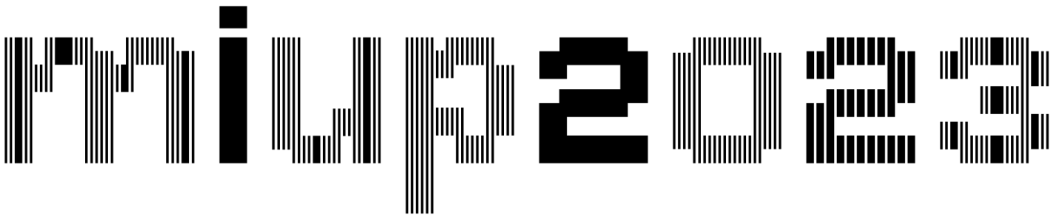


Maratona Inter-Universitária de Programação

mip2023

Maratona Inter-Universitária de Programação

Local Organisation:

Teams from the following Universities:

Coimbra, 21 de Outubro de 2023

CONTENTS

Scientific Committee

- Alexandre Jesus, Universidade de Coimbra
- André Restivo, FEUP / Universidade do Porto
- Fábio Marques, Universidade de Aveiro
- **Filipe Araújo, Universidade de Coimbra (Coordinator)**
- João Neves, Universidade da Beira Interior
- João Dias, Universidade do Algarve
- Luís M. S. Russo, IST / Universidade de Lisboa
- Margarida Mamede, FCT / Universidade Nova de Lisboa
- Mário Pereira, FCT / Universidade Nova de Lisboa
- Pedro Mariano, ISCTE
- Pedro Ribeiro, FCUP / Universidade do Porto
- Rui Mendes, Universidade do Minho
- Simão Melo de Sousa, Universidade do Algarve
- Vasco Pedro, Universidade de Évora

Local Organisation Committee

- Alexandre Jesus, Universidade de Coimbra
- Amílcar Cardoso, Universidade de Coimbra
- Carlos Fonseca, Universidade de Coimbra
- Filipe Araújo, Universidade de Coimbra
- **Luís Paquete, Universidade de Coimbra (Coordinator)**

Languages and Compilers

I. Files `*.c` are assumed to be C code and are compiled with `gcc 11.4.0` with the command

```
gcc -std=gnu17 -O2 file.c -lm
```

and then executed with the command

```
./a.out
```

II. Files `*.cpp` are assumed to be C++ code and are compiled with `g++ 11.4.0` with the command

```
g++ -std=gnu++17 -O2 file.cpp -lm
```

and then executed with the command

```
./a.out
```

III. Files `*.java` are assumed to be Java code and are compiled with `OpenJDK 17.0.8.1` with the command

```
javac -encoding utf8 file.java
```

and then executed with the command

```
java -Xss128m -Xms512m -Xmx512m file
```

IV. Files `*.py` are assumed to be Python 3 and are executed with `Python 3.8.13 (PyPy 7.3.9)`. This version performs just-in-time compilation, with the command:

```
pypy3 -m py_compile file.py
```

and executes the code with the command

```
pypy3 file.py
```

Limits

Attention: The use of pragmas is explicitly prohibited in C/C++ solutions (such as modifying optimization level). Solutions containing pragmas will be considered invalid.

Compilation: The compilation process can take at most 60 seconds and the maximum source code size is 100 KB. Every program/solution must be submitted in a **single file**. For Java submissions, the file must have the same name as the class that contains the main method. There is no limit to the number of classes to be contained in that file.

Runtime The maximum runtime for each problem is 2 seconds, except in problems A, "Exploding Fireants", and F, "Stickers", which have a 1-second limit.

Input/Output

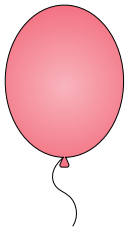
All programs should read the input from the standard input, and write the output to the standard output. All lines (both in the input and output) should end with the newline character (`\n`). Except when explicitly stated, a single space is used as a separator. No line starts or ends with any kind of white space.

Development Environment

- Operating System
  - Ubuntu 22.04
- Graphical Environment
  - Gnome 3
- Text Editors
  - vi/vim/gvim
  - Emacs
  - Gedit
  - Geany
  - VSCode (with C, C++, Python and Java extensions)
- Integrated Development Environments:
  - Codeblocks
  - Eclipse
  - IDEA Community Edition
  - PyCharm Community Edition
- Debuggers
  - gdb
  - jdb
  - pdb

  - valgrind
- Browsers
  - Firefox
- Documentation
  - Devdocs available on Mooshak

PROBLEMS



Problem A: Exploding Fireants

You have to cross a terrain full of fireants. Fireants are very hostile and you don't want to have anything to do with them!
Fortunately, you have some grenades!

Task

Each grenade blasts all ants within a given radius. Knowing that you have to traverse a given extension of terrain, which is given by a line, you have to find out the minimum number of grenades necessary to traverse the terrain.

Input

The first line contains an integer N that represents the number of test cases. Each test has two lines:

- A line with an integer that represents the diameter of the grenade blast D;
- A string of C characters consisting of the characters `'.'` for an empty space and `'F'` for a fireant.

