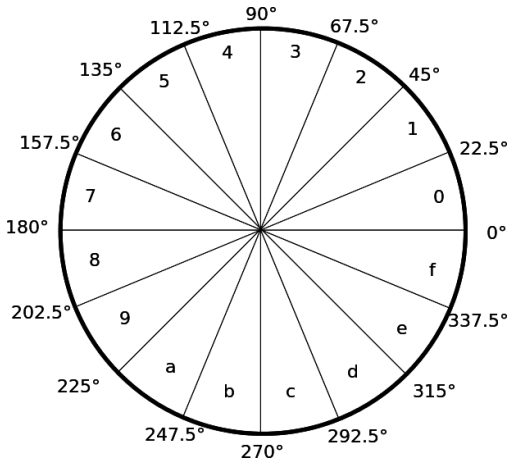
Problem A
Mars Message
Problem ID: mars

In the film *The Martian*, astronaut Mark Watney is stuck on Mars, and communicates with earth by rotating a camera on an old space prod.

The prod points at a hexidecimal digit, i.e. `0-9` or `a-f`, and Mark creates letters and other characters by converting a pair of hexadecimal digits according to ASCII encoding. To determine which digit the prod is pointing at Mark has divided a circle of 360 degrees into 16 parts of 22.5 degrees each.



For example, the pair of angles $(100{*}\circ, 30°)$ represents the hexadecimal number 41, which in decimal is 65. According to ASCII[1] that is the character `A`.

Mark is getting tired of writing down the digits manually, and he asks you to help him decode the messages he receives. You will be given the sequence of angles the prod points at and you're expected to print the message.

Input

The first line consists of an integer $2 \leq N \leq 50\,000$, the number of angles. N will be even.

The following N lines contain angles, one angle per line. An angle is given in degrees as a float in the range $[0, 360)$. The prod will never point at an angle which is exactly inbetween two hexadecial digits.

When decoding to ASCII only characters with decimal values `32-126` occurs.

Output

Print one line with the decoded message.

Grading

Your solution will be tested on several groups of test cases. To get points for a group you need to pass all the tests of that group.

Group	Points	Constraints
1	46	all angles are integers
2	54	

[1] https://en.wikipedia.org/wiki/ASCII#ASCII_printable_characters

Sample Input 1	Sample Output 1
54 91 118 157 23 167 66 158 100 157 189 47 21 172 98 139 353 59 10 139 314 146 23 160 48 162 70 63 316 55 21 94 95 144 353 53 11 166 203 147 339 158 115 46 3 157 83 149 339 160 18 177 218 69 339	Earth to mars. Do you copy?

Problem B
Free food
Problem ID: gratismat

Hsara loves free food.

Tonight Hsara is going out for a dinner at a restaurant with his organisation and Hsara is thinking about the chances of someone else paying for the dinner. Exactly one person will pay for the entire dinner (everyone's food), but not any person. It is only allowed to pay for dinner if no supperior (possibly indirect, for example, your boss' boss) in the organisation is present at the table.

The organisation consists of N people numbered from 1 to N. Each person, except for the CEO, has exactly one boss. Given each person's boss and a list of people present at the table, determine how many people are able to pay for the dinner.

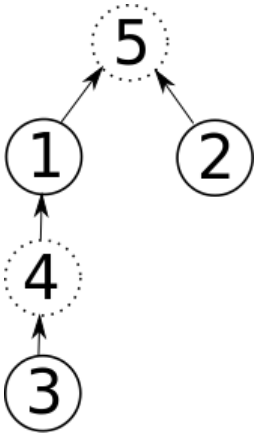


Figure B.1: Illustration of the organisation in the first sample.

Input

The first line consists of two integers, N - the number of people in the organisation and $1 \leq M \leq N$ - the number of people at the table.

The second line consists of N integers. The i:th integer is the boss of person i. 0 means that person i is the CEO. There will be exactly one CEO.

The third line consists of M integers indicating the people present at the table.

