Modified Fibonacci

Problem Description

As we know in Fibonacci series every number after the first two is the sum of the two preceding ones.

Instead of adding two preceding numbers, multiply them and print the result modulo 10^9+7.

Since this is easy, let’'s make it bit difficult. Let'’s say there are K numbers to begin with.

You have to find nth number, where nth number will be product of k previous numbers modulo 10^9+7.

Constraints

1<=t<=10

1<=n<=10^6

1<=k<=10

1<=k[i]<=100

Input Format

First line contains T number of test case,

In each test case

First line contains two integers n, k delimited by space

Second line contains k integers delimited by space

Output

T lines, each line contains modified Fibonacci number modulo 109+7

Explanation

Example 1

Input

1

4 3

1 2 3

Output

6

Explanation

4th modified Fibonacci number will be 1*2*3=6

Example 2

Input

1

10 3

1 2 3

Output

845114970

Explanation

4th , 5th , 6th modified Fibonacci numbers are 6 , 36 , 648 respectively

Similarly 10th modified Fibonacci number will be 845114970

Number Gravity

Problem Description

Prior to discovery of laws of gravitation, a brave young scientist attempted decoding the how two natural bodies "move" or "accelerate" towards each other. Later Newton, formulated the laws and the equations. Your task is to compute some metrics when two bodies move towards each other with postulates that the young scientist hypothesized.

Newton's law of universal gravitation from classical physics states that the gravitational force between two masses is proportional to the product of the masses and inversely proportional to the square of the distance between them. Imagine a scenario where a similar law applies to two specified natural numbers (positive integers) that "move" or "accelerate" towards each other on the number line. Let us explain with an example: Let us say we start with the following two different natural numbers:

n1 = 11

n2 = 22

n2 > n1

We will assume that the two numbers are initially at rest at their respective positions on the number line at positions L1 = 11 and L2 = 22 units from zero (the origin).

We calculate the initial "force" between them as a constant 10 times their product divided by the square of their difference: F = 10*11*22/(22-11)**2 = 20 units.

The initial "acceleration" of n1 towards n2 would be a1 = F/n1 = 1.82 unit while the initial "acceleration" of n2 towards n1 would be a2 = F/n2 = 0.91 unit.

With initial velocities u1 and u2 being zero,

the "distance" moved by n1 and n2 in unit time would be

s1 = 0 + a1/2 = 1.82/2 = 0.91 = 1 unit (rounded using HALF_UP semantics)
s2 = 0 + a2/2 = 0.91/2 = 0.45 = 0 unit (rounded using HALF_UP semantics).

The new velocities after unit time would be

v1 = u1 + a1 = 0 + 1.82 = 1.82 units and

v2 = v2 + a2 = 0 + 0.91 = 0.91 unit.

Hence after a unit of time, n1 would be at location 12 with a velocity of 1.82 units and n2 would be near location 22 itself with a velocity of 0.91 units. We can proceed in a similar way i.e.

Note the existing positions and velocities,

Calculate the revised force, accelerations, new positions, new velocities

Track the movements of the two numbers as follows:

t	l1	l2	l2-l1	F	u1	u2	a1	a2	s1	s2	v1	v2
0	11 (n1)	22 (n2)	11	20.00	0.00	0.00	1.82	0.91	0.91	0.45	1.82	0.91
1	12	22	10	24.20	1.82	0.91	2.20	1.10	2.92	1.46	4.02	2.01
2	15	21	6	67.22	4.02	2.01	6.11	3.06	7.07	3.54	10.13	5.06
3	22	17	-5	96.80	10.13	5.06	8.80	4.40	14.53	7.26	18.93	9.46

