Problem A
Which Base is it Anyway?

Programming languages such as C++ and Java can prefix characters to denote the base of constant integer values. For example, hexadecimal (base 16) constants are preceded by the string "`0x`". Octal (base 8) values are preceded by the character "`0`" (zero). Decimal (base 10) values do not have a prefix. For example, all the following represent the same integer constant, albeit in different bases.

```
0x1234
011064
 4660
```

The prefix makes it clear to the compiler what base the value is in. Without the "`0x`" prefix, for example, it would be impossible for the compiler to determine if 1234 was hexadecimal. It could be octal or decimal.

For this problem, you will write a program that interprets a string of decimal digits as if it were an octal value, a decimal value or a hexadecimal value.

Input

The first line of input contains a single decimal integer P, ($1 \leq P \leq 100$), which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of a single line of input. It contains the data set number, K, followed by a single space, followed by a string of at most 7 decimal digits.

Output

For each data set there is one line of output. The single output line consists of the data set number, K, followed by a space followed by 3 space separated decimal integers which are the value of the input as if it were interpreted to as octal, decimal and hexadecimal respectively. If the input value cannot be interpreted as an octal value, use the value 0.

Sample Input 1	Sample Output 1
4 1 1234 2 9 3 1777 4 129	1 668 1234 4660 2 0 9 9 3 1023 1777 6007 4 0 129 297

This page is intentionally left blank.

Problem B
FBI Universal Control Numbers

The **FBI** has recently changed its *Universal Control Numbers (UCN)* for identifying individuals who are in the FBI's fingerprint database to an eight digit base 27 value with a ninth check digit. The digits used are:

`0123456789ACDEFHJKLMNPRTVWX`

Some letters are not used because of possible confusion with other digits:

`B->8, G->C, I->1, O->0, Q->0, S->5, U->V, Y->V, Z->2`

The check digit is computed as:

`(2*D1 + 4*D2 + 5*D3 + 7*D4 + 8*D5 + 10*D6 + 11*D7 + 13*D8) mod 27`

Where `Dn` is the n^{th} digit from the left.

This choice of *check* digit detects any single digit error and any error transposing an adjacent pair of the original eight digits.

For this problem, you will write a program to parse a *UCN* input by a user. Your program should accept decimal digits and *any* capital letter as digits. If any of the *confusing* letters appear in the input, you should replace them with the corresponding valid digit as listed above. Your program should compute the correct *check* digit and compare it to the entered check digit. The input is rejected if they do not match otherwise the decimal (base 10) value corresponding to the first eight digits is returned.

Input

The first line of input contains a single decimal integer P, $(1 \leq P \leq 10\,000)$, which is the number of data sets that follow. Each data set should be processed identically and independently. Each data set consists of a single line of input. It contains the data set number, K, followed by a single space, followed by 9 decimal digits or capital (alphabetic) characters.

Output

For each data set there is one line of output. The single output line consists of the data set number, K, followed by a single space followed by the string "`Invalid`" (without the quotes) or the decimal value corresponding to the first eight digits.

Sample Input 1	Sample Output 1
3 1 12345678A 2 12435678A 3 12355678A	1 11280469652 2 Invalid 3 Invalid

This page is intentionally left blank.

Problem C
m-ary Partitions

A partition of an integer n is a set of positive integers which sum to n, typically written in descending order. For example:

```
10 = 4+3+2+1
```

A partition is m-ary if each term in the partition is a power of m. For example, the 3-ary partitions of 9 are:

```
9
3+3+3
3+3+1+1+1
3+1+1+1+1+1+1
1+1+1+1+1+1+1+1+1
```

Write a program to find the number of m-ary partitions of an integer n.

Input

The first line of input contains a single decimal integer P, $(1 \leq P \leq 1\,000)$, which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of a single line of input. The line contains the data set number, K, followed by the base of powers, m, $(3 \leq m \leq 100)$, followed by a space, followed by the integer, n, $(3 \leq n \leq 10\,000)$, for which the number of m-ary partitions is to be found.

