2014 SCUSA Scripting Contest

A • Mispelling$_4$

Misspelling is an art form that students seem to excel at. Write a program that removes the *n*th character from an input string.

Input

The first line of input contains a single integer ***N***, ($1 \leq N \leq 1000$) which is the number of datasets that follow.

Each dataset consists of a single line of input containing ***M***, a space, and a single word made up of uppercase letters only. ***M*** will be less than or equal to the length of the word. The length of the word is guaranteed to be less than or equal to 80.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and the misspelled word. The misspelled word is the input word with the indicated character deleted.

Sample Input	Sample Output
4 4 MISSPELL 1 PROGRAMMING 7 CONTEST 3 BALLOON	1 MISPELL 2 ROGRAMMING 3 CONTES 4 BALOON

2014 SCUSA Scripting Contest

B • Conversions

Conversion between the *metric* and *English* measurement systems is relatively simple. Often, it involves either multiplying or dividing by a constant. You must write a program that converts between the following units:

Type	Metric	English equivalent
Weight	`1.000 kilograms`	`2.2046 pounds`
	`0.4536 kilograms`	`1.0000 pound`
Volume	`1.0000 liter`	`0.2642 gallons`
	`3.7854 liters`	`1.0000 gallon`

Input

The first line of input contains a single integer ***N***, ($1 \le N \le 1000$) which is the number of datasets that follow.

Each dataset consists of a single line of input containing a floating point (double precision) number, a space and the *unit specification* for the measurement to be converted. The *unit specification* is one of **`kg`**, **`lb`**, **`l`**, or **`g`** referring to kilograms, pounds, liters and gallons respectively.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and the appropriately converted value rounded to 4 decimal places, a space and the *unit specification* for the converted value.

Sample Input	Sample Output

`5`
`1 kg`
`2 l`
`7 lb`
`3.5 g`
`0 l`
	`1 2.2046 lb`
	`2 0.5284 g`
	`3 3.1752 kg`
	`4 13.2489 l`
	`5 0.0000 g`

2014 SCUSA Scripting Contest

C • Encoding

Chip and Dale have devised an encryption method to hide their (written) text messages. They first agree secretly on two numbers that will be used as the number of rows (**R**) and columns (**C**) in a matrix. The sender encodes an intermediate format using the following rules:

1. The text is formed with uppercase letters [**A-Z**] and **<space>.**
2. Each text character will be represented by decimal values as follows:

   <space> = 0, **A** = 1, **B** = 2, **C** = 3, ..., **Y** = 25, **Z** = 26

The sender enters the 5 digit *binary* representation of the characters' values in a spiral pattern along the matrix as shown below. The matrix is padded out with zeroes (**0**) to fill the matrix completely. For example, if the text to encode is: "**ACM**" and **R=4** and **C=4**, the matrix would be filled in as follows:

```
0→0→0→0
      ↓
1→1→0 1
↑   ↓ ↓
0 0←1 0
↑     ↓
1←1←0←0
```

A = 00001, C = 00011, M = 01101
(one extra 0)

The bits in the matrix are then concatenated together in *row major* order and sent to the receiver. The example above would be encoded as: **0000110100101100**

This problem continues on the next page...

2014 SCUSA Scripting Contest

Input

The first line of input contains a single integer ***N***, ($1 \le N \le 1000$) which is the number of datasets that follow.

Each dataset consists of a single line of input containing **`R`** **`(1<=R<=20)`**, a space, **`C`** **`(1<=C<=20)`**, a space, and a text string consisting of uppercase letters [**`A-Z`**] and **`<space>.`** The length of the text string is guaranteed to be **`<= (R*C)/5`**.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and a string of binary digits (**`R*C`**) long describing the ***encoded*** text. The binary string represents the values used to fill in the matrix in *row-major* order. You may have to fill out the matrix with zeroes (**`0`**) to complete the matrix.

