为 k。 在 Vigenère 密码中，密钥 k 是一个字母串，$k=k_1k_2\cdots k_n$。当明文 $M=m_1m_2\cdots m_n$ 时，得到的密文 $C=c_1c_2\cdots c_n$，其中 $c_i=m_i®k_i$，运算®的规则如下表所示：

®	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Vigenère 加密在操作时需要注意：

1. ®运算忽略参与运算的字母的大小写，并保持字母在明文 M 中的大小写形式；
2. 当明文 M 的长度大于密钥 k 的长度时，将密钥 k 重复使用。

例如，明文 M=Helloworld，密钥 k=abc 时，密文 C=Hfnlpyosnd。

明文	H	e	l	l	o	w	o	r	l	d
密钥	a	b	c	a	b	c	a	b	c	a
密文	H	f	n	l	p	y	o	s	n	d

【输入】

输入文件名为 vigenere.in。

输入共 2 行。

第一行为一个字符串，表示密钥 k，长度不超过 100，其中仅包含大小写字母。第二行为一个字符串，表示经加密后的密文，长度不超过 1000，其中仅包含大小写字母。

【输出】

输出文件名为 vigenere.out。

输出共 1 行，一个字符串，表示输入密钥和密文所对应的明文。

【输入输出样例】

vigenere.in	vigenere.out
CompleteVictory Yvqgpxaimmklongnzfwpvxmniytm	Wherethereisawillthereisaway

【数据说明】

对于 100%的数据，输入的密钥的长度不超过 100，输入的密文的长度不超过 1000，且都仅包含英文字母。

2．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 $1 \le n \le 10$，$0 < a、b < 8$；
对于 40%的数据，有 $1 \le n \le 20$，$0 < a、b < 8$；
对于 60%的数据，有 $1 \le n \le 100$；
对于 60%的数据，保证答案不超过 10^9；
对于 100%的数据，有 $1 \le n \le 1{,}000$，$0 < a、b < 10000$。

3．开车旅行

(drive.cpp/c/pas)

【问题描述】

小 A 和小 B 决定利用假期外出旅行，他们将想去的城市从 1 到 N 编号，**且编号较小的城市在编号较大的城市的西边**，已知各个城市的海拔高度互不相同，记城市 i 的海拔高度为 H_i，城市 i 和城市 j 之间的距离 d[i,j]恰好是这两个城市海拔高度之差的绝对值，即$d[i,j] = |H_i - H_j|$。

旅行过程中，小 A 和小 B 轮流开车，第一天小 A 开车，之后每天轮换一次。他们计划选择一个城市 S 作为起点，**一直向东行驶**，并且最多行驶 X 公里就结束旅行。小 A 和小 B 的驾驶风格不同，小 B 总是**沿着前进方向选择**一个最近的城市作为目的地，而小 A 总是**沿着前进方向选择**第二近的城市作为目的地（注意：**本题中如果当前城市到两个城市的距离相同，则认为离海拔低的那个城市更近**）。如果其中任何一人无法按照自己的原则选择目的城市，或者到达目的地会使行驶的总距离超出 X 公里，他们就会结束旅行。

在启程之前，小 A 想知道两个问题：

1．对于一个给定的 $X=X_0$，从哪一个城市出发，小 A 开车行驶的路程总数与小 B 行驶的路程总数的比值最小（如果小 B 的行驶路程为 0，此时的比值可视为无穷大，且两个无穷

大视为相等）。如果从多个城市出发，小 A 开车行驶的路程总数与小 B 行驶的路程总数的比值都最小，则输出海拔最高的那个城市。

2. 对任意给定的 $X=X_i$ 和出发城市 S_i，小 A 开车行驶的路程总数以及小 B 行驶的路程总数。

【输入】

输入文件为 drive.in。

第一行包含一个整数 N，表示城市的数目。

第二行有 N 个整数，每两个整数之间用一个空格隔开，依次表示城市 1 到城市 N 的海拔高度，即 H_1，H_2，……，H_n，且每个 H_i 都是不同的。

第三行包含一个整数 X_0。

第四行为一个整数 M，表示给定 M 组 S_i 和 X_i。

接下来的 M 行，每行包含 2 个整数 S_i 和 X_i，表示从城市 S_i 出发，最多行驶 X_i 公里。

【输出】

输出文件为 drive.out。

输出共 M+1 行。

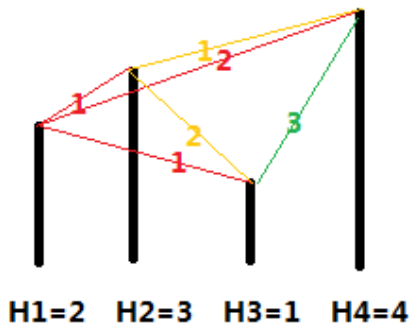
第一行包含一个整数 S_0，表示对于给定的 X_0，从编号为 S_0 的城市出发，小 A 开车行驶的路程总数与小 B 行驶的路程总数的比值最小。