Output

For each data set there is one line of output. The output line contains the data set number, K, a space, and the number of m-ary partitions of n. The result should fit in a 32-bit unsigned integer.

Sample Input 1	Sample Output 1
5 1 3 9 2 3 47 3 5 123 4 7 4321 5 97 9999	1 5 2 63 3 75 4 144236 5 111

This page is intentionally left blank.

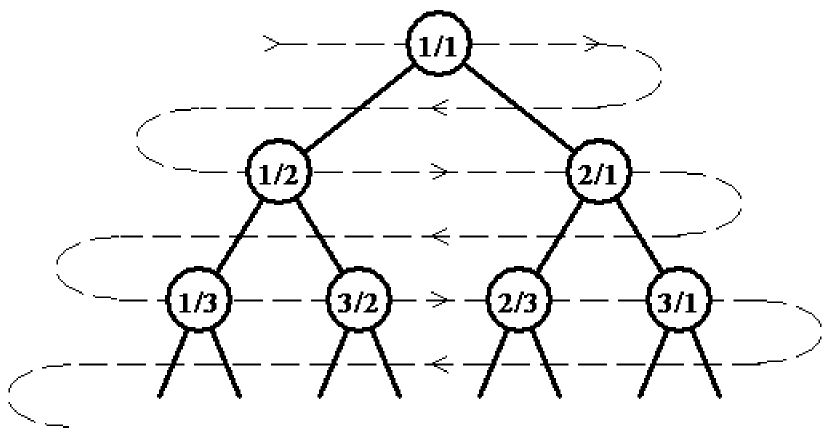
Problem D
A Rational Sequence (Take 3)

A sequence of positive rational numbers is defined as follows:

An infinite full binary tree labeled by positive rational numbers is defined by:

- The label of the root is $1/1$.
- The left child of label p/q is $p/(p+q)$.
- The right child of label p/q is $(p+q)/q$.

The top of the tree is shown in the following figure:



The sequence is defined by doing a level order (breadth first) traversal of the tree (indicated by the light dashed line). So that:

$$F(1) = 1/1, F(2) = 1/2, F(3) = 2/1, F(4) = 1/3, F(5) = 3/2, F(6) = 2/3, \ldots$$

Write a program to compute the n^{th} element of the sequence, $F(n)$. Does this problem sound familiar? Well it should! We had variations of this problem at the 2014 and 2015 Greater NY ACM ICPC Regionals.

Input

The first line of input contains a single integer P, ($1 \leq P \leq 1\,000$), which is the number of data sets that follow. Each data set should be processed identically and independently. Each data set consists of a single line of input. It contains the data set number, K, and the index, N, of the sequence element to compute ($1 \leq N \leq 2\,147\,483\,647$).

Output

For each data set there is a single line of output. It contains the data set number, K, followed by a single space which is then followed by the numerator of the fraction, followed immediately by a forward slash ('/')

followed immediately by the denominator of the fraction. Inputs will be chosen so neither the numerator nor the denominator will overflow an 32-bit **unsigned** integer.

Sample Input 1	**Sample Output 1**

`4`
`1 1`
`2 4`
`3 11`
`4 1431655765`
	`1 1/1`
	`2 1/3`
	`3 5/2`
	`4 2178309/1346269`

Problem E
Permutation Descent Counts

Given a positive integer, N, a permutation of order N is a one-to-one (and thus onto) function from the set of integers from 1 to N to itself. If p is such a function, we represent the function by a list of its values:

$$[\,p(1)\,p(2)\;\ldots\;p(N)\,]$$

For example, $[5\,6\,2\,4\,7\,1\,3]$ represents the function from $1\ldots 7$ to itself which takes 1 to 5, 2 to 6, ... , 7 to 3.

For any permutation p, a descent of p is an integer k for which $p(k) > p(k{+}1)$. For example, the permutation $[5\,6\,2\,4\,7\,1\,3]$ has a descent at 2 ($6 > 2$) and 5 ($7 > 1$).