Output

Print an integer - the number of people at the table who would be able to pay for the dinner.

Grading

Your solution will be tested on several groups of test cases. To get points for a group you need to pass all the tests of that group.

Group	Points	Constraints
1	49	$2 \leq N \leq 2\,000$
2	51	$2 \leq N \leq 100\,000$.

Explanation

In the first sample person 1 and person 2 can pay. Person 3 can't pay because person 1 is at the table.

In the second sample person 5 is the CEO and at the table. Therefore only person 5 can pay.

Sample Input 1	Sample Output 1
5 3 5 5 4 1 0 1 2 3	2

Sample Input 2	Sample Output 2
6 5 5 4 4 5 0 1 5 1 3 2 4	1

Problem C
Lab planning
Problem ID: labbplanering

Ah, university. A bastion of knowledge, the Mecca of education, and the core of bureaucracy.

Wille and Kaski are currently waiting in a computer lab to present a lab in a course. In total, there are N groups waiting to present their lab. Group i is going to present m_i different labs, where lab j takes $a_{i,j}$ minutes.

To be fair, all groups need to wait a really long time to present. There is only a single teacher taking presentations, and letting each group present all their labs in a single go would mean that they don't have to wait unnecessarily for other groups! Therefore, the teacher takes presentations in a seemingly arbitrary order...

Kashi has had enough. She wants to go home and play Pokémon, and is complaining about the inefficiencies in the presentation scheme. To demonstrate how inefficient the system is, she'll compute the longest possible total waiting time, and show just how close the teacher's system is to this value.

Assume that a group starts presenting their first lab at time a, and completes their final lab at time b. Then its waiting time is $b - a$. The total waiting time is the sum of the waiting time over all groups.

The order in which a group must present its labs must be the same as the order they are given in the input.

Input

The first line contains a single integer $N \geq 1$.

Then follow N lines, each containing first the number $m_i \geq 1$, and then m_i integers $a_{i,j}$, all between 1 och 60.

Output

You should output a single number: the longest possible total waiting time for the students over all possible presentation orders.

Grading

Your solution will be tested on a number of groups. To get the points for a group, you must pass all test cases in the group.

The sum of all the m_i will be at most $5\,000$ in the groups 1 through 4.

Group	Score	Constraings
1	30	the sum of all m_i is at most 10
2	17	all labs have the same length
3	13	all groups have exactly 2 labs to present
4	18	
5	22	the sum of all the m_i is at most $100\,000$

Explanation

In the given sample, three groups need to present their labs: the first group has labs taking 5 and 15 minutes, the second group takes 10 and 20 minutes, and the third group has a single lab taking an entire hour.

If we choose the order $5, 10, 60, 20, 15$, the third group will wait only for 60 minutes. The second group will, besides their own labs, have to wait for the lab of the third group, and therefore take $10 + 60 + 20 = 90$ minutes. The first group will instead take $5 + 10 + 60 + 20 + 15 = 110$ minutes.

The sum of this is $60 + 90 + 110 = 260$ minutes.

Note that the order 10, 15, 60, 20, 5 would have resulted in a worse time of 265, but this is not allowed: group 1 may not present their second lab before their first.

Sample Input 1	**Sample Output 1**
3 2 5 15 2 10 20 1 60	260

Problem D
Quiz
Problem ID: fragetavling

In the quiz ProgrammeringsQuiz there are N questions in total, distributed over M different categories (for example, algorithm theory, compiler construction or Sven knowledge).

The questions are worth different amounts of points. Additionally, you will get a bonus B if you answer all questions in a certain category. Simone has participated in Programmeringsolympiaden since 8th grade, so she is able to answer all the questions.

Unfortunately, there is a time limit to the quiz. Despite never giving the wrong answer, Simone will only have time to answer K questions. What is the maximum number of points she can achieve?

Input

The first line consists of four integers $1 \leq N \leq 1000$, $1 \leq M \leq N$, $1 \leq K \leq N$, $1 \leq B \leq 100\,000$. The following N lines consist of two integers each: the points given for answering the question (an integer between 1 and 1 000) and which category it belongs to (between 1 and M). Each category will have at least one question.

