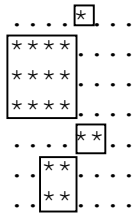
1. Blob Count

Program Name: BlobCount.java Input File: blobcount.dat

Johnny is studying different shapes in a plane. For this particular study, he refers to the shapes as blobs even though they are solid rectangles. He represents his blobs in a rectangular grid as a collection of one or more contiguous asterisks (`*`). Contiguous means that the asterisks must be adjacent either horizontally or vertically. Non-blob characters are represented by periods (`.`). In the diagram below, there are 4 blobs.



Johnny knows the location of the uppermost, leftmost corner of a blob. You are to write a program that will determine the number of characters in the blob. The largest blob in the example above has its uppermost, leftmost corner at row 2, column 1 or `2 1`.

Input

The first line of input will contain a single integer `n` that indicates the number of data sets to follow. For each data set:

- the first line will contain three integers in the form `r c s` which meet the following criteria:
  - $r \geq 3$ is the number of rows in the grid
  - $c \geq 3$ is the number of columns in the grid
  - $s > 1$ is the number of test cases for that grid
- the next `r` lines will contain the grid.
- the next `s` lines will each contain an ordered pair `x y`, $1 \leq x \leq r$ and $1 \leq y \leq c$, which is the location of a character in the grid that is either the uppermost, leftmost character in a blob or not in a blob at all.

Output

For each test case, you will print the number of characters in the blob. If the test case falls on a cell that is not part of a blob, print `NO BLOB`.

Example Input File

```
2
7 8 2
....*...
****....
****....
****....
....**..
..**....
..**....
2 1
5 3
5 6 3
...***
***...
...***
**.***
**.***
3 4
4 1
1 4
```

1. Blob Count (cont.)

Example Output to Screen

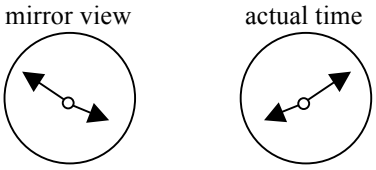
```
12
NO BLOB
9
4
3
```

2. Clock in the Mirror

Program Name: Clock.java Input File: clock.dat

Jeremy likes to lie in bed to watch TV. He knows he has to turn the TV off at a certain time. The only way he can see his analog clock is by looking into a mirror. The numbers on the clock are small so he cannot read them. He has to mentally "flip" the clock to tell the correct time. He has been curious about how the time looks in the mirror compared to the actual time.

For example, if in the mirror, the small the clock looks like it is 3:48, then the actual time would be 8:12.



You are to write a program that, given the time in the mirror, will find the actual time for him.

Input
The first line of input will contain a single integer `n` that indicates the number of mirror times to follow. Each of the following `n` lines will contain a time in the form `hh:mm` as seen in the mirror (leading zeros in hour time will be omitted).

Output
For each mirror time input, print the actual time in the form `hh:mm` with no leading zeros for hours.

Example Input File
```
5
3:48
8:27
10:30
1:23
5:00
```

Example Output to Screen
```
8:12
3:33
1:30
10:37
7:00
```

3. Digi-count

Program Name: Digicount.java Input File: digicount.dat

Mr. Appleworth was tired of one of his students, Alex, being disruptive in class. In an effort to change Alex's behavior, Mr. Appleworth decided to have Alex write all of the integers beginning with 1 and continuing to some number x, $x \leq 100$. The number x was chosen by the teacher and was based on the severity of the disruption.

After having done this exercise several times, Alex started wondering how many times he had written each of the digits, 0 through 9. Rather than count them, he decided that it would be easier if he had a program to give him that information. Being Alex's best friend, you have agreed to write the program for him.

Given the number x, the number that Alex has to write through, you will write a program that will print the number of zeros, ones, twos, threes, fours, fives, sixes, sevens, eights, and nines that Alex will write.

Input
The only line of input will contain an unknown number of positive integers, each indicating the last number that Alex is assigned to write for each of his disruptions. The integers will be separated by a space.

Output
For each disruption, on a single line and followed by a space, you will print the number of times each of the digits will be written beginning with the digit zero and continuing through the digit nine.