For permutation p, des(p) is the number of descents in p. For example, des($[5\,6\,2\,4\,7\,1\,3]$) $= 2$. The identity permutation is the only permutation with des(p) $= 0$. The reversing permutation with $p(k) = N + 1 - k$ is the only permutation with des(p) $= N - 1$.

The permutation descent count (PDC) for given order N and value v is the number of permutations p of order N with des(p) $= v$. For example:

```
PDC(3, 0) = 1 { [ 1 2 3 ] }
PDC(3, 1) = 4 { [ 1 3 2 ], [ 2 1 3 ], [ 2 3 1 ], 3 1 2 ] }
PDC(3, 2) = 1 { [ 3 2 1 ] }
```

Write a program to compute the PDC for inputs N and v. To avoid having to deal with very large numbers, your answer (and your intermediate calculations) will be computed modulo 1001113.

Input

The first line of input contains a single integer P, ($1 \leq P \leq 1\,000$), which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of a single line of input. It contains the data set number, K, followed by the integer order, N ($2 \leq N \leq 100$), followed by an integer value, v ($0 \leq v \leq N - 1$).

Output

For each data set there is a single line of output. The single output line consists of the data set number, K, followed by a single space followed by the PDC of N and v as a decimal integer number modulo 1001113.

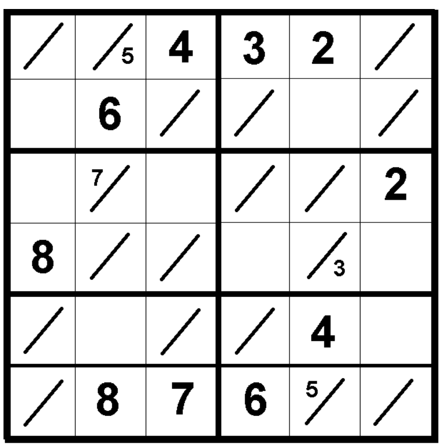
Sample Input 1	Sample Output 1
4 1 3 1 2 5 2 3 8 3 4 99 50	1 4 2 66 3 15619 4 325091

This page is intentionally left blank.

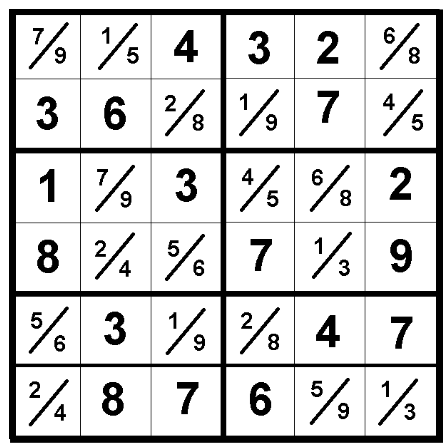
Problem F
Tight-Fit Sudoku

At some point or another, most computer science students have written a standard Sudoku solving program. In a standard Sudoku puzzle, a 9×9 grid that is partially filled in with digits is given. The task is to complete the grid so that each digit occurs exactly once in every row, column, and in each of 9 3×3 non-overlapping subregions marked in the grid. This problem adds a slight twist to standard Sudoku to make it a bit more challenging.

Digits from 1 to 9 are entered in a 6×6 grid so that no number is repeated in any row, column or 3×2 outlined region as shown below. Some squares in the grid are split by a slash and need 2 digits entered in them. The smaller number always goes above the slash.



Incomplete Grid



Solution Grid

For this problem, you will write a program that takes as input an incomplete puzzle grid and outputs the puzzle solution grid.

Input

The input consists of a single data set. This data set consists of 6 lines of input representing an incomplete Tight-Fit Sudoku grid. Each line has 6 data elements, separated by spaces. A data element can be a digit (`1-9`), a dash (‘`-`’) for a blank square or two of these separated by a slash (‘`/`’).

