DKC^3 2014 LONG COMPUTING PROBLEMS

1. Snake (75 pts)

Many people have played the snake game (a few years ago Nokia phones came with snake on it, sometimes when watching YouTube videos you could pause the video and press the left arrow to start the game…). Let's simulate that game! For this program, you will be given a matrix (15x15) of spaces and letters. The snake will initially consist of 3 contiguous 'X' characters, and the food pellets are represented by the 'F' character. The object of the game is to eat food pellets and grow, without hitting the boundary (stay within the size of the matrix). The snake will grow 1 size for each food pellet it eats.

For this program, you need to keep track of how the snake grows, how many pellets it eats, and whether the game ends or not. The game will end if either the snake goes outside the playing area OR it runs into itself.

The input will be a string of 20 of the characters "UDLRO" standing for Up, Down, Left, Right, and Onward…continue in the same direction. For each input, determine either how many food pellets are eaten, or if it is game over (by leaving the matrix). The game will restart with the same board configuration of snake and pellets. The snake will start to move to the RIGHT at the beginning of the game, and the snake will be horizontal.

Here is an example of the snake moving with the following input string:
UROOOUOOLOOOOOUOOOOO
12345678901234567890
The first 5 moves made are in bold below (12345). On the 8 the move, the snake would eat the pellet! Then, you would turn left and continue to eat the next pellet straight ahead (move 14).

```
0                          X
9                          X
8                          X
7                          X
6              --->        X
5F32109F                   54321098
        7                          7
        6                          6
    12345                      12345
 XXX    F                          F
```

Due to eating two pellets, the snake would be 5 X's long and be continuing upwards

Input

The first 15 lines consist of the matrix (15x15 characters). The next line will consist of the number of data sets. Each data set will consist of a line of 20 characters (UDLRO).

Output

For each data set, print out “N pellets” or “GAME OVER”. Then print out the final game board. Separate games with a blank line.

Input File: D:\DKC3\SnakeIn.txt
Output File: D:\DKC3\SnakeOut.txt

Examples:

Input:

```
*F*****FFF*****
***F********F**
F******F*******
****F******F***
********F*****F
****F*********F
F**XXX*********
***F**FF**F****
******FF****F**
***************
*F*F***F*******
*********F*****
***F***********
***************
******F*****F**
2
UROOOUOOLOOOOOUOOOOO
ODDRUROOOODOOOLOOOOU
```

Output:

```
GAME OVER
*F*X***FFF*****
***X********F**
F**XX**F*******
****F******F***
********F*****F
****F*********F
F**************
***F**FF**F****
******FF****F**
***************
*F*F***F*******
*********F*****
***F***********
***************
******F*****F**

8
*F*****FFF*****
***F********F**
F******F*******
****F******F***
********F*****F
****F*********F
F**************
***F********X**
***********X**
***********X**
*F*F***X****X**
*******XXXXXX**
***F***********
***************
******F*****F**
```

2. Traveling Speaker (75 pts)

Sally is a motivational speaker that travels around the country and gives presentations to employees at various companies. Each day she travels to a different city to give her presentation. Sally is afraid of flying, so she always drives herself, no matter how far the distance. Furthermore, she only takes toll roads, so she has to pay a fee to get on each road. Each day Sally is provided a map that contains the city she is currently in, the city she needs to drive to, and the other cities from the surrounding area. The map also contains the roads between each city, the time it takes to travel each road, and the cost of the toll for that road. Sally wants to take the shortest possible route from the current city to the destination city, but she only has a budget of $25 (2500 cents) per day for tolls. Find the shortest route that Sally can take without going over budget. If there is more than one shortest route, pick the one with the least cost in tolls. For simplicity, we will assume that all roads only go in one direction, and once Sally leaves a city, there is no way to get back to it (i.e. there is a directional flow on each road, and there are no cycles in the map).

Input

The input will consist of a line that contains the starting city and the destination city, separated by a comma. For simplicity the names of the cities will be represented by numbers ranging from 0 to N - 1, where N is the number of cities on the map. Then there will be a blank separation line, followed by a block of lines that represent the time on each road from each city to every other city on the map, comma separated. If there is no direct path from one city to another, that entry will be signified by -1. The times will be represented in minutes. Following this there will be another blank line, followed by another block of lines that represent the cost of each road between each city, comma separated. Again, if there is no direct path from one city to another, that entry will be signified by -1. The cost will be represented in cents. There will then be two blank lines between each test case. There are 10 test cases.