Output

Print one line containing the maximal possible number of points.

Grading

Your solution will be tested on several groups of test cases. To get points for a group you need to pass all the tests of that group.

Group	Points	Constraints
1	16	$N \leq 100, M = 1$
2	29	$M \leq 5$
3	20	all question give the same number of points
4	35	

Explanation

In the first sample Simone answers both questions from category 1 ($300 + 400 = 700$ points) and the only question in category 2 (200 points). Since these were the only questions in these two categories we get two bonuses, which gives a total of $200 + 700 + 2 \cdot 1000 = 2900$ points.

Sample Input 1	**Sample Output 1**
5 3 3 1000 300 1 400 1 200 2 200 3 300 3	2900

Sample Input 2	Sample Output 2
5 3 3 1 300 1 400 1 200 2 300 3 200 3	1001

Problem E
Vvvvvv
Problem ID: vvvvvv

Platform games is an old video game genre, where you move across platforms hanging in the air. Vvvvvv is a platform game where you cannot jump, but instead are able to switch between regular gravity, which causes you to fall downwards, and anti-gravity, which causes you to fall upwards. There are only three buttons in the game: the gravitational switch (G), the go-left-button (V) and the go-right-button (H).

Your task is to find a sequence of button presses which takes you through a maze-like level with platforms and walls, from a given start position to a given final position. In order to receive full points, the sequence needs to be as short as possible.

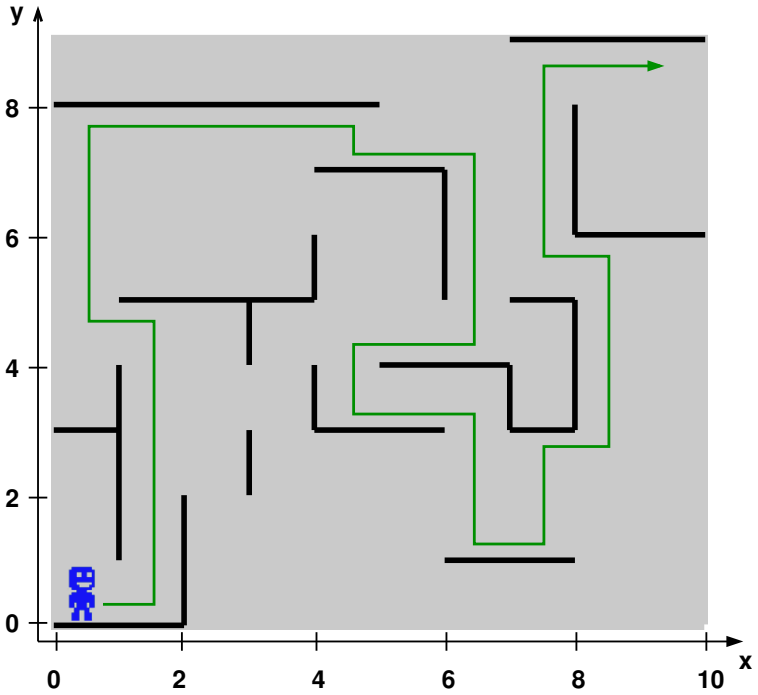


Figure E.1: The path given by the button presses given in the second sample.

Input

The first line of the input consists of three integers W, H, and N: the width of the level, the height of the level, and the number of line segments. Thereafter follows N lines, each with four integers x_1, y_1, x_2, y_2, where $0 \leq x_1, x_2 \leq W$ and $0 \leq y_1, y_2 \leq H$. Each of these lines describe a line segments with end points (x_1, y_1) and (x_2, y_2). Every line segment is either horizontal or vertical, i.e. either $y_1 = y_2$ or $x_1 = x_2$ holds. The integer coordinate system can be said to divide the rectangular level into a grid, where the button V takes you one square to the left and H takes you one square to the right. If you are not standing on a platform after the movement you will fall (upwards or downwards) until you reach another platform. It is not possible to move to the left or right while falling. If you fall or walk out of the level, you lose the game. This also happens if you try to walk through a wall.