Hence the two numbers n1 and n2 "meet" on the number line between time units 2 and 3 and the location could be taken as the average of positions (15 + 22 + 21 + 17)/4 = 18.75. The relevant equations for each unit time slice (delta t = 1) are: Equation Set 1: F = 10*n1*n2/(l2-l1)**2 a1 = F/n1; a2 = F/n2 s1 = u1 + a1/2; s2 = u2 + a2/2 v1 = u1 + a1; v2 = u2 + a2 If it is found that the average of the last four positions exceeds the position of "crossing", then we could calculate intermediate positions by slicing the last time slice into two and using the following equations (for a half unit time slice, delta t = 1/2): Equation Set 2: F = 10*n1*n2/(l2-l1)**2 a1 = F/n1; a2 = F/n2 s1 = u1/2 + a1/8; s2 = u2/2 + a2/8 v1 = u1 + a1/2; v2 = u2 + a2/2 (Note: It may be necessary to calculate intermediate positions by slicing again, delta t = 1/4 or 3/4) and use the appropriate motion equations.) Consider n1 = 11 and n2 = 40:

t	l1	l2	l2 - l1	F	u1	u2	a1	a2	s1	s2	v1	v2
0	11 (n1)	40 (n2)	29	5.23	0.00	0.00	0.48	0.13	0.24	0.07	0.48	0.13
1	11	40	29	5.23	0.48	0.13	0.48	0.13	0.71	0.20	0.95	0.26
2	12	40	28	5.61	0.95	0.26	0.51	0.14	1.21	0.33	1.46	0.40
3	13	40	27	6.04	1.46	0.40	0.55	0.15	1.74	0.48	2.01	0.55
4	15	40	25	7.04	2.01	0.55	0.64	0.18	2.33	0.64	2.65	0.73
5	17	39	22	9.09	2.65	0.73	0.83	0.23	3.06	0.84	3.48	0.96
6	20	38	18	13.58	3.48	0.96	1.23	0.34	4.09	1.13	4.71	1.30
7	24	37	13	26.04	4.71	1.30	2.37	0.65	5.89	1.62	7.08	1.95
8	30	35	5	176.00	7.08	1.95	16.00	4.40	15.08	4.15	23.08	6.35

9	45	31	-14	22.45	23.08	6.35	2.04	0.56	24.10	6.63	25.12	6.91

We find that the average of positions (30+35+45+31)/4 = 35.25 exceeds 35, the last position of n2. Hence to obtain the meeting point more accurately, we re-calculate the parameters at t=8 and t = 8.5 as follows:

8.5	36	33	-3	488.89	15.08	4.15	44.44	12.22	37.30	10.26	59.52	16.37
8	30	35	5	176.00	7.08	1.95	16.00	4.40	5.54	1.52	15.08	4.15

Hence we can surmise that the two numbers 11 and 40 meet at the location (30 + 36 + 35 + 33)/4 = 33.50. Write a program to accept two natural numbers, carry out the calculations as above and output their meeting location (up to 2 decimal places of accuracy using Rounding down method) under the influence of "gravity".

Constraints

1< = L1 < L2 <= 50

Input Format

First Line contains two Integer Numbers L1 L2 delimited by space

Output

Meeting Point up to 2 decimal places of accuracy using Rounding down method.

Explanation

Example 1

Input

11 22

Output

18.75

Example 2

Input

11 40

Output

33.50

3

Self Referential Sentence

Problem Description

Self referential sentences are fun to play with. Consider the following:

1. This sentence contains three a's and two r's.

2. This sentence contains three a's and three r's.

3. This sentence contains one b, one m and one r.

Each of the above sentences refers to itself and is true: e.g. the letter "a" in the first sentence does occur three times (once each in "cont**a**ins", "**a**'s" and "**a**nd") while "r" occurs two times (once each in "th**r**ee" and "**r**'s").

Here's the problem: Accept one or more alphabets (separated by single space) and form valid self referential sentences in the above style.

Our Test Cases do not contain the instances where It become Impossible to form Self Referential Sentences. It is guaranteed that one Self Referential sentence will be possible for the given Inputs.

Constraints

1<= non-repeated letters <= 5

Input Format

Space separated non-repeated lowercase alphabets.

Output

Starting with "This sentence contains..." followed by the spelled letters in the same order as the input.

