Problem A1
Star Wars Logo

Input: none
Output: monitor

Description: ASCII art is a graphic design technique that creates pictures using the 95 printable ASCII characters. Use ASCII art to display the classic Stars Wars logo

Input: No input.

Output: ASCII representation of the Star Wars logo exactly as shown.

Sample Input: None.

Sample Output:

```
     8888888888  888    88888
   88      88   88 88   88  88
    8888  88  88   88  88888
       88 88 888888888 88   88
88888888  88 88     88 88    888888

88  88  88   888    88888    888888
88  88  88  88 88   88  88  88
88 8888 88 88   88  88888    8888
 888  888 888888888 88  88      88
  88  88  88     88 88   88888888
```

Problem A2
Garbled Message

Input: a2.in
Output: monitor

Description: Obi-Wan has just sent an urgent message to the Jedi Temple, but the message has been garbled by subspace interference. All of the strings “th” have been replace with “iiing”. Your job is to unscramble the message

Input: One or more lines of garbled text.

Output: The ungarbled text.

Sample Input:

```
Dariiing Bane restructured iiinge cult, so iiingat iiingere
could only be two - no more, no less - a master, and an
apprentice. Bane adopted cunning, subterfuge, and stealiiing as
iiinge fundamental tenets of iiinge Siiiing order. Bane took an
apprentice. When iiingat apprentice succeeded him, iiingat new
Siiiing Lord would take an apprentice.
```

Sample Output:

```
Darth Bane restructured the cult, so that there
could only be two - no more, no less - a master, and an
apprentice. Bane adopted cunning, subterfuge, and stealth as
the fundamental tenets of the Sith order. Bane took an
apprentice. When that apprentice succeeded him, that new
Sith Lord would take an apprentice.
```

Problem A3
Yoda Talk

Input: a3.in
Output: monitor

Description: As a young jedi night you must learn to converse with Yoda. You have found a simple rule that helps change a “normal” sentence into “Yoda talk”. Take the first to words in the sentence and place them at the end. Write a program that uses this rule to change normal sentence into “Yoda talk”.

Input: The first line of input will be a positive integer, n. This will be followed by n lines each contain a “normal” sentence that you must change into “Yoda talk”.

Output: For each normal sentence print (in the order given) the corresponding sentence in Yoda talk.

Sample Input:
4
I will go now to find the Wookiee
Solo found the death star near planet Kessel
I’ll fight Darth Maul here and now
Vader will find Luck before he can escape

Sample Output:
go now to find the Wookiee I will
the death star near planet Kessel Solo found
Darth Maul here and now I’ll fight
find Luck before he can escape Vader will

Problem A4
Decoder

Input: a4.n
Output: monitor

Description: All the Sith messages are sent using a complex coding scheme. You have cracked their code and must write a program to decode coded messages. Their code works as follows. Each word in the coded message represents one letter in the decoded message. Use the first letter of the first word and for each subsequent word use the nth letter where n is the length of the previous word. If the previous word is longer than the current word the current word represents a "space" (i.e., a blank space).

Here are two example:

Coded Message	**Decode Message**
`A Tree`	`AT`
`Some one too a sas`	`S o s`

Input: A positive integer, n, on the first line. After the first line there will be n words that represent the coded message.

Output: The decoded message on one line.

Sample Input:
```
10
Give Solo a total Naboo map DEV reject bateau German
```

Sample Output:
```
Go to Vjun
```

Problem A5
Making Change

Input: a5.in
Output: monitor

Description: Anakin Skywalker is leaving Tatoonie, but before he can go he must program a droid to do his old job of making change for customers at Watto's junk business. Given the price of an item and the payment determine the correct change using $50, $20, $10, $5, and $1 bills. You don't have to worry about change, because Watto always keeps the change.

Input: A positive integer, n, on the first line indicating the number of data sets to follow. Each data set will consist of two positive numbers. The first is the cost to the customer and the second is the payment made by the customer. The payment will always be greater than the cost.

Output: For each data set print the correct change on one line using the format

```
v-$50, w-$20, x-$10, y-$5, z-$1
```

where v, w, x, y, z are the number of bills needed to make the correct change. Of course, the correct change will always use larger bills when possible (i.e., 1-$5 not 5-$1)

Sample Input:

```
3
1894.77 2000.00
77.15 100.40
25.50 100.00
```

Sample Output:

```
2-$50, 0-$20, 0-$10, 1-$5, 0-$1
0-$50, 1-$20, 0-$10, 0-$5, 3-$1
1-$50, 1-$20, 0-$10, 0-$5, 4-$1
```

Problem A6
Target Practice

Input: a6.in
Output: monitor

Description: The Trade Federation is too cheap to purchase practice targets for testing the battle droid weapon system. Your job is to write a program to print targets given an input that specifies the size of the target. The basic shape of a target is a square with an X through the middle. The following is a target of size 7.

```
|-----|        The middle characters are dashes (i.e., minus sign)
|*   *|
| * * |
|  *  |
| * * |
|*   *|
|-----|
```

Input: An odd positive integer greater than 1.

Output: Target of the appropriate size.

Sample Input:

```
9
```

Sample Output:

```
|-------|
|*     *|
| *   * |
|  * *  |
|   *   |
|  * *  |
| *   * |
|*     *|
|-------|
```

Problem A7
Escape Route

Input: a7.in
Output: monitor

Description: You are surrounded by Imperial warships and must land on a planet to escape. Your best chance is to find a route to the closest planet. Unfortunately the navigation system has been damaged. Your job is to write a program that will find the planet that is closest to your current location. The problem will be described using a mxm grid containing the characters

s	your ship
w	imperial warship
p	planet
-	open space

The two dimensional location of an object (i.e., your ship, imperial warship, or planet) corresponds to its location in the grid. For example given a 4x4 grid the locations are specified by the following table.