Example Input File
```
4 12 35 19
```

Example Output to Screen
```
0 1 1 1 1 0 0 0 0 0
1 5 2 1 1 1 1 1 1 1
3 14 14 10 4 4 3 3 3 3
1 12 2 2 2 2 2 2 2 2
```

4. Jack Be Nimble

Program Name: Jack.java Input File: jack.dat

Write a program that will read a data file and print every other line beginning with the second line.

Input
The input file will contain an unknown number of lines of text.

Output
You will output every other line of the data file beginning with the first line.

.Example Input File
```
JACK BE NIMBLE.
JACK BE QUICK.
JACK JUMP OVER
THE CANDLESTICK.
```

Example Output to Screen
```
JACK BE QUICK.
THE CANDLESTICK.
```

5. Maze Solver

Program Name: Maze.java Input File: maze.dat

The local newspaper wants to start running daily mazes for children. They contracted a software development firm that wrote software that can randomly generate mazes. Unfortunately the software is buggy and not every maze is functional. What is even worse is that the company has now gone under! To save money, the editor has contacted you to write a program that can determine if a maze is solvable or not. A maze is solvable if you can get from the starting position to the end position by moving horizontally and vertically (no diagonal movement) though the maze.

Input

The first line of input will contain a single integer `n` that indicates the number of mazes to follow. Each maze is a 20 by 20 matrix, occupying 20 characters per line for 20 consecutive lines. Each maze is guaranteed to have one beginning spot, denoted by a capital `S`, and one ending point, denoted by a capital `E`. Wall characters are denoted by pound signs (`#`) and pathable area characters are denoted by periods (`.`). There is exactly one blank line separating matrices.

Output

For each maze you should print "`Maze #N:` " where `N` is the number of the current maze, followed by a space and then by `YES` if the maze is solvable or `NO` if the maze is not solvable.

Example Input File

```
2
#S##################
#.....####.....####
####..#############
####.#########.####
####.#########.####
####...#######.####
######.#####...####
#####..###..#######
#.....####....#####
##.##.#############
##.##........######
##.#########.######
##....######.#...##
#####.##..##.######
#####.#####...######
#####.####.########
##................##
################.##
###..............##
###E################
```

(continued on next page)

5. Maze Solver (cont.)

```
####################
#.....####.....####
####S.#############
####.#########.####
####.#########.####
####...#######.####
######.#####...####
#####..####....####
#.....###........##
#####.#############
#####........######
##......####.######
############.######
########..##.#...##
##########..#######
##########.########
##................##
####################
###..............##
###E###############
```

Example Output to Screen

```
Maze #1: YES
Maze #2: NO
```

6. Octagon Means Stop

Program Name: Octagon.java Input File: octagon.dat

Nancy is making a series of posters to encourage students to stay away from drugs. She needs octagons of different sizes so she can put anti-drug information inside the octagon.

Input
The only line of input will contain an unknown number of positive integers in the range [5,15]. The integers will be separated by a single space.

Output
For each integer `n` input, you will print a regular octagon with each side of the octagon having `n x`'s as shown below. Print a blank line after each octagon.

Example Input File
```
6 7
```

Example Output to Screen
```
      xxxxxx
     x      x
    x        x
   x          x
  x            x
 x              x
x                x
x                x
x                x
x                x
x                x
x                x
 x              x
  x            x
   x          x
    x        x
     x      x
      xxxxxx

       xxxxxxx
      x       x
     x         x
    x           x
   x             x
  x               x
 x                 x
x                   x
x                   x
x                   x
x                   x
x                   x
x                   x
x                   x
 x                 x
  x               x
   x             x
    x           x
     x         x
      x       x
       xxxxxxx
```

7. Pal Palindromes

Program Name: Pals.java Input File: pals.dat

There are many numbers that are palindromes, numbers whose digits read the same backwards and forwards. For example, 121, 363, 444, 989, and 5 are all palindromes. Also, if a number `j` is not a multiple of 10, there are an infinite number of multiples of `j` which are also palindromes. I call these numbers Pal Palindromes. For example, if `j` is 3, then 6, 9, 141, and 222 are some of Pal Palindromes that are multiples of `j`.