Sample Input	Sample Output
4 4 4 ACM 5 2 HI 2 6 HI 5 5 HI HO	1 0000110100101100 2 0110000010 3 010000001001 4 0100001000011010110000010

2014 SCUSA Scripting Contest

D • Decoding

Chip and Dale have devised an encryption method to hide their (written) text messages. They first agree secretly on two numbers that will be used as the number of rows (**R**) and columns (**C**) in a matrix. The sender encodes an intermediate format using the following rules:

1. The text is formed with uppercase letters [**A-Z**] and **<space>.**
2. Each text character will be represented by decimal values as follows:

   <space> = 0, **A** = 1, **B** = 2, **C** = 3, ..., **Y** = 25, **Z** = 26

The sender enters the 5 digit *binary* representation of the characters' values in a spiral pattern along the matrix as shown below. The matrix is padded out with zeroes (**0**) to fill the matrix completely. For example, if the text to encode is: "**ACM**" and **R=4** and **C=4**, the matrix would be filled in as follows:

```
0→0→0→0
      ↓
1→1→0 1
↑   ↓ ↓
0 0←1 0
↑     ↓
1←1←0←0
```

A = 00001, C = 00011, M = 01101
(one extra 0)

The bits in the matrix are then concatenated together in *row major* order and sent to the receiver. The example above would be encoded as: **0000110100101100**

This problem continues on the next page...

2014 SCUSA Scripting Contest

Input

The first line of input contains a single integer ***N***, ($1 \leq N \leq 1000$) which is the number of datasets that follow.

Each dataset consists of a single line of input containing `R (1<=R<=20)`, a space, `C (1<=C<=20)`, a space, and a string of binary digits that represents the contents of the matrix (**R * C** binary digits). The binary digits are in *row major* order.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and the ***decoded*** text message. You should throw away any trailing spaces and/or partial characters found while decoding.

Sample Input	Sample Output
4 4 4 0000110100101100 5 2 0110000010 2 6 010000001001 5 5 0100001000011010110000010	1 ACM 2 HI 3 HI 4 HI HO

2014 SCUSA Scripting Contest

E • Flipping Burned Pancakes

The cook at the *Frobbozz Magic Pancake House* sometimes falls asleep on the job while cooking pancakes. As a result, one side of a stack of pancakes is often burned. Clearly, it is bad business to serve visibly burned pancakes to the patrons. Before serving, the waitress will arrange the stacks of pancakes so that the burned sides are facing down. You must write a program to aid the waitress in stacking the pancakes correctly.

We start with a stack of ***N*** pancakes of distinct sizes, each of which is burned on one side. The problem is to convert the stack to one in which the pancakes are in size order with the smallest on the top and the largest on the bottom and burned side down for each pancake. To do this, we are allowed to flip the top ***k*** pancakes over as a unit (so the ***k-th*** pancake is now on top and the pancake previously on top is now in the ***k-th*** position and the burned side goes from top to bottom and *vice versa*).

For example (+ indicates burned bottom, - a burned top):

+1 -3 -2 [flip 2] ⇒ +3 -1 -2 [flip 1] ⇒ -3 -1 -2 [flip 3] ⇒
+2 +1 +3 [flip 1] ⇒ -2 +1 +3 [flip 2] ⇒ -1 +2 +3 [flip 1] ⇒ +1 +2 +3

You must write a program which finds a sequence of at most ***(3n – 2)*** flips, which converts a given stack of pancakes to a sorted stack with burned sides down.

Input

The first line of the input contains a single decimal integer, ***N***, the number of problem instances to follow. Each of the following ***N*** lines gives a separate dataset as a sequence of numbers separated by spaces. The first number on each line gives the number, ***M***, of pancakes in the data set. The remainder of the data set is the numbers ***1*** through ***M*** in some order, each with a plus or minus sign, giving the initial pancake stack. The numbers indicate the relative sizes of the pancakes and the signs indicate whether the burned side is up (-) or down (+). ***M*** will be, at most, **30**.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, the number of flips (***K,*** where ***K*** **>= 0**) required to sort the pancakes and a sequence of ***K*** numbers, each of which gives the number of pancakes to flip on the corresponding sorting step. There may be several correct solutions for some datasets. For instance `3 2 3` is also a solution to the first problem below.

2014 SCUSA Scripting Contest

Sample Input	Sample Output
3 3 +1 -3 -2 4 -3 +1 -2 -4 5 +1 +2 +3 +4 -5	1 6 2 1 3 1 2 1 2 6 4 1 4 3 1 2 3 3 5 1 5

