MERCER UNIVERSITY

Spring Programming Competition

February 27, 2010

1.	Grocery Help	(`prob1`)
2.	Beads (easier)	(`prob2`)
3.	Word Stats	(`prob3`)
4.	Starbucks Search	(`prob4`)
5.	Grid Game	(`prob5`)
6.	Lattices	(`prob6`)
7.	Counting Trees	(`prob7`)
8.	The 8-Puzzle	(`prob8`)
9.	Beads (harder)	(`prob9`)
10.	Primes	(`prob10`)
11.	Alien DNA	(`prob11`)

The ***name*** in parenthesis following each problem is the name you must use as your program's name. You must add the appropriate extension depending upon your choice of programming language (.c .cpp .cs .java .py .rb).

Grocery Help (prob1)

Your company, Calculation Solutions, which specializes in accounting software, has been contacted by the Patel Grocers for help.

At the end of each day at the grocery store, the owner, Mr. Patel has a list of transactions. Each transaction is of the form:
1) p q: which means q items costing p dollars each were sold.
2) p: which means an item costing p dollars was sold.
3) –p: which means an item costing p dollars was returned.

Mr. Patel would like for you to help him by generating the revenue his store had on each day.

Input

The first line contains an integer N ($1 \le N \le 100$), the total number of days. Each day starts with an integer T ($0 \le T \le 100$) on a line, T being the total number of transactions on that day. It is followed by T lines containing transactions of type 1, 2 or 3. Since Mr. Patel's store is mid-sized, p and q are relatively small ($1 \le p, q \le 10$). The transactions are in the format:
"1 p q", "2 p" or "3 –p" (quotes for clarity).

Output:

For each day D (D=1 .. N), output the revenue R for that day in the format.
Day D: $R.

Sample Input

```
3
2
1 4 5
2 6
1
3 -7
0
```

Sample Output

```
Day 1: $26.
Day 2: $-7.
Day 3: $0.
```

Beads (prob2)

Lana and Nala enjoy being identical twins. They wear the same outfits every day, including jewelry. They especially love necklaces of colored beads, but they continually run into a problem trying to determine whether two beaded necklaces are identical or not.

Beads come in 26 colors, and the colors are identified by the lower case alphabet. A beaded necklace containing n beads is represented by a string of n lower case alphabetic letters. The necklaces are continuous (no clasps).

It is easy to see that the two beaded necklaces "abcdefg" and "abcdefg" are identical. But it is not quite so obvious that the necklaces "efgabcd" and "bagfedc" are also identical to "abcdefg".

The twins need you to write a program for them that will determine whether or not two beaded necklaces are identical.

Input

The input will begin with an integer C ($1 \leq C \leq 100$) that denotes the number of test cases. The rest of the input will consist of C pairs of necklaces, each on a line by itself. Each necklace contains at least 1 and no more than 99 beads.

Output

The output should display the case number (as shown below) followed by either "YES" or "NO" indicating whether the necklace pair is identical or not.

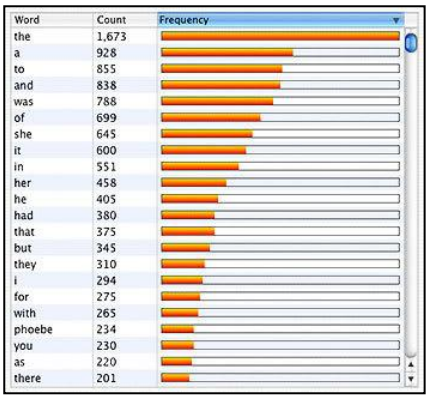
Sample Input

```
4
lana
nala
abcdefg
efgabcd
abcdefg
bagfedc
abcdefg
opqrstu
```

Sample Output

```
Case #1: YES
Case #2: YES
Case #3: YES
Case #4: NO
```

Word Stats (prob3)



Word	Count	Frequency
the	1,673	
a	928	
to	855	
and	838	
was	788	
of	699	
she	645	
it	600	
in	551	
her	458	
he	405	
had	380	
that	375	
but	345	
they	310	
i	294	
for	275	
with	265	
phoebe	234	
you	230	
as	220	
there	201	

We all know how to crunch statistics with numbers. But, let's crunch some statistics with words! That's your job with this program. You want to input all the words from a text file, convert them to lowercase, sort them, then calculate the median of your word distribution, and the mode (the word with the largest frequency).

The median is defined as the word in the middle of your sorted distribution. If you have an odd number of words, there is only one median. But, if you have an even number of words, you will have two words that define your median.

The mode is the word that occurs more often than any others. In the event of a tie (multiple words with the same largest frequency), you will print all words with the largest frequency.

Input

Your program will take one or more words as input. The number of words will be ≤ 200. A word is defined as one or more contiguous characters followed by white space (space, tab, or newline). The length of each word is ≤ 100. All words should be converted to lowercase upon input, and any non-alphabetic characters that start, end, or are contained within the word should be removed.