The start position is the lower left square, closest to the point (0, 0), with downwards gravity. The final position is the square in the upper right corner, closest to the point (W, H). There will always be a line segment under the start position and above the final position.

Output

If there is a solution, the program should print a string containing a letter for each button press (chosen from G, V and H). The sequence should take you from the start position to the final position, and has to be shorter than 10000 characters. You can arrive to the ending position with gravity in either direction, but you need to have stopped falling (see the third example below).

If there is no solution, the program should print the string `"Inte"` (without the quotation marks).

Scoring

Your solution will be tested on a set of test case groups. In order to receive points for a group, you need to solve all test cases in the group.

Group	Points	Limits	Constraints
1	20	$W, H \leq 10$, $N \leq 25$, there is a solution with at most 10 moves	the sequence does not need to be as short as possible.
2	15	$W, H \leq 20$, $N \leq 200$, there is a solution where you do not need to use V	the sequence does not need to be as short as possible.
3	15	$W, H \leq 50$, $N \leq 2000$	the sequence does not need to be as short as possible.
4	20	$W, H \leq 10$, $N \leq 50$	the sequence needs to be as short as possible.
5	30	$W, H \leq 100$, $N \leq 5000$	the sequence needs to be as short as possible.

Sample Input 1

```
4 5 9
2 5 4 5
4 5 4 1
4 4 3 4
3 0 4 0
0 0 2 0
0 3 2 3
2 2 2 3
1 1 1 2
1 1 3 1
```

Sample Output 1

```
HGHHVH
```

Sample Input 2

```
10 9 22
0 0 2 0
2 0 2 2
1 1 1 4
0 3 1 3
1 5 4 5
3 2 3 3
3 4 3 5
4 5 4 6
4 4 4 3
4 3 6 3
6 1 8 1
5 4 7 4
7 4 7 3
7 3 8 3
8 3 8 5
8 5 7 5
6 5 6 7
6 7 4 7
0 8 5 8
8 8 8 6
8 6 10 6
7 9 10 9
```

Sample Output 2

```
HGVHHHHGHHVVHHHGHVHH
```

Sample Input 3

```
2 1 2
0 0 1 0
1 1 2 1
```

Sample Output 3

```
Inte
```

Problem F
Candy
Problem ID: godis

It is Saturday and Ann Britt-Caroline is going to buy candy. She has identified several different bags of candy she is considering buying.

Each bag contains a number of pieces of candy of different types. There are 10 types of normal candy (these are numbered $1...10$), and 10 types of anti-candy (numbered $-1... -10$). It just so happens that candy of type n and type $-n$ do not go well together - they are annihilated if they come in contact with each other. Other than that anti-candy tastes just the same as normal candy.

When Ann Britt-Caroline has bought the bags of candy she mixes them in a big bowl such that all pairs of candy/anti-candy is annihilated (she mixes thoroughly). How many pieces of candy can Ann Britt-Caroline have left (after all pairs of candy/ant-candy is annihilated), if she selects her bags of candy optimally? Note that she can only buy one bag of each type. Ignore any money related issues - her parents will pay.

Input

The first line in the input consist of an integer $1 \leq N \leq 1000$ - the number of bags of candy.

The following n lines describe each bag of candy. Each line starts with an integer $1 \leq k \leq 10$, specifying the number of different types of candy in the bag. Then follow k pairs of integers $s\ n$, which means that there are n pieces of candy type s. Each type of candy is mentioned at most once per bag, and types s and $-s$ can not be in the same bag.

For all n, $1 \leq n \leq 1000$.

Output

Print an integer: the largest amount of candy Ann Britt-Caroline can have in the end.

Grading

Your solution will be tested on several groups of test cases. To get points for a group you need to pass all the tests of that group.

