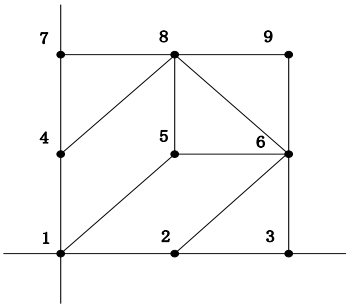

第一个开采计划，输入的第1个值为3，所以该开采计划对应的多边形有(3+0) mod 8 +1=4个点，将接下的4个数3,0,4,7，分别代入(z_i+0) mod n + 1得到4个点的编号为4,1,5,8。计算出第一个开采计划的分子为1，分母为1。

类似地，可计算出余下开采计划的多边形的点数和点的编号：

第二个开采计划对应的多边形有3个点，编号分别为5, 6, 8。

第三个开采计划对应的多边形有6个点，编号分别为1, 2, 6, 5, 8, 4。

第四个开采计划对应的多边形有5个点，编号分别为1, 2, 6, 8, 4。

第五个开采计划对应的多边形有6个点，编号分别为1, 5, 6, 8, 7, 4。

【数据范围】

对于 10%的数据，任一开发区域均为 1×1 的正方形。

对于 30%的数据，任一开发区域均为矩形（但矩形的一条边可能由平面图中的多条边组成）

对于 50%的数据，平面图中所有的边均平行于 x 轴或者 y 轴。

另有 20%的数据，$n,k \le 1000$。

对于 100%的数据，$n, k \le 2\times10^5$, $m \le 3n-6$, $|x_i|, |y_i| \le 3\times10^4$。所有开采计划的 d 之和不超过 2×10^6。

保证任何开采计划都包含至少一个开发区域，且这些开发区域构成一个连通块。

保证所有开发区域的矿量和不超过 $2^{63}-1$。

保证平面图中没有多余的点和边。

保证数据合法。由于输入数据量较大，建议使用读入优化。

【编译命令】

对于c++语言： g++ -o mine.in mine.out –lm

对于c语言： gcc -o mine.in mine.out –lm

对于pascal语言： fpc mine.pas

第 3 题：大数(number)，运行时限 2s，内存上限 128M。

【问题描述】

小 B 有一个很大的数 S，长度达到了 N 位；这个数可以看成是一个串，它可能有前导 0，

例如 00009312345。小 B 还有一个素数 P。

现在，小 B 提出了 M 个询问，每个询问求 S 的一个子串中有多少子串是 P 的倍数（0 也是 P 的倍数）。例如 S 为 0077 时，其子串 007 有 6 个子串：0,0,7,00,07,007；显然 0077 的子串 007 有 6 个子串都是素数 7 的倍数。

【程序文件名】

源程序文件名为 number.cpp/c/pas。

【输入格式】

输入文件名为 number.in。

第一行一个整数：P。

第二行一个串：S。

第三行一个整数：M。

接下来 M 行，每行两个整数 fr,to，表示对 S 的子串 S[fr…to]的一次询问。注意：S 的最左端的数字的位置序号为 1；例如 S 为 213567，则 S[1]为 2，S[1…3]为 213。

【输出格式】

输出文件名为 number.out。

输出 M 行，每行一个整数，第 i 行是第 i 个询问的答案。

【输入输出样例】

number.in	number.out
11 121121 3 1 6 1 5 1 4	5 3 2

【样例解释】

第一个询问问的是整个串，满足条件的子串分别有：121121,2112,11,121,121。

【数据范围】

对于 30%的数据，N,M<=1000；

对于 60%的数据，N<=10000，M<=1000；

对于 100%的数据，N,M<=100000，P 为素数。

【编译命令】

对于c++语言： g++ -o number number.cpp –lm

对于c语言： gcc -o number number.c –lm

对于pascal语言： fpc number.pas