Output

Your median should be output first on a single line with a prompt as shown below. If you have an odd number of words in your input, a single median word will be output. If you have an even number of words, you will output two words with a single comma between them, and brackets surrounding them. For example, "My median=[easter,egg]".

Your mode should be output on a single line following the median with a prompt as shown below. Include the number of occurrences of the mode in parentheses following. Enclose your answer with brackets like you did for median. In the event of a tie, use a single comma to separate your words & their occurrences. For example, "My mode=[an(23),the(23)]".

With ties, all words should be listed in alphabetical order for both the median and mode.

Sample Input

```
When April with his sweet showers has
pierced the drought of March to the root,
and bathed every vein in such moisture
as has power to bring forth the flower
```

Sample Output

```
My median=[moisture,of]
My mode=[the(3)]
```

Starbucks Search (prob4)

Traveling from Greenville, SC to Macon, GA, the BJU Programming Team decided it would be fun to try and visit all the Starbucks along the way. Of course "along the way" could be interpreted many ways. So for the purpose of the programming contest, we shall define it as " ≤ 2 miles of the interstate exit." Your job is to write a program that will give the BJU Programming Team a list of the Starbucks that meet this criteria.

Input

The location of all the Starbucks in SC and GA. Each Starbucks will be on a line by itself in the format:
Name of exit,Distance of exit from Greenville,Distance of Starbucks from exit

Distance of exit from Greenville will be an integer ≤ 500. *Distance of Starbucks from exit* will be a floating point number with at most 2 digits of precision.

There will be maximum 3000 Starbucks locations. Input is terminated by the string "END" (quotes for clarity). There will be **no** leading or trailing spaces in any of the lines in the input data.

Output

A list, one per line, of all the Starbucks which meet the criteria, in the order in which we will encounter them on our journey to Macon, without backtracking on the interstate. If multiple Starbucks are at the same exit, they should be listed in order of increasing distance from the exit. There will be no two Starbucks at the same exit with the same distance from the exit.

Sample Input

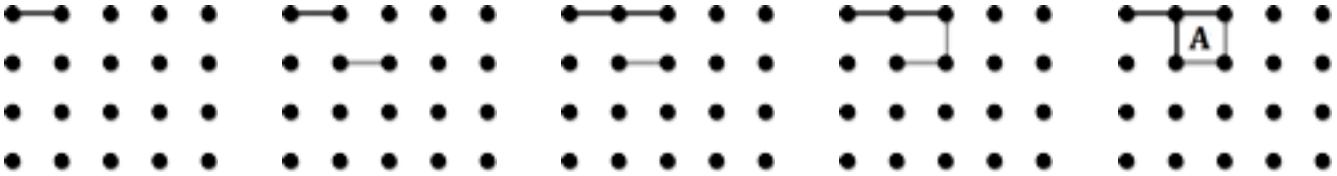
```
Epps Bridge Road,40,1.5
Tiger Blvd,25,4.2
Main Street,25,1.0
Your Street,25,50
Green Street,150,4
END
```

Sample Output

```
Main Street, Exit 25
Epps Bridge Road, Exit 40
```

Grid Game (prob5)

A grid game appears on kids' placemats at many restaurants. The game board comprises a rectangular grid of dots. The figure (on the right) in the example is a 5x4 grid. Two players, referred to as *A* and *B*, take turns connecting two adjacent dots in a vertical or horizontal direction. When a player draws a line that closes a unit square (or "box"), then he claims a point for that box. The game ends when all boxes have been closed and the winner is the player with the most points. A tie is possible if the number of boxes is even. A sequence of moves is shown below. A line made by *A* is thick, by *B* is thin. A point is recorded by putting an *A* or *B* in a box, depending on which player closed it.



The lines drawn by each player in a given game can be recorded as a sequences of moves, each move recorded as a number followed by a letter. First, number the dots row-by-row, starting at 0 for the upper-left dot, 1 for the dot right to 0. Then, specify a move by giving the starting dot number and a direction (**R**:right, **D**:down). For example, 0R connects the two dots at the left end of the top row of dots and 0D connects the two dots at the top end of the first column. The game above is recorded as 0R6R1R2D1D... (it continues until all boxes are formed).

Input

The first line of input contains a positive integer N ($1 \le N \le 200$) giving the number of lines that follow. Each of the lines after the first gives the record of a game. Assume Player A always moves first. All moves are valid. The maximum grid size is 50 x 50 dots.

Output

The output for each line is the number of the game, starting with 1, followed by the size of the grid <number of rows>X<number of columns>, followed by the result, one of A WINS, B WINS, or TIE.