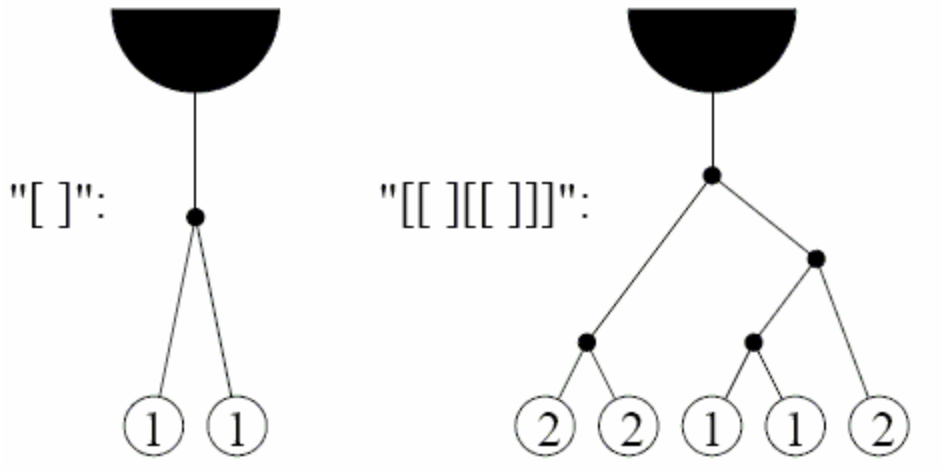
2014 SCUSA Scripting Contest

F • Monkey Vines

Deep in the Amazon jungle, exceptionally tall trees grow that support a rich biosphere of figs and juniper bugs, which happen to be the culinary delight of brown monkeys.

Reaching the canopy of these trees requires the monkeys to perform careful navigation through the tall tree's fragile vine system. These vines operate like a see-saw: an unbalancing of weight at any vine junction would snap the vine from the tree, and the monkeys would plummet to the ground below. The monkeys have figured out that if they work together to keep the vines properly balanced, they can *all* feast on the figs and juniper bugs in the canopy of the trees.

A *vine junction* supports exactly two *sub-vines,* each of which must contain the same number of monkeys, or else the vine will break, leaving a pile of dead monkeys on the jungle ground. For purposes of this problem, a *vine junction* is denoted by a pair of matching square brackets **[ ]**, which may contain nested information about junctions further down its *sub-vines*. The nesting of vines will go no further than **25** levels deep.



You will write a program that calculates the *minimum* number of monkeys required to balance a particular vine configuration. There is **always** at least one monkey needed, and, multiple monkeys may hang from the same vine.

2014 SCUSA Scripting Contest

Input

The first line of input contains a single integer ***N***, ($1 \leq N \leq 1000$) which is the number of datasets that follow.

Each dataset consists of a single line of input containing a vine configuration consisting of a string of `[` and `]` characters as described above. The length of the string of `[` and `]` will be greater than or equal to zero, and less than or equal to 150.

Output

For each dataset, you should generate one line of output with the following values: The dataset number as a decimal integer (start counting at one), a space, and the minimum number of monkeys required to reach the canopy successfully. Assume that all the hanging vines are reachable from the jungle floor, and that all monkeys jump on the vines at the same time.

Sample Input	Sample Output
3 [] [[][[]]]	1 2 2 1 3 8

Note: The second line of sample input is a blank line.

2014 SCUSA Scripting Contest

G • Model Rocket Height

Just when you thought we had run out of model rocket height problems…

Yet another method used to determine the height achieved by a model rocket is the *vertical line* method. Two observers A and B are spaced D feet apart along a base line along one edge of the flat test field. The launch platform is equidistant from observers A and B and L feet from the base line. Each observer has a theodolite or some other device for measuring angle above the horizontal (elevation angle) of a distant object and the azimuth angle (the angle the vertical plane of the sight line makes with the line from A through B measured counter-clockwise). Each measuring device is on a stand. A's device is HA feet above the level of the launch platform and B's device is HB feet above the level of the launch platform. When a rocket is fired, near the top of its flight, it deploys a parachute and emits a puff of smoke. Each observer measures the elevation angle and azimuth angle of the puff of smoke from their location. If the peak location is on the wrong side of the baseline or outside the lines determined by A and B perpendicular to the base line, it is out of bounds and disqualified. From this information, the height of the rocket may be determined as follows:

Each sight line determines a vertical plane. These two planes intersect in a vertical line (thus the name of the method). Each sight line intersects this vertical line in a point. If these points are more than ERRDIST feet apart, an error is assumed and the flight is rejected. Otherwise, the point halfway between the two points where a sight line intersects the vertical line is computed. The rocket height is the distance of this midpoint above the launch platform.

You must write a program which, given the parameters D (the distance in feet between observers A and B), L (the distance in feet from the base line to the launch platform), HA (the distance of the measuring device A above the launch platform in feet), HB (the distance of the measuring device B above the launch platform in feet), ERRDIST (the maximum distance between the intersection points of a sight line with the vertical line), α (the elevation angle of the rocket in degrees measured by the left observer A), β (the elevation angle of the rocket in degrees observed by the right observer B), γ (the azimuth angle in degrees measured by the left observer A) and δ (the azimuth angle in degrees measured by the right observer B), computes the height of the rocket above the launch platform in feet to the nearest foot.