Group	Points	Constraints
1	25	$N \leq 15$
2	25	$N \leq 1000$. Each bag will contain only one type of candy.
3	50	$N \leq 1000$

Sample Input 1

```
3
1 1 3
2 -1 1 -2 5
2 2 2 -3 1
```

Sample Output 1

```
7
```

Problem G
Solitaire chess
Problem ID: solitarschack

Mårten is good at chess. A little too good, according to some. For instance Johan. Mårten often challenges Johan at chess, leading to Johan losing. Instead of admitting defeat Johan does what he always does when he loses at something; he changes the rules.

This time he has invented a new sort of single-player chess which he calles *Solitaire chess*. The game is played on a 6×6 board with pieces. The pieces are in three different colors – bronze, silver, gold – and eight different types: 1, 2, 3, 4, torn (en: rook), lopare (bishop), dam (queen) and springare (knight). Initially the board is entirely filled with bronze-colored pieces.

Each move of the game consists of Johan removing one piece from the board. If the piece which has been removed is a bronze piece, it is replaced by a new silver piece of random type. If it is a silver piece, it is replaced by a random bronze piece. If the piece Johan removes is a gold piece, its tile is left empty. The basic goal of the game is to perform as many moves as possible.

In the first move, Johan gets to choose freely which piece to remove. For every other move the pieces that can be removed is limited by a number of rules, which depend on the type and position of the last removed piece. Johan may continue making moves as long as he wishes and is able.

Let (r, c) be the row and column of the piece that was last removed, and (r', c') row and column of a piece you want to remove next. The rows and columns are 1-indexed. Whether this move is allowed is determined by the rule below that matches the type of the last removed piece:

1 the next piece must be exactly 1 step away (horizontally, vertically or diagonally) from this piece. More formally one of the following must hold:

$$|r - r'| = 1, |c - c'| = 0$$
$$|r - r'| = 0, |c - c'| = 1$$
$$|r - r'| = 1, |c - c'| = 1$$

2 the next piece must be exactly 2 step away (horizontally, vertically or diagonally) from this piece. More formally one of the following must hold:

$$|r - r'| = 2, |c - c'| = 0$$
$$|r - r'| = 0, |c - c'| = 2$$
$$|r - r'| = 2, |c - c'| = 2$$

3 the next piece must be exactly 3 step away (horizontally, vertically or diagonally) from this piece. More formally one of the following must hold:

$$|r - r'| = 3, |c - c'| = 0$$
$$|r - r'| = 0, |c - c'| = 3$$
$$|r - r'| = 3, |c - c'| = 3$$

4 the next piece must be exactly 4 step away (horizontally, vertically or diagonally) from this piece. More formally one of the following must hold:

$$|r - r'| = 4, |c - c'| = 0$$
$$|r - r'| = 0, |c - c'| = 4$$
$$|r - r'| = 4, |c - c'| = 4$$

`torn` the next piece must be either straight up, down, to the left or to the right of this piece, moved away all the way to the edge. More formally one of the following must hold:

$$r' \in \{1, 6\}, |c - c'| = 0$$

$$|r - r'| = 0, c' \in \{1, 6\}$$

`lopare` the next piece must be along the sides of the board, on the same diagonal as this piece. More formally one of the following must hold:

$$|r - r'| = |c - c'|, r' \in \{1, 6\}$$

$$|r - r'| = |c - c'|, c' \in \{1, 6\}$$

`dam` the next piece must be places as if this piece was either a `torn` or a `lopare`. More formally one of the following must hold:

$$r' \in \{1, 6\}, |c - c'| = 0$$

$$|r - r'| = 0, c' \in \{1, 6\}$$

$$|r - r'| = |c - c'|, r' \in \{1, 6\}$$

$$|r - r'| = |c - c'|, c' \in \{1, 6\}$$

`springare` the next piece must be placed either on the row above/below and two columns to the left/right, or on the column to the left/right and two rows above/below. More formally one of the following must hold:

$$|r - r'| = 2, |c - c'| = 1$$

$$|r - r'| = 1, |c - c'| = 2$$

It is not allowed to have $(r', c') = (r, c)$ (in either of the cases).