Sample Input

```
4
0R1D2R0D
0R1D4R4D7R6R2D3D5D3R1R0D
0R1R0D2D3R4R1D
3D0D2R4R2D1D6R0R1R5R
```

Sample Output

```
1 2X2 B WINS
2 3X3 TIE
3 3X2 A WINS
4 4X2 B WINS
```

Lattices (prob6)

A lattice is an important mathematical concept that has applications in number theory, group theory, coding theory, cryptography, material science, solid-state physics, computational physics, etc. We are going to deal with two-dimensional lattices.

A lattice point is an ordered pair (x, y) where x and y are both integers. Given the coordinates of the vertices of a triangle (which happen to be lattice points), you are to count the number of lattice points which lie completely inside of the triangle (points on the edges or vertices of the triangle do not count).

Input

The input will contain multiple test cases (maximum of 200 test cases). Each input test case consists of six integers x1, y1, x2, y2, x3, y3 where (x1, y1), (x2, y2), (x3, y3) are the coordinates of vertices of the triangle. All triangles in the input will be non-degenerate (will have positive area), and be in the range $-15000 \leq x1, y1, x2, y2, x3, y3 \leq 15000$. The end of input is marked by a test case with x1 = y1 = x2 = y2 = x3 = y3 = 0 and should not be processed.

Output

For each test case, the program should print the number of internal lattice points on a single line.

Sample input

```
0 0 1 0 0 1
0 0 5 0 0 5
0 0 0 0 0 0
```

Sample output

```
0
6
```

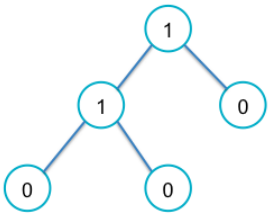
Counting Trees (prob7)

You own your company, UberTries Solutions. One of your clients wants to represent binary trees in a compressed format. He proposes an ingenious method of compressing binary trees by performing an in-order traversal, and saving the resulting string.

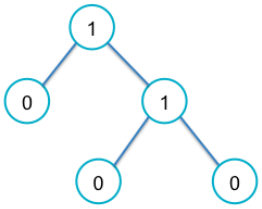
You being the keen problem solver, notice that his method has errors. In fact, you notice that there are multiple binary trees that you can recreate from the same string representation. Your task is to find the total [1] number of such valid binary trees.

You will be given a rooted binary tree, which means that the tree has a root node, and each internal node can have either *one* or *two* children. The internal nodes are numbered 1s, and leaves are numbered 0s. You will be given a string representation of the binary tree. The string representation is a result of performing the standard in-order traversal of the given tree. An outline of an in-order traversal is given below.

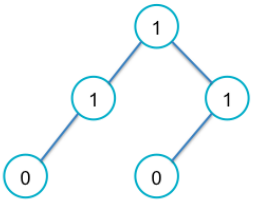
```
in-order-traversal(node)
    if node is leaf then
        print(0)
    else
        if node.left exists then
            in-order-traversal(node.left)
        end if
        print(1)
        if node.right exists then
            in-order-traversal(node.right)
        end if
    end if
```



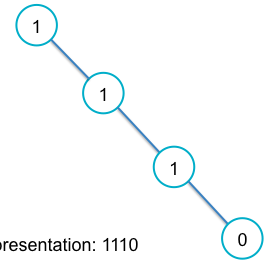
i) Representation: 01010



ii) Representation: 01010



iii) Representation: 01101



iv) Representation: 1110

Figures i, ii, iii and iv shows examples of binary trees and its string representation. Note that internal nodes can have *one* child too. Also, notice that tree (i) and tree (ii) share the same string representation. In fact, for the string representation 01010, there are only 2 possible binary trees.

Input

The first line contains the number of test cases, N ($1 \leq N \leq 100$). In each of the following N lines, there will be a string S ($1 \leq$ length of S ≤ 100) consisting of 1s and 0s only.

Output

For each string S, output the number of possible valid trees modulo [2] 1,000,001 (one million and one) in a single line.

Sample Input

```
5
0
010
01010
0110010
1
```

Sample Output

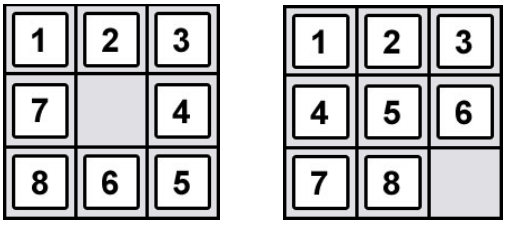
```
1
1
2
0
0
```

Notes:
a) Beware of integer overflow.
b) Beware of run time limit exceeded.

[1] Not the total but the total modulo 1,000,001.
[2] A modulo N = A%N (in Java, C++, C #, Python).

The 8-Puzzle (prob8)

The 8-puzzle consists of a 3x3 grid and 8 tokens numbered 1 to 8. Each token occupies one of the positions in the grid leaving only one position open. Here is an example of how the 8-puzzle would look (see figure on left).