Multiple occurrences of a letter should be tagged with 's while a single occurrence should not.

Use of conjunction “and” is permitted exactly once in the sentence. It’s position should be before last spelled letter

Use of “,” should be made when there are more than two spelled letter.

If multiple such sentences are possible, identify all of them, sort them lexicographically and separated by "OR" with entire output terminated by a period.

Explanation

Example 1

Input

a

Output

This sentence contains two a's.

Example 2

Input

a r

Output

This sentence contains three a's and three r's OR This sentence contains three a's and two r's.

Dice and Buckets

Problem Description

We have a set of 6 dice and 49 buckets. Each bucket has the same volume. Each dice has 6 faces numbered with a digit between 1 and 9(both inclusive). It is not necessary to know which dice has which numbers. The numbers on a dice might be repeated. The buckets are numbered from 6 to 54. The 6 dice are rolled together at a time for 100 times to get the dice roll data ‘x’ given below. The buckets have a volume of ‘m’ units. After, the dice are rolled, the number on each of the visible face is summed up to a value, say ‘n’, and n units of liquid are added to the nth bucket. After a bucket is filled to ‘m’ units, the rest of the liquid added overflows out of the bucket.

The data of 100 rolls is given below:

Dice Roll Data(x)= [[4, 6, 3, 8, 1, 1], [6, 3, 2, 6, 4, 9], [3, 2, 8, 9, 4, 5], [4, 3, 1, 2, 7, 8], [6, 5, 9, 7, 1, 7], [5, 7, 4, 4, 4, 7], [9, 7, 6, 7, 2, 9], [4, 9, 3, 2, 5, 9], [7, 3, 2, 5, 8, 8], [3, 3, 1, 7, 4, 9], [5, 7, 3, 9, 9, 4], [9, 6, 5, 4, 9, 5], [1, 7, 8, 1, 1, 1], [9, 6, 3, 6, 8, 4], [3, 5, 2, 6, 5, 5], [4, 2, 5, 2, 2, 5], [3, 2, 2, 1, 5, 1], [1, 2, 7, 5, 9, 5], [5, 6, 2, 6, 6, 9], [3, 8, 3, 4, 8, 1], [7, 5, 7, 1, 9, 9], [7, 1, 2, 8, 3, 8], [7, 6, 3, 1, 3, 6], [2, 2, 5, 7, 2, 8], [7, 3, 5, 3, 4, 8], [9, 3, 5, 5, 8, 4], [8, 1, 3, 2, 1, 3], [3, 2, 5, 9, 4, 7], [8, 5, 8, 5, 2, 8], [7, 7, 4, 6, 5, 7], [9, 4, 4, 3, 6, 9], [2, 5, 2, 2, 1, 4], [3, 7, 7, 9, 7, 5], [7, 2, 7, 6, 6, 1], [1, 7, 3, 9, 9, 7], [7, 1, 1, 1, 4, 8], [6, 9, 6, 6, 4, 6], [9, 8, 9, 2, 5, 8], [8, 8, 8, 9, 6, 8], [8, 1, 9, 2, 4, 6], [5, 8, 2, 2, 4, 9], [8, 9, 8, 3, 7, 1], [7, 1, 5, 1, 6, 8], [8, 9, 9, 4, 8, 1], [9, 3, 4, 4, 2, 6], [2, 6, 8, 9, 6, 2], [5, 9, 3, 1, 9, 4], [9, 2, 7, 4, 7, 3], [1, 7, 8, 2, 7, 8], [8, 4, 4, 4, 6, 2], [3, 4, 8, 6, 8, 4], [6, 6, 9, 3, 9, 6], [1, 1, 4, 6, 6, 2], [6, 4, 4, 5, 7, 4], [4, 2, 1, 2, 3, 1], [8, 9, 5, 3, 4, 6], [3, 3, 5, 8, 2, 5], [9, 7, 3, 8, 7, 3], [9, 3, 6, 5, 7, 1], [7, 7, 2, 9, 1, 9], [6, 8, 7, 4, 5, 4], [5, 8, 5, 4, 2, 4], [7, 3, 4, 5, 9, 4], [4, 4, 9, 1, 8, 3], [5, 3, 9, 1, 3, 1], [1, 9, 6, 1, 1, 1], [1, 3, 1, 2, 9, 7], [5, 7, 2, 9, 7, 6], [2, 5, 1, 8, 9, 5], [5, 1, 8, 4, 1, 4], [4, 5, 1, 2, 5, 3], [1, 6, 3, 4, 3, 6], [5, 8, 4, 5, 7, 5], [7, 5, 5, 7, 2, 4], [6, 7, 1, 6, 8, 7], [3, 3, 6, 3, 1, 6], [7, 6, 8, 9, 6, 9], [2, 4, 5, 7, 1, 4], [1, 8, 4, 8, 2, 5], [7, 5, 6, 9, 9, 8], [8, 8, 1, 8, 9, 5], [5, 2, 8, 4, 5, 1], [9, 8, 1, 6, 8, 9], [1, 8, 9, 3, 6, 3], [3, 4, 5, 6, 2, 8], [8, 5, 9, 5, 3, 2], [7, 6, 5, 9, 6, 5], [7, 2, 7, 8, 7, 8], [5, 7, 6, 6, 9, 2], [5, 7, 2, 6, 2, 5], [4, 1, 6, 9, 4, 9], [1, 3, 4, 7, 4, 5], [5, 8, 9, 7, 4, 1], [1, 3, 8, 4, 8, 9], [2, 8, 7, 4, 8, 3], [7, 3, 5, 2, 2, 8], [1, 6, 5, 4, 1, 1], [1, 2, 6, 9, 6, 9], [8, 5, 2, 7, 3, 8], [2, 1, 2, 4, 1, 2]]

