2015/07/03

MCU Coding Club Contest

- Problems: There are 5 problems in this contest. (Total 11 pages excluded the cover page.)
- Input: We will provide inputs from *stdin* for all problems. Inputs may have multiple dataset according to each problem.
- Output: For all of the problems, answer must output to *stdout*.
- Time limit: The judges will run your programs in the limit according to each problem without any operation, including keyboard and mouse.
- The third problem is spiral related in order to tell the team Spiral that we look forward to see you in the contest.

Index of the Problems

	Problem Name
Problem A	Anagrams (II)
Problem B	Do Your Own Homework
Problem C	Spiral Tap
Problem D	Simple Division
Problem E	Patches

* Paste login information here *

Problem A
Anagrams (II)

Time limit: 3 seconds

One of the preferred kinds of entertainment of people living in final stages of XX century is filling in the crosswords. Almost every newspaper and magazine has a column dedicated to entertainment but only amateurs have enough after solving one crossword. Real professionals require more than one crossword for a week. And it is so dull - just crosswords and crosswords - while so many other riddles are waiting out there. For those are special, dedicated magazines. There are also quite a few competitions to take part in, even reaching the level of World Championships. Anyway - a lot.

You were taken on by such a professional for whom riddle solving competing is just a job. He had a brilliant idea to use a computer in work not just to play games. Somehow anagrams found themselves first in the line. You are to write a program which searches for anagrams of given words, using a given vocabulary, tediously filled with new words by yours employer.

- Input

  The first line of the input is an integer *M*, then a blank line followed by *M* datasets. There is a blank line between datasets. The structure of each dataset is given below:

```
<number of words in vocabulary>
<word 1>
..............
<word N>
<test word 1>
.................
<test word k>
END
```

  <number of words in vocabulary> is an integer number $N < 1000$.

  <word 1> up to <word *N*> are words from the vocabulary.

  <test word 1> up to <test word *k*> are the words to find anagrams for.

  All words are lowercase (word END means end of data - it is NOT a test word). You can assume all words are not longer than 20 characters.

- Output

  For each <test word> list the found anagrams in the following way:

```
Anagrams for: <test word>
<No>) <anagram>
................
```

  <No> should be printed on 3 chars.

  In case of failing to find any anagrams your output should look like this:

```
Anagrams for: <test word>
No anagrams for: <test word>
```

  Print a blank line between datasets.

- Sample Input

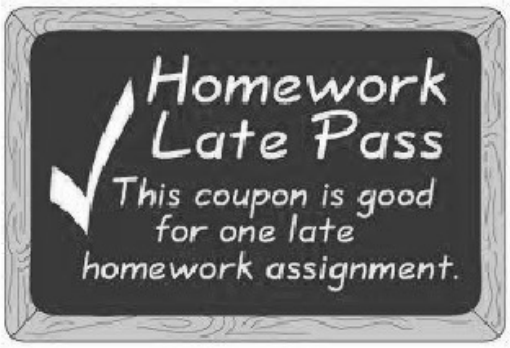
```
1

8
atoll
lato
microphotographics
rata
rola
tara
tola
pies
tola
kola
aatr
photomicrographics
END
```

- Sample Output

```
Anagrams for: tola
   1) atol
   2) lato
   3) tola
Anagrams for: kola
No anagrams for: kola
Anagrams for: aatr
   1) rata
   2) tara
Anagrams for: photomicrographics
   1) microphotographics
```

Problem B
Do Your Own Homework

Time limit: 1 second



These days Soha is so busy that he doesn't have time to do his own homework. But this is not a big problem since he has got many friends who are willing to help. One of his friend's name is Sparrow. Whenever Soha is assigned any homework, he turns to Sparrow for her help.

Sparrow has given a list of subjects that she is comfortable with along with the number of days it will take her to complete an assignment for each subject. Soha has got only *D* days to complete his next assignment. However, the professor of this subject is a little flexible and allows late submissions up to 5 days. That means he will not accept any submission that is after *D* + 5 days from now. Will Sparrow be able to do it for Soha this time?

■ Input

First line of input is a positive integer *T* that determines the number of test cases. Each case starts with a line containing an integer *N* that represents the number of subjects Sparrow is comfortable with. Each of the next *N* lines contain the name of a subject followed by the number of days it will take Sparrow to complete an assignment of that subject. All these subject names will be distinct. The next line contains an integer *D*. The meaning of *D* is described above. The following line contains the name of the subject whose homework is due. All the subjects' names consist of lowercase letters and the length of each is at least 1 and at most 20. All the integer inputs are positive in the range [1, 100].

■ Output

For each case, first output the case number first starting from 1.
If Sparrow doesn't take more than *D* days to completely the assignment, output '`Yesss`'; if she takes more than *D* days but not more than *D* + 5, output '`Late`'; if she takes more than *D* + 5 days or if she isn't comfortable with the subject, output '`Do your own homework!`'. Quotes are for clarify only and don't need to be part of the output. Look at the samples for more details. Be careful about the spelling.

