8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 1 – Approaching the sine function.
Executable Program : Prog1.exe, Prog1.class
Source Program: Prog1.cpp, Prog1.java, Prog1.c, Prog1.pas
Input file: Prog1.in **Output file:** Prog1.out

Let $S(x,n)=\sum_{k=0}^{n}\frac{(-1)^k x^{2k+1}}{(2k+1)!}$ and note that $\lim_{n\to\infty} S(x,n)=\sin(x)$.
Write a program that finds the minimum value n so that, for a given real value x and an integer d, we have that both $S(x,n)$ and sin(x), rounded to d decimal places, coincide.

For example, given x=1 and d=5, then since:

sin(1) ¿ 0.841470985 rounded to 5 decimal places is 0.84147,

and

$S(1,3)=\frac{4241}{5040}\approx 0.841468254$ rounded to 5 decimal places is 0.84147,

but

$S(1,2)=\frac{101}{120}\approx 0.841666667$ rounded to 5 decimal places is 0.84167,

then 4 terms of the sum $\sum_{k=0}^{n}\frac{(-1)^k x^{2k+1}}{(2k+1)!}$, (i.e., k=0, 1, 2, 3) are enough for approximating sin(1) to five decimal places. That is, the minimum value for n, in this case is 3.

INPUT
Each line of an input text file contains two numbers `x d`. Process them until the end of file.

OUTPUT
For each input line output a single integer n.

Sample input
```
1 5
3.1416 7
-4 2
0 12
```

Sample output
```
3
8
6
0
```

8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 2 – Super-Mazes
Executable Program : Prog2.exe, Prog2.class
Source Program: Prog2.cpp, Prog2.java, Prog2.c, Prog2.pas
Input file: Prog2.in **Output file:** Prog2.out

A super-maze is a maze where any two cells are guaranteed to be connected by exactly one path, with no entries from or exits to the exterior. Write a program that reads super-mazes, displays them, and finds the path that takes from the upper-left corner to the lower-right.

INPUT
The input file consists of various super-mazes to be processed. The definition of each maze begins with a line with two integers r and c ($0 \le r < 20$, $0 \le c < 20$) that represent the number of rows and columns in the maze, followed by r lines, each with c integers whose values are all between zero and three, inclusive. If $r = 0$ or $c = 0$, it means no more mazes.

Each entry in this matrix represents the walls (or openings) of a cell to the right and bottom. An even value (0 and 2) represents a cell with a wall (no opening) to its right, while an odd value (1 and 3), a cell with no wall (an opening) to its right. Similarly, a low value (0 and 1) represents a cell with a wall (no opening) to its bottom, while a high value (2 and 3), a cell with no wall (an opening) to its bottom. Left and top walls (openings) of a cell can be determined by examining values in the corresponding adjacent cells.

Each entry in the input file is guaranteed to represent a super-maze.

OUTPUT
For each maze, display a line with the title “Super-maze k:”, where k is a consecutive number starting from 1, followed by a graphical representation for the maze. Then, in the following line(s), the title “Solution:”, followed by one space and a string composed of the characters R, L, U, D, (Right, Left, Up, Down) that represents a path for the solution. Put no more than 80 characters per line and leave a blank line between runs (see sample output).

Sample input
```
3 3
1 3 2
1 2 0
1 1 0
4 7
2 2 3 0 3 3 2
3 0 3 1 0 0 2
1 3 3 0 3 0 2
1 0 1 1 0 1 0
15 15
3 0 2 3 3 0 2 1 1 2 2 1 1 3 0
2 2 1 2 3 2 1 2 1 1 3 1 1 2 2
3 1 1 2 0 3 3 3 2 2 0 3 0 3 0
3 0 3 2 2 0 0 0 3 1 3 2 1 3 2
0 2 2 3 0 1 1 3 3 0 2 3 0 2 0
2 2 0 0 2 3 1 2 0 2 2 1 1 3 0
2 2 2 3 1 3 0 0 2 1 0 1 2 2 2
1 3 1 2 1 1 0 2 3 1 1 2 3 1 0
1 3 0 2 2 1 1 2 2 3 2 0 1 0 2
2 3 0 3 3 0 3 1 3 2 0 3 1 3 0
3 1 0 0 1 0 0 2 0 3 1 3 2 3 0
```

```
2 1 2 2 1 3 1 2 2 2 3 2 0 1 2
1 3 1 3 0 2 1 1 3 0 0 3 2 3 0
2 3 0 0 3 3 0 2 1 0 3 0 0 2 2
1 1 1 1 0 1 1 0 1 1 0 1 1 1 0
0 0
```

