Computer Olympiad '98 Task Overview Day 2

All directories are located in 'C:\DAY2'

Note: Your programs are required to read input data from the current directory and write output data to the current directory. Do not use hard-coded paths

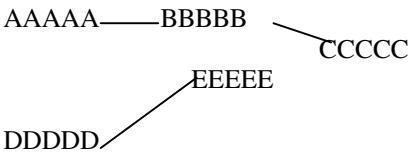
TASK:	Towns	Shelves	Triangle
Directory:	TOWNS	SHELVE	TRI
Prog. name:	TOWNS.EXE	SHELVE.EXE	TRI.EXE
Input file:	TOWNS.DAT	SHELVE.DAT	TRI.DAT
Output file	TOWNS.OUT	SHELVE.OUT	TRI.OUT
Max time:	20 Seconds	20 Seconds	20 Seconds
Test cases:	5	5	5
Total points:	100	100	100

The Maximum score for day 1 is 300 point

Towns

Connecting Towns

Given a list of roads directly connecting the pair of towns you are required to find how many disjoint sets of towns there are. Two towns are in the same set if it is possible to reach one from the other by one road or a combination of roads. Two towns are in disjoint sets if it is not possible to get between two towns on the given roads.
For example there are two disjoint sets, {AAAAA,BBBBB,CCCCC} and {DDDDD,EEEEE} in the following map:



Task

You are required to write a program that reports the number of disjoint sets of towns there are in the input data

Input

The input file name is **`TOWNS.DAT`**
The input will consist of a list of roads (given by the two towns they connect), terminated with the word XXXXX .

Constraints

- All town names will consist of five letters. There will be no separate list of towns given. You should assume that the only towns of interest are those given in the road list.
- No towns will be named XXXXX and no two towns will share the same name.
- There will be no more that 250 towns in any list, though the list may contain several millions of roads. Note roads can be travelled in both directions, and may be given either way round. Due to the carelessness of the mapmakers, town planning committees and the Pentium floating point bug, roads may be listed more than once.

Output

The output filename is **`TOWNS.OUT`**
Your output should consist of a number of towns, followed on the same line by a single space and then the number of disjoint sets.

Example 1

Input:

```
AAAAA BBBBB
CCCCC BBBBB
DDDDD EEEEE
XXXXX
```

Output:

```
5 2
```

Example 2

Input:

```
WURMF AMFPE
IGHSD NARES
GRIDL NDATF
NARES DIBBE
XXXXX
```

Output:

```
7 3
```

Scoring

- An exact solution will score 100%
- A solution with the correct number of towns but incorrect number of subsets will score 50%
- A Solution with the correct number of disjoint sets but with the wrong number of towns will score 25%
- Any other solution will score 0%

Shelves

Shelves

There are two shelves filled with books somewhere. Your task is to write a program that will move the books into numerical order: volume 1 at the top left, volume 2 next to it, etc. Each book is numbered from 1 to *N*, where *N* is the number of books in the set (all of the books are part of an encyclopaedia set.) This is a difficult job, because all of the books need to be moved using a special crane, and the amount of energy used by the crane must be minimised. All of the books are exactly the same size, shape and weight.

How the crane moves the books

N is always odd. Initially there are (*N*+1)/2 books on the top shelf, and *N*/2 books on the bottom shelf. This leaves space for one book, and the crane starts in this space (it is the right-most position of the shelf.) The crane always starts in the right-most position of the bottom shelf, and when the books are correctly ordered it must move to that position again. It can transport 1 book at a time by removing it from the shelf, transporting it to a free space, and placing the book on the shelf in the free space (all three of these actions require the crane to use energy.) The crane can move from the top shelf to the bottom shelf or from the bottom to the top. It may do so at any time, however if the crane is transporting a book this movement requires *E* units of energy; if it is not transporting a book the movement requires 1 unit of energy. The same amount of energy is required to move up from the bottom shelf, or down from the top shelf. The value of *E* is provided in the input file. The crane can move only rightwards when it is working on the top shelf. It may not move leftwards here. Similarly it can only move leftwards while on the bottom shelf, and it cannot move rightwards here.

The energy used to transport books

All cost is dealt with in energy units. The crane uses 1 energy unit:
to pick a book up off the shelf
to put a book down onto the shelf
to move a book horizontally by a distance equal to the width of one book
to move up to the top shelf if it is not transporting a book
or to move down to the bottom shelf if it is not transporting a book

If the crane is not transporting a book it moves horizontally without using any units of energy.

Input format

The input consists of integers only, one integer on each line.
The first number is N, the number of encyclopaedias in the set. N is always odd, greater than 0 and less than or equal to 1001.
The next number is E, the cost of moving the crane from the top shelf to the bottom shelf, or from the bottom to the top. E is greater than or equal to zero, and it is expressed in energy units.
The following N numbers in the input file are the book numbers of the encyclopaedia set. Each book is uniquely numbered from 1 to N. The first book number that appears in the input file, (the number on the third line of the file), is the volume number of the left-most book on the top shelf. The next number is the book to its right. The number on line, 2+(N+1)/2 indicates the volume of the right-most book on the top shelf. The next number in the input file is the left-most book on the bottom shelf. In other words the input file indicates the volume numbers proceeding from left to right along the top shelf, and then from left to right along the bottom shelf.