The only valid operation is to move a token from its position to the open one, and this is only allowed if the token we are moving shares an edge with the open position. For instance in the above example (see figure on left), we could either move 7 to the right, 6 up, 2 down, or 4 to the left but we cannot move 1, 3, 8, or 5. The 8-puzzle is, given any random starting position for the eight tokens, to find the minimum number of movements required to obtain the final positioning (see figure on right).

Input

The first line of the input will contain an integer N ($1 \leq N \leq 10{,}000$) which represents the number of test cases. Then, N test cases follow consisting each of three lines. Each line will contain 3 integers (0 to 8) separated by spaces. The number 0 will represent the open position in the grid. Every test case will contain the number 0 to 8 exactly once.

Output

Output a single line containing the minimum number of moves required to reach the final position. Each input is guaranteed to have a finite solution.

Sample Input

```
3
1 2 3
4 5 6
7 8 0
1 2 3
7 0 4
8 6 5
4 2 6
3 1 5
7 0 8
```

Sample Output

```
0
8
17
```

Beads (prob9)

Lana and Nala enjoy being identical twins. They wear the same outfits every day, including jewelry. They especially love necklaces of colored beads, but they continually run into a problem trying to determine whether two beaded necklaces are identical or not.

Beads come in 26 colors, and the colors are identified by the lower case alphabet. A beaded necklace containing n beads is represented by a string of n lower case alphabetic letters. The necklaces are continuous (no clasps).

It is easy to see that the two beaded necklaces "abcdefg" and "abcdefg" are identical. But it is not quite so obvious that the necklaces "efgabcd" and "bagfedc" are also identical to "abcdefg".

The twins need you to write a program for them that will determine whether or not two beaded necklaces are identical.

Input

The input will begin with an integer C ($1 \leq C \leq 100$) that denotes the number of test cases. The rest of the input will consist of C pairs of necklaces, each on a line by itself. Each necklace contains at least one and no more than 100,000 beads.

Output

The output should display the case number (as shown below) followed by either "YES" or "NO" indicating whether the necklace pair is identical.

Sample Input

```
4
lana
nala
abcdefg
efgabcd
abcdefg
bagfedc
abcdefg
opqrstu
```

Sample Output

```
Case #1: YES
Case #2: YES
Case #3: YES
Case #4: NO
```

Primes (prob10)

Laurie and Bob like to encrypt data. They have a simple encryption scheme. Given a number (which represents an 8-bit character), they encode it by replacing the number with the n^{th} prime sum. The n^{th} prime sum is the sum of all the prime numbers ≤ the n^{th} (0-indexed) prime number. The first 8 n^{th} prime sums are shown below.

N	0	1	2	3	4	5	6	7	..
n^{th} Prime Number	2	3	5	7	11	13	17	19	..
n^{th}Prime Sum	2	5	10	17	28	41	58	77	..

Input

The first line contains an integer T ($1 \leq T \leq 100$), T being the number of test cases. In the next T lines, each line contains an integer n ($0 \leq n \leq 255$).

Output

For each integer n, output the special code.

Sample Input

```
3
0
5
7
```

Sample Output

```
2
41
77
```

Alien DNA (prob11)

An alien species has a strange genome with a DNA sequence of only two types of nucleotides: `A` and `B`. A valid DNA sequence must also obey the constraint that no two `A`'s can be adjacent in the sequence. For example, `ABBAB` is a valid sequence and `BAABA` is invalid. Such a DNA sequence has a natural orientation, so the reverse of a sequence is not necessarily the same as the original. For example, `ABBAB` and `BABBA` are considered two distinct sequences.

Dr. Watson is studying the alien DNA and he is interested in the number of possible sequences. Because the length n of a DNA sequence can be extremely large, it may be difficult to use the total number $f(n)$ of valid sequences directly. Instead, Dr. Watson would like to know the remainder of the number of sequences modulo m. (In a project for a future date but *not* today, using the Chinese Remainder Theorem, he should be able to recover the total number with properly chosen moduli.) You are to write a program to compute $f(n) \bmod m$, the number of valid alien DNA sequences of a given length modulo m.

Input

There will be several sets of input. Each set will consist of two integers on a single line:

n m

The first number, n ($1 \leq n \leq 8{,}000{,}000{,}000{,}000{,}000{,}000$), represents the length of a DNA sequence. The

second number, m ($1 \leq m \leq 2{,}000{,}000{,}000$), is the modulus for the operation. The input will be terminated by a line with two 0's.

Output

For each input set, print a single line containing one integer, specifying the number of valid DNA sequences modulo m. There should be no blank lines between outputs.

Sample Input

```
1 10
2 10
5 10
6 8
0 0
```

Sample Output

```
2
3
3
5
```