Your task is, given inputs x and m, print the following:

1. The total capacity consumed by buckets.

2. The total volume of liquid over flown from the buckets.

3. The total volume of empty space remaining in the buckets.

Constraints

1. 0 < m <=50

2. m is an integer

Input Format

First line contains an integer depicting volume of the bucket.

Output

1. The total capacity consumed by buckets.

2. The total volume of liquid over flown from the buckets.

3. The total volume of empty space remaining in the buckets.

Explanation

Example 1

Input

10

Output

300

2731

190

Example 2

Input

1

Output

30

3001

19

Cross Words

Problem Description

A crossword puzzle is a square grid with black and blank squares, containing clue numbers (according to a set of rules) on some of the squares. The puzzle is solved by obtaining the solutions to a set of clues corresponding to the clue numbers.

The solved puzzle has one letter in each of the blank square, which represent a sequence of letters (consisting of one or more words in English or occasionally other languages) running along the rows (called "Across", or "A") or along the columns (called "Down" or "D"). Each numbered square is the beginning of an Across solution or a Down solution. Some of the across and down solutions will intersect at a blank square, and if the solutions are consistent, both of them will have the same letter at the intersecting square.

In this problem, you will be given the specifications of the grid, and the solutions in some random order. The problem is to number the grid appropriately, and associate the answers consistently with the clue numbers on the grid, both as Across solutions and as Down solutions, so that the intersecting blank squares have the same letter in both solutions.

Rules for Clue Numbering

The clue numbers are given sequentially going row wise (Row 1 first, and then row2 and so on)

Only blank squares are given a clue number

A blank square is given a clue number if either of the following conditions exist (only one number is given even if both the conditions are satisfied)

It has a blank square to its right, and it has no blank square to its left (it has a black square to its left, or it is in the first column). This is the beginning of an Across solution with that number

It has a blank square below it, and no blank square above it (it has a black square above it or it is in the first row). This is the beginning of a Down solution with that number

Constraints

5<=N<=15

5<=M<=50

Input Format

The input consists of two parts, the grid part and the solution part

The first line of the grid part consists of a number, N, the size of the grid (the overall grid is N x N) squares. The next N lines correspond to the N rows of the grid. Each line is comma separated, and has number of pairs of numbers, the first giving the position (column) of the beginning of a black square block, and the next giving the length of the block. If there are no black squares in a row, the pair “0,0” will be specified. For example, if a line contains “2,3,7,1,14,2”, columns 2,3,4 (a block of 3 starting with 2), 7 (a block of 1 starting with 7) and 14,15 (a block of 2 starting with 14) are black in the corresponding row.