You are to write a program that, given a positive integer `m`, will determine how many of `m`'s Pal Palindromes contain a specific number of digits.

Input
The first line of input will contain a single integer `n` that indicates the number of test cases to follow. Each of the following `n` lines will contain two positive integers in the format `m d`, where $m \leq 50$ is the number for which you are to find its Pal Palindromes and $d \leq 7$ is the number of digits in the Pal Palindrome.

Output
For each test case, you will print a line containing the number of `m`'s Pal Palindromes that contain `d` digits.

Example Input File
```
4
5 2
7 3
20 5
12 4
```

Example Output to Screen
```
1
12
0
7
```

8. Post Office

Program Name: PostOffice.java Input File: postoffice.dat

Mark has a business that ships widgets to cities all over the country. If the box he ships weighs 5 pounds or more, the cost of shipping increases significantly. The boxes he uses to ship his widgets weigh 0.4 pounds each and the widgets vary in weight. You are to write a program to help him with his shipping by determining if the package containing the widgets listed is overweight or not.

Input

The first line of input will contain a single integer `n` that indicates the number of boxes he needs to ship. Each of the following `n` lines will contain a list of weights of the widgets for that particular box. Each of the weights will be in pounds, rounded to hundredths and separated by a space.

Output

For each package, and on a separate line, print `OK` if the package weighs less than 5 pounds or print `OVERWEIGHT` if the package weighs 5 pounds or more.

Example Input File

```
3
1.21 0.33 1.42 0.90 0.45 1.30 0.30
1.20 0.59 1.23 0.99
0.62 0.61 0.63 0.64 0.65 0.66 0.67 0.68 0.69
```

Example Output to Screen

```
OVERWEIGHT
OK
OVERWEIGHT
```

9. Rattle

Program Name: Rattle.java Input File: rattle.dat

Kim and Pat like to text each other but are afraid that their parents might read their text messages. They have decided to write in code so their parents would not know what they are saying. They have developed a method for encoding their messages. The process for their code is:

- Put the message, including spaces, in the smallest square matrix that will hold the message. Fill cells in each row, beginning with column one, from left to right before moving to the next row. Columns are numbered beginning with column 1 as shown below.
- If there are unused cells at the end of the matrix, the first cell will be filled with an asterisk (*) and the remaining unused cells will be filled with consecutive letters of the alphabet beginning with the letter `A` and continuing until all empty cells have been filled.
- "Rattle" the matrix to encode the message as follows:
  - odd numbered columns rotate each letter down one cell with the last cell becoming the first.
  - even numbered columns rotate each letter up one cell with the first cell becoming the last.
- Rewrite the coded message in the new order by rows
- Send the coded message.

For example, the message: `I love Computer Science`

would fit into the 5x5 matrix at the left below and after the "Rattle" the matrix would look like the matrix on the right below.

1	2	3	4	5
I		L	O	V
E		C	O	M
P	U	T	E	R
	S	C	I	E
N	C	E	*	A

1	2	3	4	5
N		E	O	A
I	U	L	E	V
E	S	C	I	M
P	C	T	*	R
		C	O	E

The encoded message sent would be: `N EOAIULEVESCIMPCT*R  COE`

You are to write a program that will decode the encoded message.

Input
The first line of input will contain a single integer `n` that indicates the number of encoded messages to be sent. Each of the following `n` lines will contain a single encoded message less than 625 characters long.

Output
For each encoded message, you will print the decoded message. Do not include the characters used to fill the matrix.

Example Input File
```
2
N EOAIULEVESCIMPCT*R  COE
EIG IIKTWINTIIG UHL DISSR CT PSTTWERF RTTTS*EB DOFAHDJSLAIENANC
```

Note: There is an intentional space at the end of data case 2

Example Output to Screen
```
I LOVE COMPUTER SCIENCE
WINNING UIL DISTRICT IS THE FIRST STEP TOWARD STATE
```

10. Spring Trip

Program Name: Spring.java Input File: spring.dat

The band is selling cookie dough to raise money for their spring trip to the coast. Cookie dough is sold in packages for $14 each and in tubs for $15 each.