Sample output

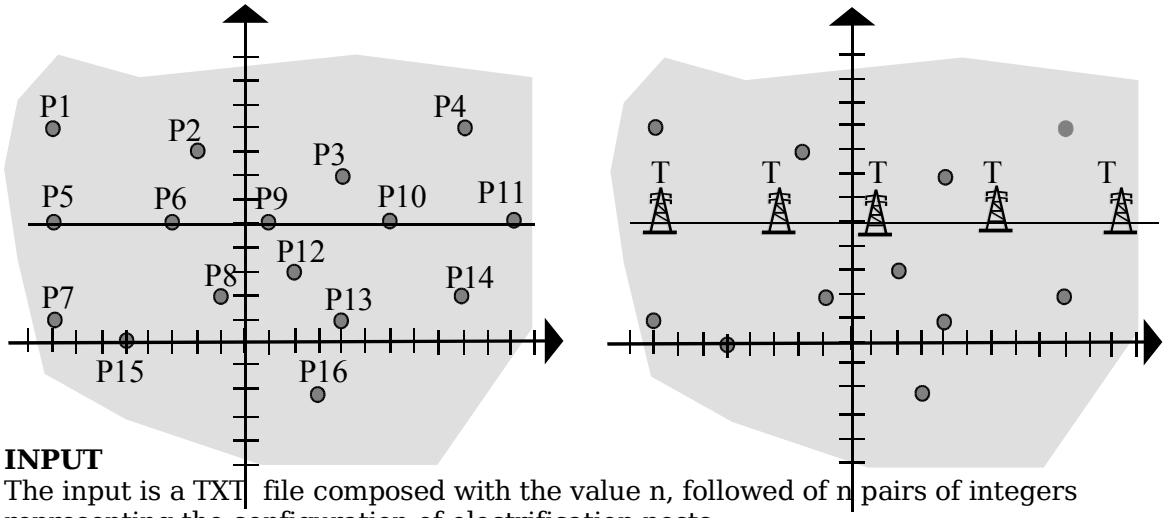
```
Super-maze 1:
+---+---+---+
|           |
+---+   +   +
|       |   |
+---+   +---+
|           |
+---+---+---+
Solution: RDDR

Super-maze 2:
+---+---+---+---+---+---+---+
|   |   |       |           |
+   +   +   +---+   +   +   +
|       |           |   |   |
+   +---+   +---+---+---+   +
|               |       |   |
+---+   +   +---+   +---+   +
|       |           |       |
+---+---+---+---+---+---+---+
Solution: DDRRURRURRDDD

Super-maze 3:
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|       |   |           |   |           |   |               |
+   +---+   +   +---+   +---+---+   +   +---+---+   +---+
|   |   |       |       |           |                   |   |
+   +   +---+   +   +   +---+   +---+---+   +---+---+   +   +
|               |   |               |   |   |       |       |
+   +---+---+   +---+   +   +   +   +   +---+   +---+   +---+
|       |       |   |   |   |   |               |           |
+   +---+   +   +   +---+---+---+   +---+   +   +---+   +   +
|   |   |   |       |                   |   |       |   |   |
+---+   +   +   +---+---+---+   +   +---+   +   +---+   +---+
|   |   |   |   |   |           |   |   |   |               |
+   +   +---+---+   +   +---+   +---+   +   +---+---+   +---+
|   |   |   |               |   |   |       |       |   |   |
+   +   +   +   +---+   +---+---+---+   +---+---+---+   +   +   +
|               |           |   |               |           |
+---+   +---+   +---+---+---+   +   +---+---+   +   +---+---+
|           |   |   |           |   |       |   |       |   |
+---+   +---+   +   +---+---+   +   +   +   +---+---+---+   +
|   |       |           |               |   |               |
+   +   +---+   +   +---+   +---+   +---+   +---+   +   +---+
|           |   |       |   |   |   |               |       |
+   +---+---+---+---+---+---+   +---+   +---+   +   +   +---+
|   |       |   |               |   |   |       |   |       |
+   +---+   +   +---+   +---+   +   +   +   +---+---+   +   +
|                   |   |               |   |       |       |
+---+   +---+   +---+   +---+---+   +---+---+   +   +   +---+
|   |       |   |           |   |       |       |   |   |   |
+   +   +---+---+   +   +---+   +---+---+   +---+---+   +   +
|                   |           |           |               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
Solution: DDRRRUURDRDRRRDDLDLLDLLDLLDDDLDDRDDRRRURUURRDRRUURRURRDDRDLDDR
```