The solution part of the input appears after the grid part. The first line of the solution part contains M, the number of solutions. The M subsequent lines consist of a sequence of letters corresponding to a solution for one of the Across and Down clues. All solutions will be in upper case (Capital letters)

Output

The output is a set of M comma separated lines. Each line corresponds to a solution, and consists of three parts, the clue number, the letter A or D (corresponding to Across or Down) and the solution in to that clue (in upper case)

The output must be in increasing clue number order. Ifa clue number has both an Across and a Down solution, they must come in separate lines, with the Across solution coming before the Down solution.

Explanation

Example 1

Input

5
5,1

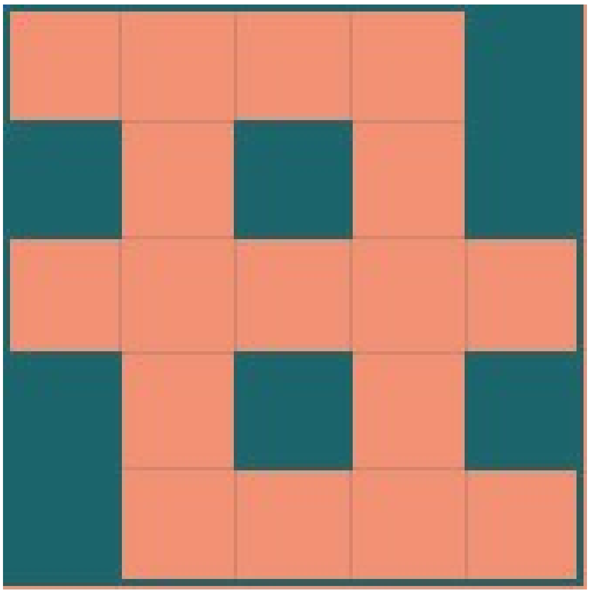
1,1,3,1,5,1
0,0
1,1,3,1,5,1
1,1
5
EVEN
ACNE
CALVE
PLEAS
EVADE

Output

1,A,ACNE
2,D,CALVE
3,D,EVADE
4,A,PLEAS
5,A,EVEN

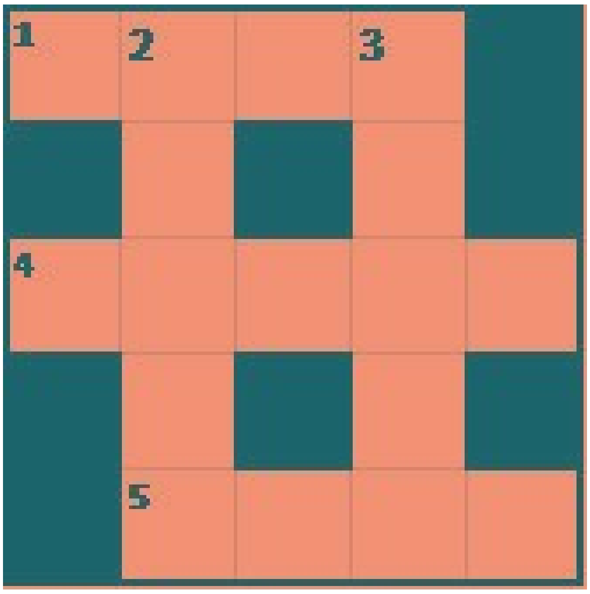
Explanation

N is 5, and the disposition of the black squares are given in the next 5 (N) lines. The grid looks like this