(0,0)	(0,1)	(0,2)	(0,3)
(1,0)	(1,1)	(1,2)	(1,3)
(2,0)	(2,1)	(2,2)	(2,3)
(3,0)	(3,1)	(3,2)	(3,3)

Recall that the distance between two points (x_1, y_1) and (x_2, y_2) is given by the formula

$$\text{dist} = \left((x_1 - x_2)^2 + (y_1 - y_2)^2 \right)^{1/2}$$

Input: The first line will have a positive integer, n, specifying the number of data sets to follow. Each data set will have a positive integer, m, on the first line followed by m lines and each line will have exactly m characters. The characters will be either s, w, p, or -.

Output: For each data set print one line with the coordinates of your current location followed by a colon, coordinates of a closest planet followed by a colon, and the distance to that planet formatted to two decimal places.

Sample Input:

```
2
4
-wp-
s--w
-w-p
-pw-
5
s--wp
w--pw
-w-p-
p--wp
w--pw
```

Sample Output:

```
(1,0):(0,2):2.24
(0,0):(3,0):3.00
```

Problem A8
Asteroid Field

Input: a8.in
Output: monitor

Description: Plot a path through the asteroid field. Given a starting location, final destination, and a description of the asteroid fields plot a **shortest** path that takes you from the starting location to the final destination without running into any asteroids. The asteroid field is described using a mxm grid of characters with

s for starting location
d for final location
- for open space
* asteroid

Here is an example of a 4x4 grid.

```
s*-*
-*-*
----
*-*d
```

Your ship can move up, down, left, and right (not diagonally). Each position in a mxm grid will be assigned an integer between 0 and m^2-1 as follows.

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

Input: The first line will have a positive integer n representing the number of data sets. The first line of each data set will contain an integer m, followed by m lines, and each line will contain m characters. The character s will always be in the top left corner and d will always be in the bottom right corner.

Output: For each data set print the minimal number of moves needed to reach the destination or -1 if there is no solution

Sample Input:

```
2
4
s*-*
-*-*
--*-
*-*d
6
s*---*
-*-*--
---**-
***---
--*-**
*-*--d
```

Sample Output:

```
-1
18
```

Problem A9
Which Way

Input: a9.n
Output: monitor

Description: You are trapped in Jabba's Palace. You have a coded map that describes the way out from your current location. The map contains a sequence of positive integers. Each integer corresponds to one of three directions (left, straight, right). To determine the direction you must convert the number into binary (base two with no leading 0s). If the binary number has more 0's than 1's go left. If the binary number has the same number of 0's and 1's go straight, and if it has more 1's than 0's go right. Your job is to read the sequence of positive integers and print the directions to find your way out.

Here are three example of converting positive (decimal) integers into binary and then into a direction.

Decimal	Binary	Direction
17	10001	left
9	1001	straight
22	10110	right

Input: A sequence of positive integers, one per line

Output: The correct directions (left, straight, right) for escaping the Jabba's Palace. You should write each move (left, straight, right) on a separate line with no extra lines.

Sample Input:
```
17
7
4
9
22
```

Sample Output:
```
left
right
left
straight
right
```

Problem A10
Let the Wookiee Win

Input: a10.in
Output: monitor

Description: You are playing Chewbacca in a tough game of Glactic tic-tac-toe, which is just like simple tic-tac-toe except the playing board is a 5x5 and the goal is to get 4 in a row. Han Solo warns you that Wookiee's have been known to pull people's arms out of their sockets when they lose. Wisely, you decide to let the Wookiee win!

Input:
Input consists of a series of 5x5 playing boards, each separated by a blank line. Each square in a row is separated by a single space. You are 'O', Chewie is 'X' and empty squares are denoted by '*'. For each board there is exactly one empty square that you can play that does not give you a win and does not block any of Chewie's winning moves on his next turn. The last board is followed immediately by the word "Finished" on the next line.

Output: For each board, print a line consisting of the number of the square that you should play to avoid having your arms pulled out of their sockets. The numbering should conform to the table below.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Sample Input:
```
X O * X X
O O X X *
O O * X X
O X O O X
X O O X O

X X X * O
O * * O X
O O X * X
O * X X O
* X O O O
Finished
```

Sample Output:
```
3
17
```

Problem A11
The Gauntlet

Input: a11.in
Output: monitor

Description: As part of your Jedi training you must run the gauntlet collecting as many points as possible. You are awarded points for each sector of the gauntlet you pass through. The number of points awarded depends on the difficulty of the sector. You enter the top left corner of the gauntlet and exit at the bottom right corner. You must do this as quickly as possible, always moving forward (right or down) and never backing up (up or left). The gauntlet playing field is a NXM grid with each sector of the grid worth a fixed number of points. You will start in the top right corner of the playing field and find a path through the gauntlet the gives you a maximum number of points. Here is an example of a 3x4 that has a best score of 37.

5	8	10	4
6	2	7	5
9	11	3	2

Input: The first line will have a positive integer n representing the number of data sets. The first line of each data set will contain two positive integers, N and M, followed by N lines, and each line will contain M non negative integers.

Output: For each data set print on one line the best possible score the Jedi could have when running the gauntlet.

Sample Input:

```
2
3 4
5 8 10 4
6 2 7  5
9 11 3 2
4 4
3 7  3 9
6 10 7 8
5 7  7 5
9 9  5 9
```

Sample Output:

```
37
50
```