8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 4 – Electric Reconfiguration
Executable Program : Prog4.exe, Prog4.class
Source Program: Prog4.cpp, Prog4.java, Prog4.c, Prog4.pas
Input file: Prog4.in **Output file:** Prog4.out

The electric company is reconfiguring the electric distribution system in the farm. The engineer in charge is evaluating the feasibility to develop the project. The idea is to determinate the maximum number of electrification posts lined up in that zone to replace them by high voltage towers, the cost of the project depends exclusively of that number. He was hired to build a program to determinate it. The program input will be composed by the number n (5 <= n <= 100) of electrification posts, and their configuration in a coordinated system. The output is the maximum number of electrification posts lined up.



INPUT
The input is a TXT file composed with the value n, followed of n pairs of integers representing the configuration of electrification posts.

OUTPUT
The output file has the OUT extension, and gives the integer representing the maximum number of electrification posts lined up.

Sample Input

```
16
-8 9
-2 8
4 7
9 9
-8 5
-3 5
-8 1
-1 2
1 5
6 5
11 5
2 3
4 1
9 2
-5 0
3 -2
```

Sample Output

```
5
```

8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 5 – Groupings
Executable Program : Prog5.exe, Prog5.class
Source Program: Prog5.cpp, Prog5.java, Prog5.c, Prog5.pas
Input file: Prog5.in **Output file:** Prog5.out

A common requirement in report generation for commercial application software, is the presentation of information in a categorized or group form. Nowadays there are software components and report applications that allow the user to visualize his information, (normally stored in a Data Base) in an aggregated or group form, so that the user could take the most advantage of his information.

You were ask to make a small program that will receive a text file as input with a set of data in a tabular form, as well as some set of instructions to make groups and generate an output file with the data grouped as requested. The following data correspond to the Customers table in the Database Northwind stored in MSSQL Server.

CustomerID	CompanyName	ContactTitle	City	Country
ANATR	Ana Trujillo Emparedados	Owner	México D.F	Mexico
ANTON	Antonio Moreno Taquería	Owner	México D.F	Mexico
AROUT	Around the Horn	Sales Representative	London	UK
BERGS	Berglunds snabbköp	Order Administrator	Luleå	Sweden
BLAUS	Blauer See Delikatessen	Sales Representative	Mannheim	Germany
BLONP	Blondesddsl père et fils	Marketing Manager	Strasbourg	France
BOLID	Bólido Comidas preparadas	Owner	Madrid	Spain
ALFKI	Alfreds Futterkiste	Sales Representative	Berlin	Germany

In this case it was requested to group the information by Country

CustomerID	CompanyName	ContactTitle	City	Country
Country: France				
BLONP	Blondesddsl père et fils	Marketing Manager	Strasbourg	France
Country: Germany				
ALFKI	Alfreds Futterkiste	Sales Representative	Berlin	Germany
BLAUS	Blauer See Delikatessen	Sales Representative	Mannheim	Germany
Country: Mexico				
ANATR	Ana Trujillo Emparedados	Owner	México D.F	Mexico
ANTON	Antonio Moreno Taquería	Owner	México D.F	Mexico
Country: Spain				
BOLID	Bólido Comidas preparadas	Owner	Madrid	Spain
Country: Sweden				
BERGS	Berglunds snabbköp	Order Administrator	Luleå	Sweden
Country: UK				
AROUT	Around the Horn	Sales Representative	London	UK