■ Sample Input

```
3
3
compiler 4
cplusplus 1
java 8
5
compiler
2
algorithm 3
math 9
4
math
2
java 8
ai 3
6
calculus
```

■ Sample Output

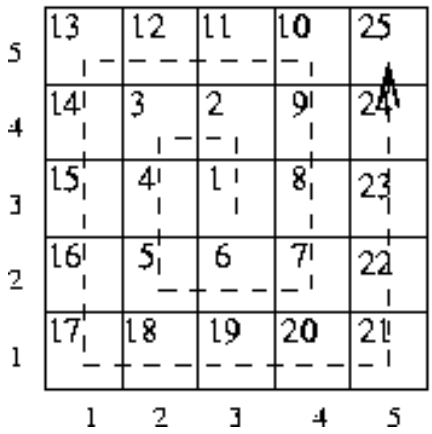
```
Case 1: Yesss
Case 2: Late
Case 3: Do your own homework!
```

Problem C
Spiral Tap

Time limit: 1 second

The game of Spiral Tap is played on a square grid. Pieces are placed on a grid and the moves are realized according to the position of the pieces on the grid. However, the coordinate systems in the game of Spiral Tap are a bit different that those find in traditional board games, such as chess.

The cell-numbering schemes follow a spiral, starting from the center of the grid in an anti-clockwise fashion. The following figure illustrates the cell numbering scheme.



The goal is, given the spiral tap coordinates of a cell; find its Cartesian coordinates (line *1* is at the bottom, and column *1* is the leftmost).

■ Input

The input is a series of lines. Each line is composed of two numbers: *SZ* and *P*. *SZ* is the size of the border of the grid and is an odd number no larger than *100000*. *P* is the spiral position of a cell in this grid. The line such that *SZ = P = 0* marks the end of the input (and is not part of the data set).

■ Output

For each line in the data set of the input, your program must echo a line "`Line = X, column = Y.`", where *X* and *Y* are the Cartesian coordinates of the corresponding cell.

■ Sample Input

```
3 1
3 3
3 9
5 9
5 10
0 0
```

■ Sample Output

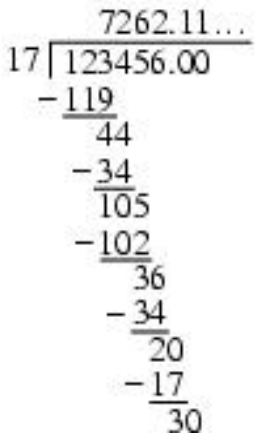
```
Line = 2, column = 2.
Line = 3, column = 1.
Line = 3, column = 3.
Line = 4, column = 4.
Line = 5, column = 4.
```

Problem D
Simple Division

Time limit: 3 seconds

Integer division between a dividend n and a divisor d yields a quotient q and a remainder r. q is the integer which maximizes $q * d$ such that $q * d \leq n$ and $r = n - q * d$.

For any set of integers there is an integer d such that each of the given integers when divided by d leaves the same remainder.



■ Input

Each line of input contains a sequence of nonzero integer numbers separated by a space. The last number on each line is 0 and this number does not belong to the sequence. There will be at least 2 and no more than 1000 numbers in a sequence; not all numbers occurring in a sequence are equal. The last line of input contains a single 0 and this line should not be processed.

■ Output

For each line of input, output the largest integer which when divided into each of the input integers leaves the same remainder.

Sample Input

```
701 1059 1417 2312 0
14 23 17 32 122 0
14 -22 17 -31 -124 0
0
```

Sample Output

```
179
3
3
```

Problem E
Patches

Time limit: 2 seconds

Carlos is very concerned with the environment. Whenever possible, he tries to use less polluting means of transport. He recently got a job close to home and is now using his bike to go to work.
Unfortunately, in the route between his home and his job there is a nail factory, and often some nails fall from their trucks, and end up puncturing Carlos' bike tires. Therefore he ends up having to make several patches on the tires of his bike.

To make the repairs, Carlos uses two different types of patches. Both types are as wide as a bike tire, but differ in length. As the cost of the patch is proportional to its length, Carlos is trying to find a way to save money, using the least possible length of patches to make the repairs, without cutting the patches.

The first step in repairing a tire is making a chalk mark on a position of the tire and then writing down the distances, measured clockwise, of each of the holes in relation to the chalk mark. Each hole must be completely covered by a patch. Carlos would like your help to determine, given the positions of the holes, the most economic way to make the repair.

■ Input

The input contains several test cases. Each test case is composed of two lines. The first line contains four integers N, C, T_1 and T_2.
Integer N indicates the number of holes in the tire, and C indicates the circumference length of the tire, in centimeters. The lengths of the patches in centimeters are given by integers T_1 and T_2. The second line contains N integers F_i, representing the distance, in clockwise direction, from the chalk mark to hole i, in centimeters.

■ Output

For each test case your program must print a single line, containing a single integer, the smallest total length of patches needed to make all the repairs.

Restrictions

- $1 \le N \le 1000$
- $1 \le C \le 10^6$
- $1 \le T_1, T_2 \le C$
- $0 \le F_i \le C - 1, 1 \le i \le N$
- If the distance between two holes is exactly k centimeters, a patch of length k centimeters covers both holes.

■ Sample Input

```
5 20 2 3
2 5 8 11 15
4 20 12 9
1 2 3 13
```

■ Sample Output

```
8
12
```