Output

Output 6 lines showing the solution grid for the given input data set. Each line will have 6 data elements, separated by spaces. A data element can be a digit (`1-9`) or 2 digits separated by a slash (‘`/`’).

Sample Input 1	Sample Output 1
-/- -/5 4 3 2 -/- - 6 -/- -/- - -/- - 7/- - -/- -/- 2 8 -/- -/- - -/3 - -/- - -/- -/- 4 - -/- 8 7 6 5/- -/-	7/9 1/5 4 3 2 6/8 3 6 2/8 1/9 7 4/5 1 7/9 3 4/5 6/8 2 8 2/4 5/6 7 1/3 9 5/6 3 1/9 2/8 4 7 2/4 8 7 6 5/9 1/3

Problem G
Magical Mystery Knight's Tour

A knight's tour on a rectangular board of n rows and m columns of squares (traditionally 8-by-8) is a labelling of the squares by integers 1 through $n * m$ so that label $n + 1$ is a knight's move from label n. That is, 2 squares horizontally and 1 square vertically or 1 square horizontally and 2 squares vertically. The image below shows an 8-by-8 knight's tour.

1	56	13	26	3	46	15	28
24	37	2	57	14	27	4	47
55	12	25	38	45	58	29	16
36	23	64	61	42	39	48	5
11	54	41	44	59	62	17	30
22	35	60	63	40	43	6	49
53	10	33	20	51	8	31	18
34	21	52	9	32	19	50	7

A knight's tour (on a square board) is (*semi-*)magical if the sum of the values in each row and column is the same (for the 8-by-8 case the sum would be 260). For this problem, you will be given a sequence of semi-magical 8-by-8 knight's tours with many of the labels removed (see the image below). Write a program to fill in the missing labels so the knight's tour is *semi*-magical.

1	48			33		63	18
30	51		3				
				15			
			45			36	
		25		9		21	60
				24	57	12	
	6			39			
54		42					

Input

The input contains a single data set consisting of 8 lines of input. Each line contains 8 integers separated by spaces giving the labels for the corresponding row. If the label value is -1, the label has been removed and your program is to find the correct value to put in that place.

Output

Output 8 lines containing 8 integers each, separated by spaces, filling in the removed values to give a complete semi-magical knight's tour which includes the positive labels from the input. There may be multiple correct answers. Your result will be graded correct if it is a semi-magical knight's tour and the positive labels from the input are in the same square in your answer.

Note: Your output does not have to be lined up as shown in the Sample Output. Just make sure that each of the 8 lines of output has at least one space between each value on the line.

Sample Input 1

```
1 48 -1 -1 33 -1 63 18
30 51 -1 3 -1 -1 -1 -1
-1 -1 -1 -1 15 -1 -1 -1
-1 -1 -1 45 -1 -1 36 -1
-1 -1 25 -1 9 -1 21 60
-1 -1 -1 -1 24 57 12 -1
-1 6 -1 -1 39 -1 -1 -1
54 -1 42 -1 -1 -1 -1 -1
```

Sample Output 1

```
 1 48 31 50 33 16 63 18
30 51 46  3 62 19 14 35
47  2 49 32 15 34 17 64
52 29  4 45 20 61 36 13
 5 44 25 56  9 40 21 60
28 53  8 41 24 57 12 37
43  6 55 26 39 10 59 22
54 27 42  7 58 23 38 11
```

Problem H
DA-Sort

You recently learned a new way to sort an array of numbers in your algorithms course. The algorithm sorts an array of numbers by repeatedly performing the *Delete-and-Append* operation. The *Delete-and-Append* operation consists of three steps:

1. Choose an element from the array.
2. Delete the chosen element from the array.
3. Append the chosen element to the end of the array.

Being a curious student, you wonder what is the minimum number of *Delete-and-Append* operations required to sort a given array.

Input

The first line of input contains a single decimal integer P, $(1 \leq P \leq 100)$, which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of two or more lines of input. The first line contains the data set number, K, followed by a single space, followed by an integer N, $(1 \leq N \leq 1\,000)$, which is the length of the array to sort. The remaining lines in the dataset contains N positive integers that comprise the array to be sorted, 10 values per line, except for the last line which may have less than 10 values. All the array elements are no larger than 10^9. The same value may appear more than once in the array to be sorted.