INPUT

The program will receive as input several lines the first one is the string [DATA], then there will be n+1 lines, each one will be a set of strings separated by the pipe symbol "|", there may be characters form other language in some strings. Of the n+1 lines the first one correspond to the title of the information. Following the n+1 lines there should be a line with [GROUPS]. After this line you should find m line each one indicating the groups to make and their order.

OUTPUT

The data and groups should be indented with one or several tab characters, depending their level in the hierarchy father-son, for example:

```
[GROUP]
[TAB][GROUP]
[TAB][TAB][DATA]
```

Sample Input:

```
[DATA]
CustomerID|CompanyName|ContactName|ContactTitle|City|Country
BOTTM|Bottom-Dollar Markets|Elizabeth Lincoln|Accounting Manager|BC|Canada
BSBEV|B's Beverages|Victoria Ashworth|Sales Representative|London|UK
CACTU|Cactus Comidas para llevar|Patricio Simpson|Sales Agent|Buenos Aires|Argentina
CENTC|Centro comercial Moctezuma|Francisco Chang|Marketing Manager|México D.F.|Mexico
CHOPS|Chop-suey Chinese|Yang Wang|Owner|Bern|Switzerland
COMMI|Comércio Mineiro|Pedro Afonso|Sales Associate|Sao Paulo|Brazil
CONSH|Consolidated Holdings|Elizabeth Brown|Sales Representative|London|UK
DRACD|Drachenblut Delikatessen|Sven Ottlieb|Order Administrator|Aachen|Germany
DUMON|Du monde entier|Janine Labrune|Owner|Nantes|France
EASTC|Eastern Connection|Ann Devon|Sales Agent|London|UK
ERNSH|Ernst Handel|Roland Mendel|Sales Manager|Graz|Austria
FAMIA|Familia Arquibaldo|Aria Cruz|Marketing Assistant|Sao Paulo|Brazil
FISSA|FISSA Fabrica Inter. Salchichas S.A.|Diego Roel|Accounting Manager|Madrid|Spain
FOLIG|Folies gourmandes|Martine Rancé|Assistant Sales Agent|Lille|France
FOLKO|Folk och fä HB|Maria Larsson|Owner|Bräcke|Sweden
[GROUPS]
Group:Country
Group:City
```

Sample Output:

```
Country: Argentina
        City: Buenos Aires
                CACTU|Cactus Comidas para llevar|Patricio Simpson|Sales Agent|Buenos Aires|
Argentina
Country: Austria
        City: Graz
                ERNSH|Ernst Handel|Roland Mendel|Sales Manager|Graz|Austria
Country: Brazil
        City: Sao Paulo
                FAMIA|Familia Arquibaldo|Aria Cruz|Marketing Assistant|Sao Paulo|Brazil
                COMMI|Comércio Mineiro|Pedro Afonso|Sales Associate|Sao Paulo|Brazil
Country: Canada
        City: BC
                BOTTM|Bottom-Dollar Markets|Elizabeth Lincoln|Accounting Manager|BC|Canada
Country: France
        City: Lille
                FOLIG|Folies gourmandes|Martine Rancé|Assistant Sales Agent|Lille|France
        City: Nantes
                DUMON|Du monde entier|Janine Labrune|Owner|Nantes|France
Country: Germany
        City: Aachen
                DRACD|Drachenblut Delikatessen|Sven Ottlieb|Order Administrator|Aachen|
Germany
Country: Mexico
        City: México D.F.
                CENTC|Centro comercial Moctezuma|Francisco Chang|Marketing Manager|México
D.F.|Mexico
Country: Spain
        City: Madrid
                FISSA|FISSA Fabrica Inter. Salchichas S.A.|Diego Roel|Accounting Manager|
Madrid|Spain
Country: Sweden
        City: Bräcke
                FOLKO|Folk och fä HB|Maria Larsson|Owner|Bräcke|Sweden
Country: Switzerland
        City: Bern
                CHOPS|Chop-suey Chinese|Yang Wang|Owner|Bern|Switzerland
Country: UK
        City: London
                CONSH|Consolidated Holdings|Elizabeth Brown|Sales Representative|London|UK
                EASTC|Eastern Connection|Ann Devon|Sales Agent|London|UK
                BSBEV|B's Beverages|Victoria Ashworth|Sales Representative|London|UK
```