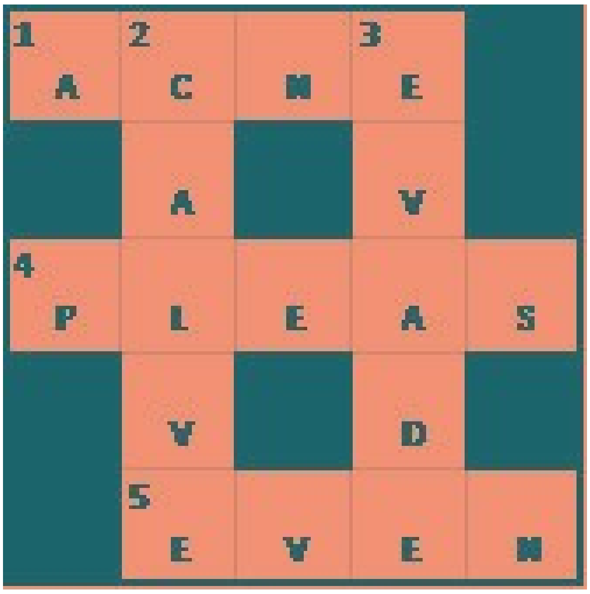


M=5, and there are 5 (M) solutions.

If the grid is numbered according to the rules, the numbered grid loos like this. Note that row 3 has no blanks, and the input line says “0,0”



The solutions are fitted to the grid so that they are consistent, and the result is shown below. Note that this is consistent, because the letter at each intersecting blank square in the Across solution and the Down solution.



Based on this the output is given in clue number order. 1 Across is ACNE, and hence the first line of the output is 1,A,ACNE. The same logic gives all the remaining solutions.