Output

The output should be the number that represents each city on the path Sally needs to take, separated by commas.

Input File: C:\DKC3\TravelIn.txt
Output File: C:\DKC3\TravelOut.txt

Examples:

Input

```
0,4

0,20,15,-1,50
-1,-1,-1,-1,40
-1,-1,-1,35,-1
-1,-1,-1,-1,20
-1,-1,-1,-1,-1

-1, 600, 300,-1,2600
-1,-1,-1,-1,1800
```

```
-1,-1,-1,400,-1
-1,-1,-1,-1,800
-1,-1,-1,-1,-1
```

Output

0,1,4

3. Mapping Mars (75 pts)

In 2015 the new Mars rover, Enterprise, makes a startling discovery. Using radar and other imaging technology the rover discovers what appears to have been a small outpost or settlement at the base of Olympus Mons. The site is buried several feet below the surface and is 1 kilometer square. NASA has launched 3 new archeological rovers to dig to the buried artifacts at the site. NASA has mapped out the site and divided it into a 10 by 10 grid, with each grid being 1 square hectometer. Since the rovers will have to remove several feet of Martian soil to get to the artifacts the maximum area of coverage of each rover will be 4 square hectometers. Based on the imaging from Enterprise each square hectometer has been assigned a number from 0 to 100 to indicate the level of interest of the scientists at NASA, with a 0 indicating no interest and 100 indicating "ludicrous" interest. Your job is to calculate the 3 grid points at which the 3 archeological rovers need to start their digging in order to maximize the amount of interest points covered. As NASA doesn't want to risk the rovers damaging each other during the dig the 4 square hectometers covered by each rover must not overlap.

Input
There will be 10 lines per test case. Each line will be a list of 10 numbers from 0 to 100 indicating the interest level for that grid square. The grid is laid out such that coordinate (0,0) is the lower left corner, and coordinate (9,9) is the upper right corner. The first coordinate is the column number and the second coordinate in the pair is the row. There will be no ties in level of interest between the three top areas.

Each test case will be separated by an asterisk.

Output
For each test case print out the three starting coordinates (lower left corner of the 4 square hectometer coverage) of the 3 rovers that will maximize the interest points covered. Order the coordinates for each test case by priority, highest level of interest to the lowest.

Input File: C:\DKC3\MappingIn.txt
Output File: C:\DKC3\MappingOut.txt

Examples:

Input:

```
100 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 40 0 0
0 0 10 0 0 0 0 0 10 0
0 0 0 20 0 0 0 50 0 0
0 0 0 0 0 0 0 0 0 0
0 0 5 0 0 0 0 60 0 0
0 0 0 0 0 0 50 0 0 0
0 0 10 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 5 0
0 0 0 0 2 0 0 0 0 0
*
100 0 0 0 0 0 0 0 100 0
0 0 0 100 0 0 0 40 0 0
0 0 10 0 0 0 0 0 10 0
0 0 0 20 0 0 0 50 0 0
0 0 0 0 0 0 0 0 0 0
```

0 0 5 0 0 0 0 60 0 0
0 0 0 0 0 0 50 0 0 0
0 0 10 0 0 100 0 0 0 0
0 0 0 0 0 0 0 0 5 0
0 0 0 0 2 0 0 0 0 0
*

Output:

(6,3) (0,8) (7,6)
(5,2) (7,8) (2,7)

4. Trip Less Traveled (75 pts)

Write a program that will read in destinations and the distance between them. The distance is the number of miles to go from one city to the other or vice-versa. Then calculate the order of travel to go from point A to point B in the shortest distance.

Input
Each line of input has two cities listed, along with the distance between them. The numbers are all whole numbers. The last line of each test case will list the starting point and ending point of the trip along with a distance of zero. Each test case is separated by a blank line.

Output
Output the order of travel along with the distance traveled for the shortest trip.

Input File: C:\DKC3\LessIn.txt
Output File: C:\DKC3\LessOut.txt

Examples:

Input:

Paris London 300
London Boston 1800
Boston Quebec 900
Dallas Houston 200
Paris Quebec 0

Moscow Madrid 2000
Paris Quebec 3000
Paris Madrid 800
Paris Moscow 3000
Paris Moscow 0

Output:

Paris London Boston Quebec 3000
Paris Madrid Moscow 2800