(This problem is continued on the next page)

2014 SCUSA Scripting Contest

Input

The first line of input contains a single integer ***N***, ($1 \leq N \leq 1000$) which is the number of datasets that follow.

The second line contains the parameters D, L, HA, HB and ERRDIST in that order as (floating point) decimal values. These values would be measured once at the beginning of the day and remain fixed through all rocket shots.

Each succeeding line of input represents a single dataset. Each dataset will contain the angles α, β, γ and δ in that order (measured in degrees) as (floating point) decimal values for a rocket shot.

Output

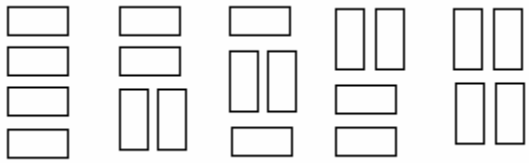
For each dataset of four angles, the output consists of a single line . If angles α, β and γ are not strictly between 0 and 90 degrees or δ is not strictly between 90 degrees and 180 degrees, the line should contain the dataset number, a space and the word “DISQUALIFIED” (without the quotes). Otherwise, if the distance between the intersection points of a sight line with the vertical line is more that ERRDIST feet, the line should contain the dataset number, a space and the word “ERROR” (without the quotes). Otherwise, the line should contain the dataset number, a space and the height above the launch platform in feet to the nearest foot.

Sample Input	Sample Output
4 100.0 300.0 5.25 2.92 5.00 40.1 36.2 35.3 151.6 64.9 71.1 15.7 160.1 44.9 41.2 33.1 152.5 44.9 41.2 33.1 52.5	1 50 2 ERROR 3 58 4 DISQUALIFIED

2014 SCUSA Scripting Contest

H • Tiling a Grid With Dominoes

We wish to tile a grid 4 units high and ***N*** units long with rectangles (dominoes) 2 units by one unit (in either orientation). For example, the figure shows the five different ways that a grid 4 units high and 2 units wide may be tiled.



Write a program that takes as input the width, ***W***, of the grid and outputs the number of different ways to tile a 4-by-***W*** grid.

Input

The first line of input contains a single integer ***N***, ($1 \le N \le 1000$) which is the number of datasets that follow.

Each dataset contains a single decimal integer, the width, ***W***, of the grid for this problem instance.

Output

For each problem instance, there is one line of output: The problem instance number as a decimal integer (start counting at one), a single space and the number of tilings of a 4-by-***W*** grid. The values of ***W*** will be chosen so the count will fit in a 32-bit integer.

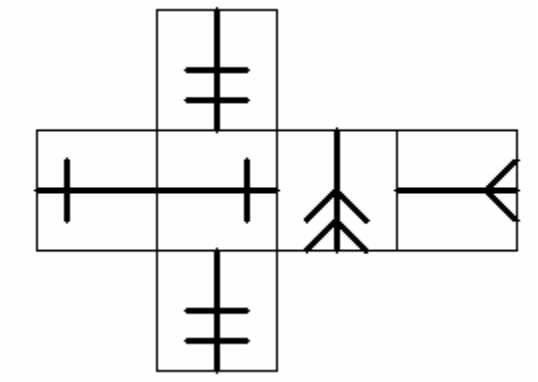
Sample Input	Sample Output
3 2 3 7	1 5 2 11 3 781

2014 SCUSA Scripting Contest

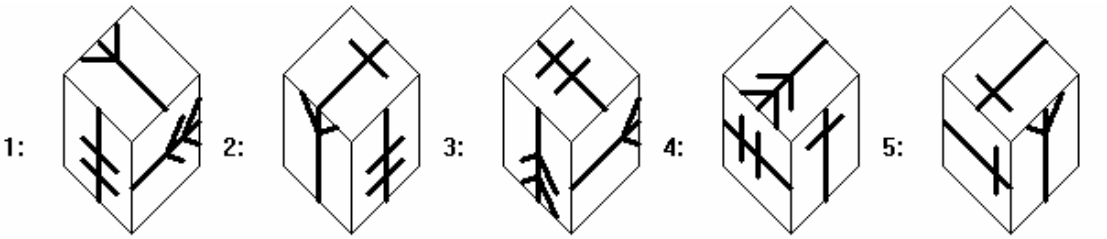
I • Spatial Concepts Test

The *Flathead Testing Corporation* (FTC) supplies various tests for Human Resources departments at many companies. One type of test they supply includes spatial concepts questions such as:

When the following figure is folded back on the interior lines it forms a cube.



