编码干扰

（code）

题目描述：

Alice 和 Bob 此刻分别身处于两个相距遥远的国度。他们因为不能相见，所以只能通过互联网传输信息。简单来说，Alice 每次会向 Bob 发送一条长度为 k 位的 q 元码。所谓 q 元码，即编码每一位都是小于 q 的非负整数。

然而，信息在传输的过程中，往往会受到一些干扰。Bob 已经知道这个干扰的规模为非负整数 d，如果取 e = [(d-1) / 2] （即 d-1 除以 2 向下取整），则干扰会最多造成 e 位的错误。这里 e <= k，如果 e = 0 则不会产生任何错误，若 e = k，则对于一条长度为 k 的编码，有可能因为干扰而变成任意一条长度为 k 的编码。同时 Bob 还知道，这个干扰的规模不会因为编码长度的增加而变化，只是由地理因素决定的。也就是说，如果 Alice 传送一条长度为 k+1 的 q 元码给 Bob，编码在传输中收到的干扰规模仍然是 d。但是编码长度的增加却会带来更多的费用问题。

Bob 希望可以设计一个纠错系统，通过发送长度 n >= k 位的 q 元码，来避免因为干扰而造成收到错误的信息。这个纠错系统由两部分组成：CODE 系统与 DECODE 系统。Bob 希望设计好后，将 CODE 系统寄给 Alice，将 DECODE 系统留在自己手中。

之后 Bob 与 Alice 约定：每一次 Alice 在发出长度为 k 的编码 I 前，先通过 CODE 系统得到长度为 n 的编码 II（这里编码 I 与编码 II 都是 q 元码），然后将编码 II 发送给 Bob，编码 II 在发送途中会受到规模为 d 的干扰，所以 Bob 真实收到的编码 III 可能与 II 不同。最后 Bob 利用 DECODE 系统，通过编码 III 计算出编码 I。这就完成了纠错工作。

注意到：Alice 每一次发送的编码总是长度为 k 的 q 元码，且收到的干扰规模总是 d。传送编码的长度直接决定了传送费用（这里，通过互联网传送信息并不被认为是免费的，而需要与传送编码长度成正比的费用）。所以 Bob 希望这个系统在能实现正确纠错的基础上，每一次发出的编码 II 的长度 n 尽可能小。

程序要求：

本题要求选手编写两个独立的程序，分别对应 CODE 系统与 DECODE 系统。

对于 c/cpp 选手，需要提交 code.h 与 decode.h 程序。code.h 中的过程：

```
int Alice_solve(int k , int d , int q , int B[]);
```

为 CODE 系统，对于给定系数 k，d，q 和编码 I（用 B[1]到 B[k]保存），需要返回编码 II 的长度 n，并将编码 II 保存在 B[]中。

而 decode.h 中的过程：

```
void Bob_solve(int k , int d , int q , int B[]);
```

为 DECODE 系统，对于给定系数 k，d，q 与编码 III，假设编码 III 的长度为 n，则 B[1]

到 B[n]存储了编码 III，需要通过 B[]找出最早的编码 I（Alice 发出的原始编码），并保存在 B 中（B[1]到 B[k]）。

对于 code.h 与 decode.h，我们要求选手的两份程序之间不存在互相的调用关系，且我们禁止在程序中出现“#define”、“#if”等字符串。比赛中所提供的 forbidden_c.txt 列出了对于 c/cpp 选手所提交的 h 文件中禁止出现的字符串。

我们提供有评测程序 code_administrator.cpp，当选手的代码（code.h 与 decode.h）提交上来之后，我们会将 code_administrator.cpp 与 code.h、decode.h 一起编译（code_administrator 中有调用到 code.h 与 decode.h）。注意到 code_administrator.cpp 从输入文件 code.in 读入信息并对选手提供的纠错系统进行检测，检测结果将反馈到输出文件 code.out 中。需要注意的是，因为 code.h 与 decode.h 作为了一个程序被编译，所以重复的变量名定义（包括与 code_administrator.cpp 中变量出现重复的情况）有可能会造成程序编译无法通过。

我们为选手提供了一份参考代码，code.h 与 decode.h，以便选手可以更好地理解所需要完成的工作。

对于 pascal 选手，需要提交 code.pas 与 decode.pas。code.pas 中，前四行需要保证是：

```
unit code;
interface
function Alice_solve(k,d,q:longint;var B:array of longint):longint;
implementation
```

且最后两行需要保证是：

```
begin
end.
```

对于 decode.pas，后两行有着相同的要求，前四行要保证是：

```
unit decode;
interface
procedure Bob_solve(k,d,q:longint;var B:array of longint);
implementation
```

详细可以参见比赛中提供的文件 code.pas 与 decode.pas。

选手只需要编写 Alice_solve 和 Bob_solve 过程内的内容，变量可在 implementation 和过程申明之间申明。选手的 code.pas 与 decode.pas 不得存在相互调用的情况。为此，比赛中提供 forbidden_pas.txt 列出了所有对于 pascal 选手提交的程序应该被禁止出现的所有字符串。

我们提供了评测程序 code_administrator.pas。注意到 code_administrator.pas 从输入文件 code.in 读入信息并对选手提供的纠错系统进行检测，检测结果将反馈到输出文件 code.out 中。

我们为选手提供了一份参考代码，code.pas 与 decode.pas，以便选手可以更好地理解所需要完成的工作。

输入文件：

对于输入文件 code.in，第一行给出了系数 k，d，q。之后一行有一个整数 T，表示 Alice

要给 Bob 传送 T 条长度为 k 的 q 元码。之后 T 行，每行 k 个非负整数，描述了一条 q 元码。比赛中，我们提供了一份合法的 code.in 以供选手测试使用。

输出文件

输出文件一共 T 行，由 code_administrator 生成，每一行先是字符串“YES”或“NO”，说明了纠错系统是否正确，若为“YES”，之后一个整数为 n，即编码 II 的长度。比赛中，选手可以通过提供的程序和输入文件 code.in 得到一份合法的输出文件 code.out。

得分判定：

对于输出文件 code.out，如果出现“NO”，则纠错系统失败，不得分，考虑编码 II 的最大长度。根据评测系数 p(1)到 p(10)我们给出这一个数据点的得分情况。如果编码 II 的最大长度不超过 p(u)则得到至少 u 分，且每一个测试点的得分不超过 10 分。

数据规模：

数据编号	k	d	q
1	<=10	7	<=4
2	<=20	5	2
3	116	3	3
4	502	3	2
5	<=100	5	1000<=q<=10000
6	<=50	7	1000<=q<=10000
7	<=40	9	1000<=q<=10000
8	120	4	2
9	247	4	2
10	42	8	2

对于所有数据，T<=100。