8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 6 – Emergency Services
Executable Program : Prog6.exe, Prog6.class
Source Program: Prog6.cpp, Prog6.java, Prog6.c, Prog6.pas
Input file: Prog6.in **Output file:** Prog6.out

As result of the intense rain, many cities in the country as well as their habitants are suffering flooding. Most of the time emergency services were not able to respond promptly to the habitants requests for help. As a contingency plan city authorities have decided to assign Emergency Anti Flooding Teams (EAFT) to strategic points in the city to assist the habitants help calls during the flooding. The EAFT task is to go from their base point to the place where there is a problem, As soon as possible and release the drainage lid open, so the water level decrease faster.

When it starts to rain and some person see that the water level is raising in his street, He must call the EAFT center, where the responsible of the center should send the nearest team.

Your task is to develop a program that finds the routes that the teams should follow to assist to the habitants calls for help. So that the water level in their streets reminds as low as possible.

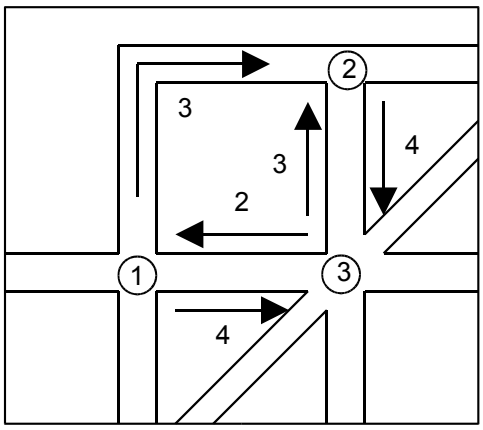
You should consider the following assumptions:

- The drainage lids are located in street intersections.
- The EAFT are located in street intersections.
- If you can go directly from intersection A to intersection B, that does not imply that you could go directly from intersection B to intersection A. (The street may be ONE way only).
- The time elapsed to go from one intersection to another is known.
- Even though not all intersection can be reach directly, there will always be a possible route between two points.

INPUT
The input to your program is a file name prog8.in with the following information.
There may be one or more sets of data. The first line is a number that defines how many sets are in the file.
With in each set of data:
the first line is the number of intersections in the map, its value is **n**.
From the second line up to the line **n+1** you find the values of a square matrix of size **n X n**, that represents the time required to go from intersection **i** (rows) to intersection **j** (columns). When **i=j** (since the start and the destination are the same) the time is cero. The time is –1 when the intersections do not connect directly between them.
The line n+2 has the following values: a number **p** that represent the place where there is a problem. Then there may be one or more numbers representing the locations of the EAFT.

OUTPUT
The output should have one line with the set number, as follows:
Set Number #, Where "#" should be replaced for 1, 2, etc. then, With in each set,
there should be one or more lines, (at least) one for each EAFT representing the shortest path from the starting point of the EAFT to the point of problem **p**. The format of each line should be: the number of the starting point of the EAFT, the number of the intersection with a problem of flooding, the time elapsed to get there and the path followed. (the path is a set of intersections form the start to the destination) All the values of these line must be separated by a tabulator. The lines must be ordered from the least to the greatest time, …

Example of a City map



Sample Input

```
1
6
0       3       4       -1      -1      -1
-1      0       4       5       -1      -1
2       3       0       -1      -1      2
8       9       5       0       1       -1
7       2       1       -1      0       -1
5       -1      4       5       4       0
2       4       5       6
```