Each member must raise $500 for his portion of the trip and can get credits toward raising his $500 as follows:

- For each for each package of cookie dough that a member sells, he gets a credit of 48% of the money collected toward his share of the trip.
- For each tub of cookie dough that a member sells, he gets a credit of 45% of the money collected toward his share of the trip.
- For any donation that a member receives, he gets a credit of 100% of the money donated toward his share of the trip.

You are to write a program that will compute the amount of money that each member has raised.

Input

The first line of input will contain a single integer `n` that indicates the number of students in the band. For each student:

- The first line will contain the member's first name and a space followed by single integer `t` ≥ 0 that indicates the number of transactions that that member has.
- Each of the following `t` lines will contain one transaction consisting of up to three codes that will identify the sales for that transaction.
    - The codes will indicate the type of item and how many of that item the member sold or the amount of a donation received.
        - The code types are a `P` (a package of cookie dough), a `T` (tub of cookie dough), or a `D` (a donation)
        - Each type will be followed immediately (no space) by how many of the items he sold or the integer amount of the donation he received.
    - The codes will be separated by a space.

Output

For each student, and in the order the students appear in the data file, you will output the student's name and a space followed by the amount of money that he has left to raise. If he has raised more than $500, you will print his name, a space and `OVER` followed by the amount of extra money he raised. The money output should be preceded by a `$` and rounded to the nearest penny. Do not round until all mathematical operations are complete.

Example Input File

```
3
STEVE 4
T3 P5 D100
T4 D25
P3 T4
D100
MARY 3
D300 T3 P5
T8 P8
P10
JOE 5
P20 D10 T25
T5 P10 D15
D25
P20
D100
```

10. Spring Trip (cont.)

Example Output to Screen

```
STEVE $146.99
MARY OVER $28.81
JOE OVER $188.50
```

11. Values

Program Name: Values.java Input File: values.dat

The value of a word or phrase can be determined by finding the sum of the value of each of its letters. The value of a letter is determined by the following process:

A=1; B=2; C=3; ... ; Z=26; space = 0

You are to write a program to find the value of each word or phrase and print all of the words or phrases in order by its value from lowest to highest. In case there is more than one word or phrase with the same value, they should be printed in alphabetical order.

Input

The first line of input will contain a single integer `n` that indicates the number of words or phrases to follow. Each of the following `n` lines will contain a word or phrase. All words and phrases will be composed only of uppercase letters of the alphabet and spaces.

Output

In order by its value from lowest to highest, print the value followed by a space and the word or phrase. If there is more than one word or phrase with the same value, they should be printed in alphabetical order.

Example Input File

```
5
HARD WORK
QUIT
SCHOOL
KNOWLEDGE IS POWER
HOMEWORK
```

Example Output to Screen

```
67 QUIT
72 SCHOOL
98 HARD WORK
108 HOMEWORK
201 KNOWLEDGE IS POWER
```

12. XML Checker

Program Name: XML.java Input File: xml.dat

XML is an extension of HTML. In XML, the user can introduce a tag set that describes the data that he is working with. There are various rules that determine a valid XML document. The program that you will write will check two of those rules.

Rule I: A beginning tag must have an ending tag. A beginning tag is represented by <*tag_name*> and the corresponding ending tag is represented by <*/tag_name*>.

Rule II: An element defined by one pair of beginning and ending tags may nest completely inside another pair of beginning and ending tags but they may not overlap.

Valid XML:
<address> <street> 123 Any Street </street> <city> Any City </city> </address>

Invalid XML:

<address> <street> 123 Any Street </street> <city> Any City </address> </city>

Your program will determine if a snippet of XML code conforms to these two rules or not.

Input
The first line of input will contain a single integer `n` that indicates the number of data sets to follow. For each data set, the first line will contain a single integer `m` that indicates the number of lines that follow in that snippet of XML code.

Output
For each snippet of XML code you will write `valid` or `invalid` on a line by itself.

Example Input
```
2
4
<student>
<name>
John Doe
</student>
2
<gpa>
</gpa>
```

Example Output to Screen
```
invalid
valid
```