The scoring is as follows. For each tile when the game is over, you get the following score depending on tile:

bronze 0 points

silver 1 point

gold 2 points

empty 3 points

You can also get bonus points during the game. The following bonuses exist:

- If you remove $N \geq 2$ pieces in a row of the same type you get $2N$ points.
- If you remove the pieces $1, 2, 3, 4$ in a row in order or reverse order you get 12 points.
- If you remove the pieces $1, 2, 3, 4$ in a row in some other order you get 8 points.
- If you remove the pieces $torn, lopare, dam, springare$ in a row in some order you get 8 points.
- Removing the numbers $1, 2, 3, 4$ in some order is called a *number set*. Removing the pieces $torn, lopare, dam, springare$ in some order is called a *piece set*.

  If you remove $K \geq 2$ number sets and piece sets in *alternating order*, i.e. either number set - piece set - number set - ... or piece set - number set - piece set - ..., you get an additional $8K$ points.

Bonuses of the same sort cannot overlap, and are given greedily. For instance, the sequence of removed pieces `4 1 2 3 4 dam torn springare lopare` would give $8 + 8 = 16$ bonus points: 8 for the sequence `4 1 2 3`, none for the sequence `1 2 3 4` because it overlaps with the earlier one, and 8 for the sequence `dam torn springare lopare`. No additional points for alternative piece sets are given, because we have not been given points for the sequence `1 2 3 4`. The sequence `1 1 2 3 4` gives $4 + 12 = 16$ points: `1 1` and `1 2 3 4` are different types of bonuses, so it is okay for the to overlap.

It turns out to be harder than expected to get a good score in Solitaire chess than beating Mårten at regular chess... Help Johan get as good score as possible in Solitaire chess!

Input

Input starts with 6 lines, each of which contains 6 white-space separated strings. This denote what the original bronze pieces are.

Interaction

This is an interactive problem. To make a move, you print two integers `r c`, the row and column that contains the piece you want to remove.

You should then read one line - the name of the piece that replaces the one you removed. If you just removed a gold piece, the line will contain the work `blank`.

When you don't make to make any more moves, you should print a line `0 0` and terminate the program.

Don't forget to flush standard out when you have printed! In Java this can be done using `System.out.flush()`, in C with `fflush(stdout)`, in C++ with `flush(cout)`, in C# with `Console.Out.Flush()` and in Python with `sys.stdout.flush()`.

Examples

The judge's output is in bold.

1 1 1 1 1 1
1 1 1 1 1 1
1 1 2 1 1 torn
1 1 3 4 1 dam
1 1 1 1 1 1
1 1 1 1 1 1
6 6
dam
5 6
springare
6 6
torn
1 1
lopare
0 0

In this example you first removed the 1 at position $(6, 6)$, then the 1 at $(5, 6)$, then the queen (dam) that had appeared at $(6, 6)$, and finally the 1 at position $(1, 1)$. Thereafter you gave up out of boredom.

Note that the moves you make are based on the sort of piece that was *just removed*, not the one that was just shown. After the removal of the queen you could also have removed a piece at position $(6, 1)$ or $(1, 6)$, though not e.g. position $(4, 1)$ because with queen (and rook, bishop) moves you must move all the way until an edge.

Scoring

Say that on a test case worth P points your program gets X points, and the best program Y points. Then you get $P \times \frac{X}{Y}$ points. During the contest all submissions will be judged with $Y = 200$. After the end of the contest this will be rejudged with the best submissions on each test case.

The test cases are divided into a number of groups.

Group	Points	Constraints
1	4	Only the piece springare appears.
2	4	Only the piece 1 appears.
3	6	Only the pieces 1 and 2 appear.
4	9	Only the pieces torn, lopare, dam appear.
5	24	Only the pieces 1, 2, 3, 4 appear.
6	53	All pieces may appear.