Sample Output

Set Number 1

```
5       2       2       5       2
4       2       3       4       5       2
6       2       6       6       5       2
```

8th México and Central America
ACM International Collegiate Programming Contest
Problem No. 7 – Indexing Web Pages
Executable Program : Prog7.exe, Prog7.class
Source Program: Prog7.cpp, Prog7.java, Prog7.c, Prog7.pas
Input file: input.txt **Output file:** output.txt

Write a program to create an index of a small collection of World Wide Web pages. Each page is a text file in a special format called HTML (HyperText Markup Language). The HTML format includes regular text and special HTML tags, which are always enclosed in angle braces. For example, the string `<A HREF="layout.html">` is an HTML tag meaning that the following text should be a link to the file `layout.html`.

Your program's task is to read an HTML file called `index.html` and all the files referenced within `index.html` by the `HREF` command and all the files referenced by those files, and so on until there were not new files to read. Your program should also read the file `input.txt` containing a list of words and show a list of all the files referenced from `index.html` which contain each word (see the Sample Output).

Assumptions

- Any opening angle bracket (the "<" character) will be followed sooner or later by a matching closing angle bracket (the ">" character).
- A word is any string of characters found in a file that:
  - Does not lie between matching angle braces.
  - Contains only letters (no spaces, hyphens, apostrophes, etc.)
  - Is not part of a longer word (e.g., in "balloon", we would not consider "loon" to be a word).
- Words which differ only in case should be considered to be the same. Thus, "Word", "word", "WORD", and "wOrD" would be considered to be the same word.
- The only HTML tag you need to worry about is the HREF tag, and you can assume that it will always be in the form <A HREF="filename">, with no additional spaces or other characters; the name of the file is legal and in the same directory as the file you are already reading; and the name of the file will not exceed twelve characters in length. Filenames will always end with ".html".
- HTML files may be mutually referential or self referential, but there will be at most one hundred different files to read.

INPUT

You should read the file `input.txt` which contains in the first line the name of the initial HTML file you should start indexing. The next n lines contain the words that will be searched, one word per line, with no additional spaces.

OUTPUT

Write a file named `output.txt` with the list of each word found in the `input.txt` file (except for the first line, which is the initial HTML file) followed by a list of the file names the word is found in, in the following format:

```
"word" can be found in the following pages:
     filename1
     filename2

"word" can be found in the following pages:
     filename3

"word" can not be found in any page.
```

Where `word` is the word from the input file, and `filename1`, `filename2`, and so on, are the names of the files containing the word. Each file name should be indented five spaces: a single blank line should separate each listing.

Sample Input (there are three files)

input.txt

```
index.html
file
index
html
HTML
recursion
word
is
```

index.html

```
<HTML>
<HEAD>
<TITLE>Indexing Web Pages</TITLE>
</HEAD>
<BODY>
<P>Write a program to create an index of a small collection
of World Wide Web pages.  Each "page" is a text file in a
special format called HTML (HyperText Markup Language).  The
HTML format includes regular text and special HTML tags,
which are always enclosed in angle braces.  For example, the
string <A HREF="layout.html"> is an HTML command meaning that
the following text should be highlighted; a user click on
the highlighted text would cause a web browser to fetch and
display the file layout.html.</P>
<H1>Following Links</H1>
<P>Don't forget that links can be <A HREF="index.html">
self-referential</A>!</P>
</BODY>
</HTML>
```

layout.html

```
<A bunch of gibberish and a word>
Note that there is no rule that the file needs to be legal HTML
(if you know the rules), or that words really be wordseiwlaoieu;a.
<A HREF="index.html">Watch out for mutual references!
</HTML>
```

Sample Output

```
"file" can be found in the following pages:
     index.html
     layout.html

"index" can be found in the following pages:
     index.html

"html" can be found in the following pages:
     index.html
     layout.html

"HTML" can be found in the following pages:
     index.html
     layout.html

"recursion" can not be found in any page.

"word" can not be found in any page.

"is" can be found in the following pages:
     index.html
     layout.html
```