Junior Knights Assignments: File Input, List of Lists

Problem: Pizza Store Program (pizza.py)

This is the same problem as last week, but instead of the input being inputted by the user via question prompts, this time you'll read the input from a file, "pizza.txt".

In this problem, you'll ask the pizza store for the price of each of its pizzas. Then, you'll take several orders. For each order, you'll print out the total cost with tax and tip. Assume that tax and tip together amount to 20% of the order.

File Input Specification

Note: It is guaranteed that the file you read from will adhere to these specifications. This means that you do NOT have to check them!

1. The first line of the input file will contain a single positive integer, ***n*** *(**n** < 100)*, specifying the number of different pizza orders for Lazy Moon.

2. The next ***n*** lines will each contain one positive real number specifying the price for that corresponding pizza. (The first line will have the price for pizza 0, the next line for pizza 1, etc.)

3. The following line will contain a single positive integer ***k*** (***k*** < 50) representing the number of orders you have to evaluate.

4. The next ***k*** lines will contain information about each of the ***k*** orders with one order per line.

5. Each of these lines will contain ***n*** non-negative integers, representing how many of those pizzas, respectively are in the order. (For example, if n=5 and the line contains the values 0 0 6 0 9, then the order contains 6 slices of pizza #2 and 9 slices of pizza #4.)

Output Specification

For each of the ***k*** test cases, output a line with the following format:

```
On day X, you will spend $Y at Lazy Moon.
```

where X is the test case number (starting with 1), and Y is the price of the pizza in dollars specified as a floating point number.

Sample Input File (pizza.txt)

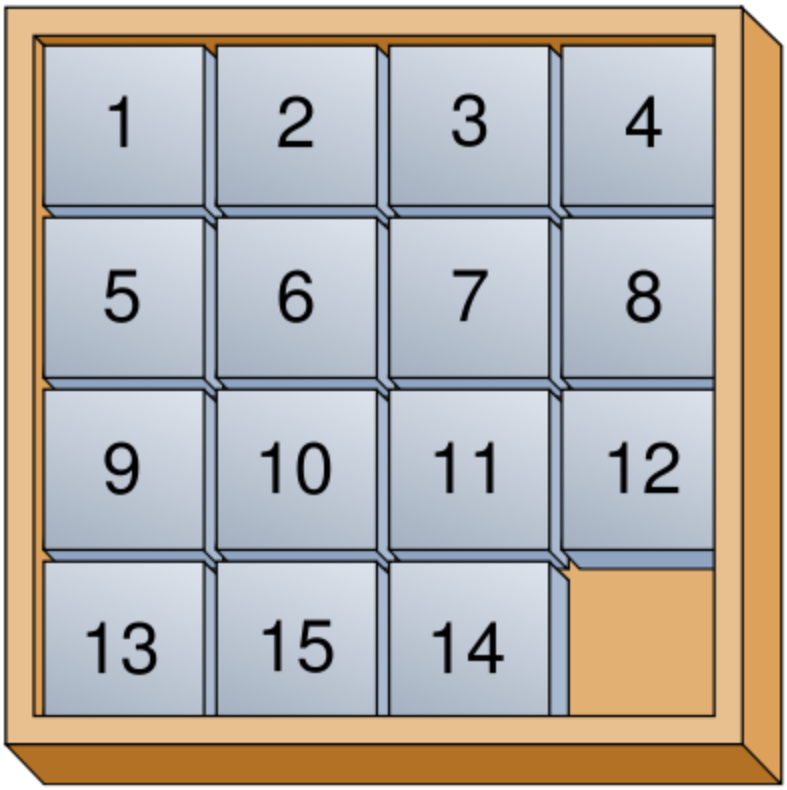
```
5
3.00
3.50
4.50
5.00
6.00
3
1 1 1 1 1
0 0 2 1 6
1 0 3 2 3
```

Sample Output (Corresponding to Sample Input File)

```
On day 1, you will spend $26.4 at Lazy Moon.
On day 2, you will spend $60.0 at Lazy Moon.
On day 3, you will spend $53.4 at Lazy Moon.
```

Problem: 15 Puzzle

This is a common puzzle with a 4x4 playing space with 15 tiles, numbered 1 through 15. One "spot" is always left blank. Here is an example of the puzzle:



The goal is to get the tiles in order, 1 through 15, from left to right, top to bottom, by just sliding tiles into the empty square. In this configuration, the goal would be to get the 14 and 15 to switch places, without affecting any of the other squares.

Your goal will be to write a program that allows the user to play this game.

Input File Specification

The first part of your program will read in possible puzzle configurations from a file and choose one of them randomly. The file format is as follows:

The first line of the input file will contain a single positive integer *n*, representing the number of puzzles in the file. The puzzles will be contained in the next 5*n* lines. In particular, each puzzle will be stored in 5 lines. The first line will contain the first row of values in the puzzle separated by spaces. The second line will contain the second row, the third line, the third row and the fourth line, the fourth row. The blank spot will be designated by the integer 0. The last line (fifth) will be a blank line.

Sample Input File

```
3
1 0 2 3
5 6 7 4
9 10 11 8
13 14 15 12

15 1 2 3
14 13 12 11
10 9 8 7
6 0 5 4

1 2 3 4
5 6 0 8
9 10 7 11
13 14 15 12
```

Output Specification

At the very beginning of the program, you will prompt the user to enter in the name of the file. Then, the program will open the file and load a puzzle into its memory. Once this is done, the puzzle should be displayed to the user and the user should choose a tile to move. Roughly the board should print out as follows:

```
1     2          3
5     6     7    4
9     10    11   8
13    14    15   12
```

Note that an underscore is to be used to denote the blank square and that internally, this is stored as 0.

Prompt the user with the following question after showing them the board:

```
Which piece would you like to slide into the open slot?
Note, answering 0 means you quit the game without winning.
```

If the user chooses a valid square, process the move and print out the board again. If they do not, print out the following message:

```
Sorry, that is not a valid square to slide into the open slot.
No move has been executed.
```

If the user chooses 0, print out:

```
Sorry, looks like you gave up on the puzzle.

1. Load a new puzzle.
2. Quit
```

Output Sample

Two samples are provided in the files puzzle15-1.out and puzzle15-2.out