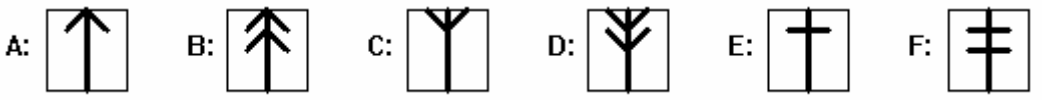
Which of the following could be an image of one corner of the resulting cube?



Unfortunately, FTC was recently embarrassed when one such question on a test had no solution among the choices and another (given in the example) had two solutions among the choices (1 and 3).

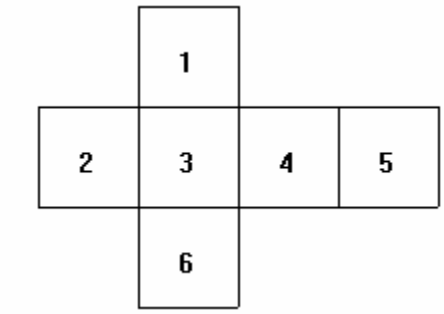
FTC needs a routine which will read in a specification of the unfolded cube and specifications of corner views and determine, for each corner view, whether it is a view of a corner of the cube specified in the unfolded part.

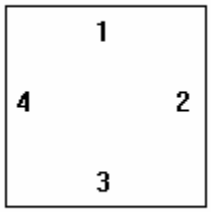
FTC uses the following images as faces of each cube. Each image is symmetrical about the vertical axis and has a distinguished end (up in each image).



2014 SCUSA Scripting Contest

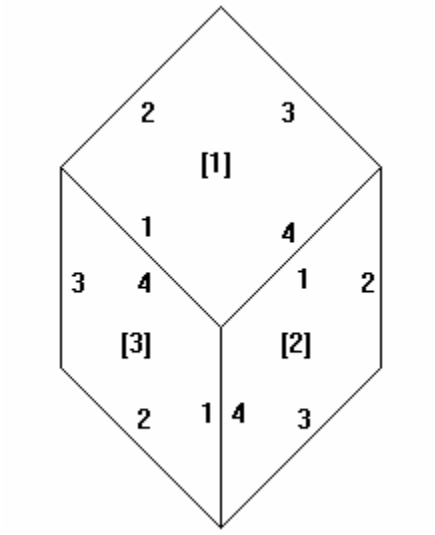
The unfolded cube is specified by a string of six pairs of a letter indicating the image on the face and a number indicating the orientation of the distinguished end of the face: 1 is up, 2 is right, 3 is down and 4 is left. The faces are specified in the order given in the following figure with the orientations indicated in the square to the right:





So the unfolded cube in the example is specified as "**F3E4E2D3C2F3**". FTC has a routine which reads this specification and generates the unfolded image for the question.

The answer images are specified by three pairs of a letter and a digit indicating a face image and an orientation as indicated in the following diagram. The faces are specified in the order top, right, left (indicated by numbers in brackets in the figures), that is clockwise around the center vertex starting at the top. The orientation of the distinguished end of each face is indicated by the numbers on the edges in the diagram. They circle each face clockwise, starting at the center vertex.



For the example, the answer figures are specified as "**C2D2F2**", "**E3F3C4**", "**F2C2D2**", "**D1E1F3**" and "**E1C1E1**". Again, FTC has a routine which reads this specification and generates each answer image for the question. They just need your routine to make sure there is exactly one correct answer to each question.

2014 SCUSA Scripting Contest

Input

The first line of input contains a single integer **N**, (1 ≤ **N** ≤ 1000) which is the number of datasets that follow.

Each dataset consists of six lines of input. The first line of input is the specification for the folded out cube as described above. This line is followed by five lines, each of which gives the specification of one answer image as described above.

Output

For each dataset, output on a single line the dataset number, (1 through **N**), a blank, the number of answers which are solutions of the problem (corners of the cube specified in the folded out line), a blank and five 'Y' or 'N' characters separated by a blank indicating which of the answer images was a solution ('Y' for a solution, 'N' for not a solution).

Sample Input	Sample Output
2 F3E4E2D3C2F3 C2D2F2 E3F3C4 F2C2D2 D1E1F3 E1C1E1 A2F4F1A3A3C4 C3A4A2 F3F4A1 F3C4A1 A2C3A2 A4A4F1	1 2 Y N Y N N 2 0 N N N N N