Output

For each data set there is one line of output. The single output line consists of the data set number, K, followed by a single space followed by an integer which is the minimum number of *Delete-and-Append* operations required to sort the array.

Sample Input 1

```
3
1 3
1 3 2
2 6
1 5 2 4 3 6
3 23
67890 56312 999999999 12345 23456 38927 45632 100345 98765 23456
87654 43278 23456 117654 321899 25432 54326 217435 26845 31782
33456 41234 56213
```

Sample Output 1

```
1 1
2 3
3 15
```

Problem I
A Mazing!

A maze consists of a collection of equal sized square cells, where any or all of the sides may be a wall or a door. The maze may have no exit or multiple exits. Cells are typically arranged so that they may share sides with other cells as shown in the four sample mazes below:

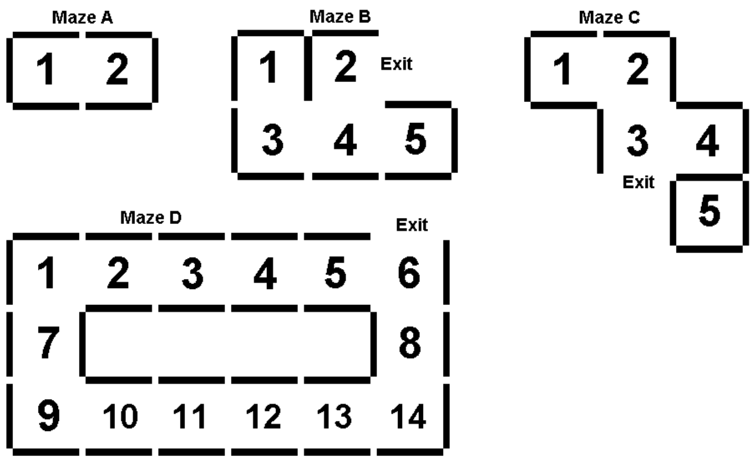
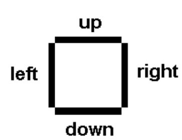


Figure I.1: (The numbers in the cells above are for illustrative purposes only.)

Each side of a cell is labeled with a direction to allow navigation:



For each maze, the starting point is someplace in the maze (for example, any of the numbered cells in the samples above). In the samples above:

- Maze A has no way out.
- Maze B has an exit (solution) to the right of cell 2.
- Maze C has an exit down from cell 3, unless the starting point is cell 5, in which case there is no way out
- Maze D has an exit up from cell 6.

For example, using Maze D above, if the starting point is cell 9, one possible set of directions to get to the exit would be: `right`, `right`, `right`, `right`, `right`, `up`, `up`, `up`.

For this problem, you will write a program that finds an exit to a maze. Your program must operate *interactively*. That is, your program will make a move by providing a direction (right, down, left or up), and the judging software will send back one of four responses:

1. **wall** - indicates that a wall is there and you cannot proceed in that direction
2. **ok** - indicates that there is door there and you may proceed in that direction to the neighboring cell.
3. **solved** - indicates that you have successfully found an exit to the maze.
4. **wrong** - indicates that your program made an error, as discussed below.

If your program determines there is no way out of the maze, you should send the precise string "`no way out`" (without the quotes) instead of a direction. If there is in fact no way out of the maze, you will receive a **solved** reply.

Your program will receive a **wrong** indication if any of the following occur:

1. Your program sends "`no way out`", even though there is a way out.
2. Your program makes the same move (direction) from the same cell twice.

After receiving a **wrong** or a **solved** reply, your program should exit.

Input/Output

This is an interactive program. The input you receive is a function of the output you generate. All input and output strings must end in a new-line character. You should never send extra blank lines.