Output format

Output one line to describe how one book is moved. The line consists of 'R' 'L' 'U' 'D' characters and one number (which is the volume number of the book picked up.) Do not place any spaces or punctuation on the line. Your output file will probably consist of a number of lines. The letters have the following meaning:

U	Crane moves up to top shelf
D	Crane moves down to bottom shelf
L	Crane moves left, the width of one book
R	Crane moves right, the width of one book

The D, U, L and R are used to describe the movements of the crane to position it at the book. The number is the volume number of the book that the crane picks up. D's, U's, R's and L's are used to describe to movement of the crane while it is transporting the book. The end of the line indicates that the crane stops moving and puts the book on the shelf at that position. This means that the crane must be positioned over an empty space when it has done all the movements in the line and gets to the end. The last line in the output file will not contain a number; it describes the movement of the crane from where it just placed at book, to the right most position on the bottom shelf.

Shelves

Scoring

Your program will score zero for a data set if:

- the crane makes an illegal move, examples of illegal moves are:
- U when already on the top shelf
- L when on the top shelf
- R when at the furthest right position on any shelf
- the books are not correctly sorted at the end
- a book is attempted to be placed on the shelf where another book already is
- a book is picked up from the wrong place (the program reports that volume 1 is being picked up, but the crane is positioned at book 4 for instance)

Your solution scores 100% multiplied by *optimal* divided by *your*, where *optimal* is the best known solution (using the least possible energy) and your is the amount of *energy* your solution required.

Triangle

Triangles

A piece of paper, which contains a large number of intersecting straight lines, will probably contain one or more triangles. Most people are surprised by how many triangles can be found in an ordinary picture. A *triangle* consists of three points, which are all connected to each other. The vertices of a triangle can be any points of intersection between two lines, not just end points.

Many points will be given to you in the input, but few of them will be connected. The formula provided below must be used to determine whether points are connected.

Task:

You are required to write a program, which accepts the points specified in an input file and outputs the triangles, which are present.

Calculating whether two points are connected:

The input file does not explicitly describe which points are connected to each other. It can be determined whether any two given points are connected by applying the following rule:

Let $(x_1 ; y_1)$ and $(x_2 ; y_2)$ be two points. They are only connected if and only if

$$|x_1 - x_2| + |y_1 - y_2|$$

is equal to 1 or a prime number. Recall that $|x| = -x$ when $x < 0$, and $|x| = x$ otherwise. The set of prime numbers starts with 2, 3, 5, 7, 11, 13, 17, 19…

Sample calculation

There are three points, and therefore 3 calculations to do:

	(20 ; –5) and (–8 ; –2)	(20 ; –5) and (0 ; 0)	(0 ; 0) and (–8 ; –2)
	| x1 – x2 | + | y1 – y2 |	| x1 – x2 | + | y1 – y2 |	| x1–x2 | + | y1–y2 |
=	| 20 – (–8) | + | (–5) – (–2)	| 20 – (0) | + | (–5) – 0 |	| 0 – (–8) | + | 0 – (–2) |
=	| 20 + 8 | + | –5 + 2 |	| 20 – 0 | + | –5 – 0 |	| 0 + 8 | + | 0 + 2 |
=	| 28 | + | –3 |	| 20 | + | –5 |	| 8 | + | 2 |
=	28 + 3	20 + 5	10
=	31	25	
is	Prime	not prime	not prime
	Connected	not connected	not connected

Constraints:

- All integers used in this question are signed numbers in the range –15000 to 15000.
- Triangles may not have zero area. For example the three points , 0 0, 1 0, 2 0 are all connected to each other (see the rule above) but they do not form a triangle since they are all in a straight line i.e.:

Is not a triangle

- There will be less than 1000 points in the input file.
- There will always be less than 10 000 triangles.

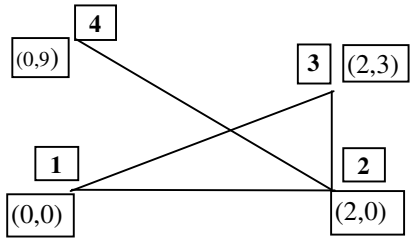
Input:

You will receive input in the following format:

- A single integer *N* indicating the number of points described in the input file, followed by a new line.
- The x and y co-ordinates are represented by 2 integers, separated by a space. Each pair of integers is on a separate line. The points can be considered to be numbered sequentially from 1 to *N*.
- The integers may contain a leading '+' or '–' sign to indicate positive or negative.

Sample input

TRI.DAT	**Explanation:**
4	4 points
0 0	Point 1 at (0,0)
2 0	Point 2 at (2,0)
2 3	Point 3 at (2,3)
0 9	Point 4 at (0,9)



Output:

- A single integer, which indicates the number of triangles found using the rules, followed by a new line.
- A number of lines of 3 pairs of integers (i.e. 6 integers). Each pair of integers represents the end points (numbered according to the sequence of co-ordinates in the input file) of the line that forms one side of the triangle.
- You may not place commas, brackets or any other punctuation in your output.

Sample output:

TRI.OUT	Explanation
3	Three triangles found
1 2 1 3 2 3	First triangle
1 2 1 3 2 4	Second triangle
1 3 2 3 2 4	Third triangle

Triangle

Scoring:

- The score is the number of correct triangles found expressed as a percentage of the total possible triangles.
- If your solution contains invalid triangles, then it will receive a score of 0.