接下来的 M 行，每行包含 2 个整数，之间用一个空格隔开，依次表示在给定的 S_i 和 X_i 下小 A 行驶的里程总数和小 B 行驶的里程总数。

【输入输出样例 1】

drive.in	drive.out
4 2 3 1 4 3 4 1 3 2 3 3 3 4 3	1 1 1 2 0 0 0 0 0

【输入输出样例 1 说明】



各个城市的海拔高度以及两个城市间的距离如上图所示。

如果从城市 1 出发，可以到达的城市为 2,3,4，这几个城市与城市 1 的距离分别为 1,1,2，但是由于城市 3 的海拔高度低于城市 2，所以我们认为城市 3 离城市 1 最近，城市 2 离城市 1 第二近，所以小 A 会走到城市 2。到达城市 2 后，前面可以到达的城市为 3,4，这两个城市与城市 2 的距离分别为 2,1，所以城市 4 离城市 2 最近，因此小 B 会走到城市 4。到达城市 4 后，前面已没有可到达的城市，所以旅行结束。

如果从城市 2 出发，可以到达的城市为 3,4，这两个城市与城市 2 的距离分别为 2,1，由于城市 3 离城市 2 第二近，所以小 A 会走到城市 3。到达城市 3 后，前面尚未旅行的城市为 4，所以城市 4 离城市 3 最近，但是如果要到达城市 4，则总路程为 2+3=5>3，所以小 B 会直接在城市 3 结束旅行。

如果从城市 3 出发，可以到达的城市为 4，由于没有离城市 3 第二近的城市，因此旅行还未开始就结束了。

如果从城市 4 出发，没有可以到达的城市，因此旅行还未开始就结束了。

【输入输出样例 2】

drive.in	**drive.out**
10 4 5 6 1 2 3 7 8 9 10 7 10 1 7 2 7 3 7 4 7 5 7 6 7 7 7 8 7 9 7 10 7	2 3 2 2 4 2 1 2 4 5 1 5 1 2 1 2 0 0 0 0 0

【输入输出样例 2 说明】

当 X=7 时，

如果从城市 1 出发，则路线为 1 -> 2 -> 3 -> 8 -> 9，小 A 走的距离为 1+2=3，小 B 走的距离为 1+1=2。（在城市 1 时，距离小 A 最近的城市是 2 和 6，但是城市 2 的海拔更高，视为与城市 1 第二近的城市，所以小 A 最终选择城市 2；走到 9 后，小 A 只有城市 10 可以走，没有第 2 选择可以选，所以没法做出选择，结束旅行）

如果从城市 2 出发，则路线为 2 -> 6 -> 7 ，小 A 和小 B 走的距离分别为 2，4。

如果从城市 3 出发，则路线为 3 -> 8 -> 9，小 A 和小 B 走的距离分别为 2，1。

如果从城市 4 出发，则路线为 4 -> 6 -> 7，小 A 和小 B 走的距离分别为 2，4。

如果从城市 5 出发，则路线为 5 -> 7 -> 8 ，小 A 和小 B 走的距离分别为 5，1。

如果从城市 6 出发，则路线为 6 -> 8 -> 9，小 A 和小 B 走的距离分别为 5，1。

如果从城市 7 出发，则路线为 7 -> 9 -> 10，小 A 和小 B 走的距离分别为 2，1。

如果从城市 8 出发，则路线为 8 -> 10，小 A 和小 B 走的距离分别为 2，0。

如果从城市 9 出发，则路线为 9，小 A 和小 B 走的距离分别为 0，0（旅行一开始就结束了）。

如果从城市 10 出发，则路线为 10，小 A 和小 B 走的距离分别为 0，0。

从城市 2 或者城市 4 出发小 A 行驶的路程总数与小 B 行驶的路程总数的比值都最小，但是城市 2 的海拔更高，所以输出第一行为 2。

【数据范围】

对于 30%的数据，有 $1 \leqslant N \leqslant 20$，$1 \leqslant M \leqslant 20$；

对于 40%的数据，有 $1 \leqslant N \leqslant 100$，$1 \leqslant M \leqslant 100$；

对于 50%的数据，有 $1 \leqslant N \leqslant 100$，$1 \leqslant M \leqslant 1,000$；

对于 70%的数据，有 $1 \leqslant N \leqslant 1,000$，$1 \leqslant M \leqslant 10,000$；

对于 100%的数据，有 $1 \leqslant N \leqslant 100,000$，$1 \leqslant M \leqslant 10,000$，$-1,000,000,000 \leqslant H_i \leqslant 1,000,000,000$，$0 \leqslant X_0 \leqslant 1,000,000,000$，$1 \leqslant S_i \leqslant N$，$0 \leqslant X_i \leqslant 1,000,000,000$，数据保证 H_i 互不相同。