Example 2

Input

5
1,1
1,1,3,2
0,0
1,1,3,2
0,0
5
ASIAN

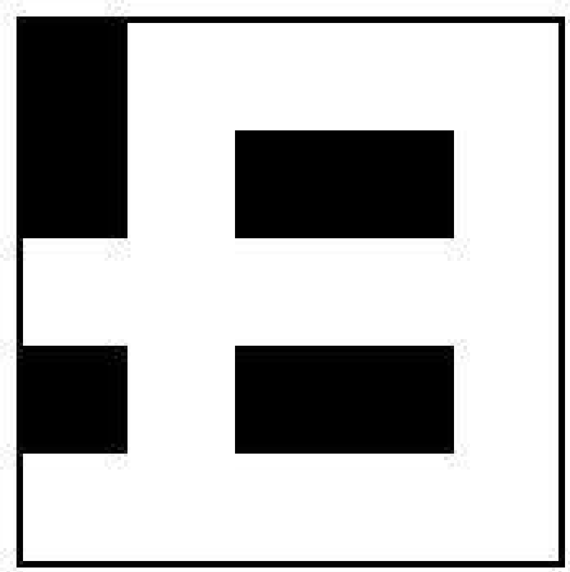
RISEN
FEAR
CLAWS
FALLS

Output

1,A,FEAR
1,D,FALLS
2,D,RISEN
3,A,CLAWS
4,A,ASIAN

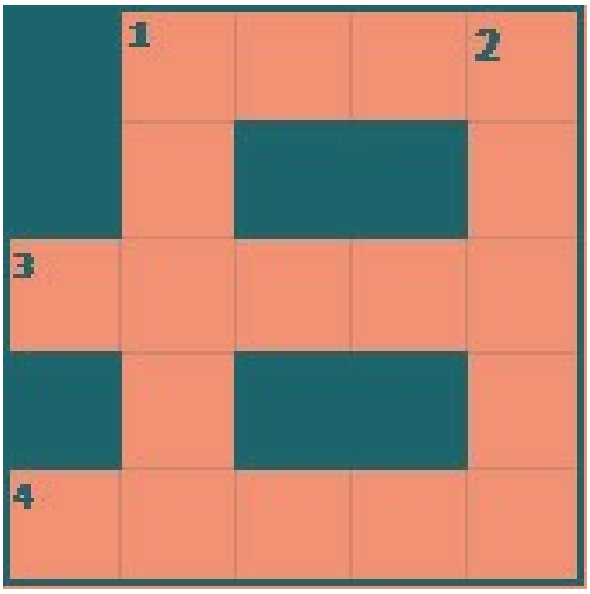
Explanation

N=5, and the grid looks like this

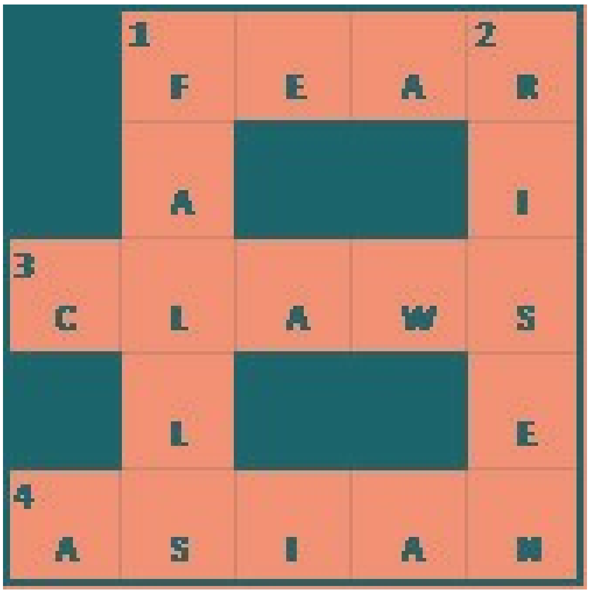


M=5, and the 5 solutions are given

The numbered grid looks like this



The consistently populated grid (with the solutions) look like this



The output can be easily given from this. Note that clue number 1 has both an Across solution (FEAR) and a DOWN solution (FALLS). The Across solution must precede the Down solution in the output.

Skateboard

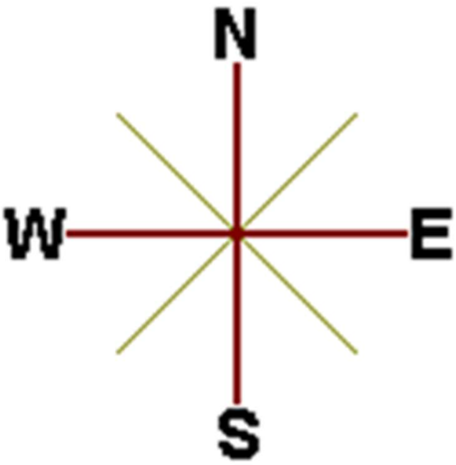
Problem Description

The amusement park at Patagonia has introduced a new skateboard competition. The skating surface is a grid of N x N squares. Most squares are so constructed with slopes that it is possible to direct the skateboard in any of up to three directions of the possible four (North ,East, South or West, represented by the letters N, E, S and W respectively). Some squares however have a deep drop from the adjacent square from which it is impossible to go to any adjacent square. These are represented by D (for Drop) in that square. The objective is to maneuver the skateboard to reach the South East corner of the grid, marked F.

Each contestant is given a map of the grid, which shows where the Drop squares are (marked D), where the Final destination is (marked F), and, for each other square, the directions it is possible to maneuver the skateboard in that square.

The contestant draws lots to determine which of the squares on the boundaries of the grid on the North or the West of the grid (the top or the left in the diagram) he or she should start in. Then, using a map of the grid, he or she needs to try to reach the South East corner destination by maneuvering the skateboard.

ES	ES	SE	ES	ES	S
SE	ES	SE	ES	ES	S
ES	ES	SE	ES	SE	S
ES	SE	ES	SE	E	D
SE	ES	D	WSE	NES	NS
E	E	NE	E	E	F



In some cases, it is impossible to reach the destination. For example, in the diagram above, if one starts at the North East corner (top right in the diagram), the only way is to go is South, until the Drop square is reached (three squares South), and the contestant is stuck there.