You must make sure that your program's standard output stream is flushed after you output a the new-line character that completes a command. This is accomplished with `System.out.flush()` in Java, `stdout.flush()` in Python, `fflush(stdout)` in C, and `cout « flush` in C++.

The first thing your program must do when it starts up is to send its first move (`up`, `down`, `right` or `left`), followed by a new-line character. It will then wait for a new-line terminated response on the standard input. The response will be one of **wall**, **ok**, **solved**, or **wrong** indication. Your program will then make another move based on the response it received as discussed above. This process will repeat until your program receives a **wrong** or **solved** indication.

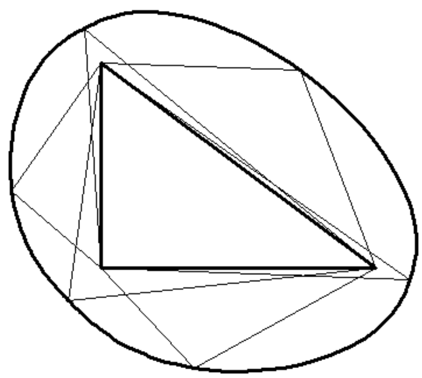
Example (User output in `Teletype`, Computer judge output in **Bold**). (This sample run has no relationship to the samples shown above).

Sample Run
`down`
wall
`right`
wall
`left`
wall
`up`
ok
`right`
ok
`down`
ok
`down`
wall
`right`
wall
`left`
wall
`up`
ok
`right`
solved

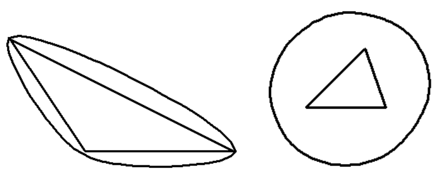
This page is intentionally left blank.

Problem J
Smoothed Gardens

The *Flathead Fancy Landscaping Company's* customers are too high-class to have gardens with straight edges so Joe P. Flathead, the owner, has come up with a way to smooth out the contours. He puts a stake in each corner of a triangular plot and drops a loop of rope around the three stakes. Then using a fourth stake in the loop, he pulls the rope tight to mark out a smoothed version of the triangle (see the figure below, the thinner lines are various positions of the rope). This process is similar to the method you learned in middle school to draw an ellipse using 2 push-pins, a piece of string and a pencil, but J.P. Flathead is using *three* stakes (not two), a rope and another stake instead of a pencil.



The longer the rope loop, the smoother the outline will be (see the examples below):



In order to determine how much soil and how many plants are required for the garden, Joe needs to find out the area of the resulting smoothed outline.

For this problem you will write a program which takes as input the coordinates of the corners of the triangle and the length of the rope loop and outputs the area of the smoothed region. The coordinate system will be chosen so that the first vertex is at the origin ($A(0, 0)$), the x-axis is along the line from the first vertex to the second vertex ($B(B_x, 0)$) and the final vertex is above the x-axis ($C(C_x, C_y)$).

Input

The first line of input contains a single decimal integer P, ($1 \leq P \leq 10\,000$), which is the number of data sets that follow. Each data set should be processed identically and independently.

Each data set consists of a single line of input. It contains the data set number, K, followed by a single space, followed by 4 floating point values B_x, C_x, C_y, ($0 < B_x \leq 100$, $0 < C_y \leq 100$, $-100 \leq C_x \leq 100$), and the rope length L ($L < 1\,000$) all measured in feet. It is guaranteed that points A, B, C form a triangle with a positive area. The length L of the rope will be at least 0.05 feet larger than the perimeter of this triangle. All floating numbers will be provided with no more than 3 digits after the period.

Output

For each data set there is one line of output. The single output line consists of the data set number, K, followed by a single space followed by the area of the smoothed region in square feet. Your answer will be consider correct if its absolute error does not exceed 10^{-2}.

Sample Input 1	Sample Output 1
3 1 4 0 3 13 2 3 -2 3 14.5 3 4 3 3 14	1 23.486243 2 37.460673 3 42.239747