A contestant asks you to figure out the number of squares at the North or West boundary (top or left boundary in the map) from which it is feasible to reach the destination.

Constraints

5<=N<=50

Input Format

The first line of the input is a positive integer N, which is the number of squares in each side of the grid.

The next N lines have a N strings of characters representing the contents of the map for that corresponding row. Each string may be F, representing the Final destination, D, representing a drop square, or a set of up to three of the possible four directions (N,E,S,W) in some random order. These represent the directions in which the contestant can maneuver the skateboard when in that square.

Output

The output is one line with the number of North or West border squares from which there is a safe way to maneuver the skateboard to the final destination.

Explanation

Example 1

Input

6

ES,ES,SE,ES,ES,S

SE,ES,SE,ES,ES,S

ES,ES,SE,ES,SE,S

ES,SE,ES,SE,E,D

SE,ES,D,WSE,NES,NS

E,E,NE,E,E,F

Output

9

Explanation

N =6, and the size of the grid is 6x6. The map of the grid is as below.

ES	ES	SE	ES	ES	S
SE	ES	SE	ES	ES	S
ES	ES	SE	ES	SE	S
ES	SE	ES	SE	E	D
SE	ES	D	WSE	NES	NS
E	E	NE	E	E	F

There are many ways to maneuver the skateboard. For example, if the contestant starts from the North West corner (top left in the map) and goes East all the way until he reaches the top right corner in the map, and then goes South, he will reach a Drop square. But if he goes South all the way from the same square until he reaches the bottom left square on the map, and keeps going East from there, the Final destination will be reached. Hence the top left square (1,1) is safe.

Similarly, from the square (1,5), all the paths lead to a drop square., The other 9 North and West border squares have ways skateboard to get to the final destination. The output is 9

Example 2

Input

5

ES,SE,ES,SE,S

S,EWS,SE,E,S

E,D,SEW,NSE,S

NE,N,SEW,D,W

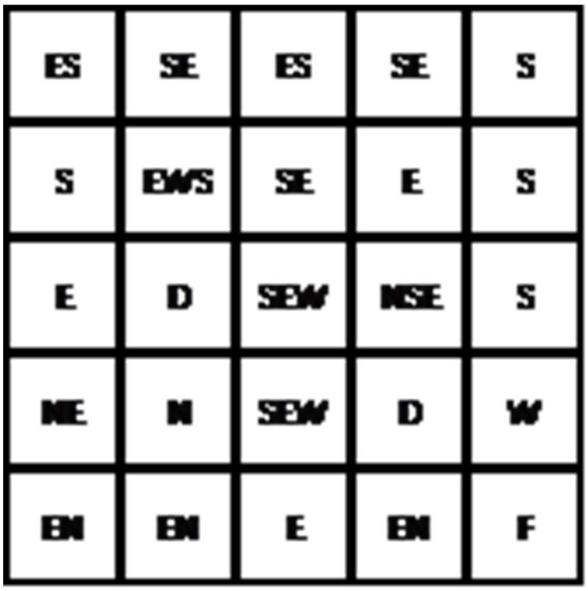
EN,EN,E,EN,F

Output

4

Explanation

N=5, and the grid is 5 x 5 squares. The map of the grid looks like this.



ES	SE	ES	SE	S
S	EWS	SE	E	S
E	D	SEW	NSE	S
NE	N	SEW	D	W
EN	EN	E	EN	F

It can be seen that from squares (1,4) and (1,5), there is no way to maneuver the skateboard to the Final destination, and hence are not safe starting points.. Similarly, squares (2,1),(3,1), and (4,1) are not safe starting points. The only safe starting points on the North and West borders are (1,1),(1,2),(1,3),(5,1). Hence the output is 4

ESESSEEESESS

SEESSEEESESS

ESESSEEESSES

ESSEESSEED

SEESDWSENESNS

EENEEEF

ESSEESSES

SEWSSEES

EDSEWNSES

NENSEWDW

ENENEENF