Constraints

$1 \leq N \leq 1000$	Number of lines
$1 \leq D \leq C$	Diameter of the grenade blast
$1 \leq C \leq 1000$	Number of characters in each string

Output

Your program should output for each test case the minimum number of grenades necessary for exploding all the fireants.

Sample Input

```
3
4
....FF.FFF.F.....FFF....
6
....FFF.FF....F....F....
3
....FFF.F.F....F....F.F...
```

Sample Output

```
3
2
4
```

Sample Explanation

In the first case we need three grenades:

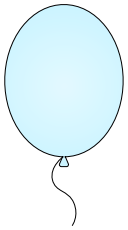
1. Kill the first 3 ants
2. Kill 3 more ants
3. Kill the last 3 ants

In the second case, we need two grenades:

1. Kill the first 5 ants
2. Kill the remaining ants

In the last case, we need four grenades:

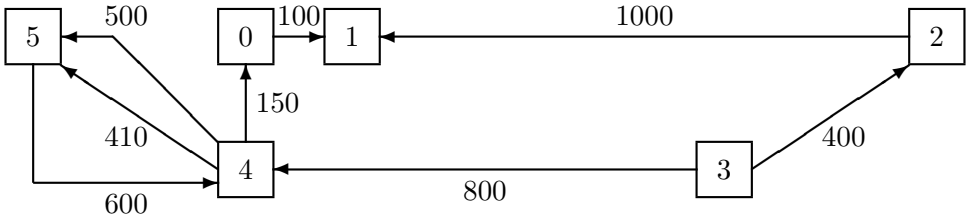
1. Kill the first 3 ants
2. Kill 2 more ants
3. Kill 1 ant
4. Kill the remaining 2 ants



Problem B: Safest Routes

In a pedestrian-friendly city, the metro network (underground and surface) is excellent and road traffic is very limited. Passenger and freight vehicles can only circulate to enter or leave the city (being not allowed to move between two city points). To make road journeys within the city more difficult and discourage drivers from using their private cars, all streets are one-way and it is not always possible to go from one location to any other.

Of course, the ban on driving does not apply to priority vehicles such as ambulances and fire trucks. However, for one of these vehicles to travel from one location to another (without leaving and re-entering the city), it often has to go through some streets in the *wrong direction* (that is, in the opposite direction to the permitted one). As travelling in the wrong direction increases the risk of accidents, the goal is to choose *safest routes*, which are routes whose *danger* (assessed by the distance travelled in the wrong direction) is as short as possible.



In the city whose map is depicted above, there are 6 squares connected by 8 streets. An arrow from a square q_1 to a square q_2, with label l, means that there is a one-way street from q_1 to q_2 (i.e., whose permitted direction is from q_1 to q_2), spanning l metres. Let us see some examples of safest routes.

- From square 4 to square 5, there are two safest routes: one through the street of length 410 and the other through the street of length 500. The danger (the distance travelled in the wrong direction) is zero.
- From square 0 to square 2, the only safest route is $0, 4, 3, 2$, whose danger is 950 metres. Notice that $0, 1, 2$ is a route with 1000 metres of danger.
- From square 1 to square 2, there is also only one safest route, through the street spanning 1000 metres. Its danger is 1000.

Task

Write a program that, given the city map and two distinct squares, o and d, computes the danger of the safest routes from o to d. It is guaranteed that, for the given inputs, there is some route from o to d.

Input

The input first line has three integers, Q, S and T, which denote, respectively, the number of squares, the number of streets and the number of test cases. Squares are identified by integers, ranging from 0 to $Q - 1$.

Each of the following S lines contains three integers, q_1, q_2 and l (where $q_1 \neq q_2$), which indicate that there is a one-way street from square q_1 to square q_2 whose length is l metres.

T lines follow, each one with two distinct integers, o_i and d_i, which represent the first and the last squares of the safest routes in the i^{th} test case (for every $i = 1, \ldots, T$).

Constraints

$2 \leq Q \leq 30\,000$	Number of squares
$1 \leq S \leq 150\,000$	Number of streets
$1 \leq T \leq 10$	Number of test cases
$1 \leq l \leq 10\,000$	Length of a street (in metres)

Output

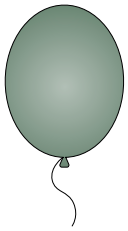
The output consists of T lines, each one with a single integer. The number on the i^{th} line represents the danger (in metres) of the safest routes from o_i to d_i (for every $i = 1, \ldots, T$).

Sample Input

```
6 8 3
0 1 100
2 1 1000
3 2 400
3 4 800
4 5 500
5 4 600
4 5 410
4 0 150
4 5
0 2
1 2
```

Sample Output

```
0
950
1000
```



Problem C: Seraphina's Lost Treasure

As Soren, a proud descendant of the legendary pirate Seraphina Seafire, delved into the depths of her family's attic, an extraordinary revelation awaited her. Amidst the forgotten relics, she unearthed a tattered and aged map, its secrets veiled by the passage of time. This weathered parchment revealed the existence of a clandestine treasure, shrouded in mystery and whispered tales of adventure.

The map unfolded, unveiling a meticulously drawn **grid**. Each square of the grid held the potential for untold riches, an intricate tapestry of possibilities. The inked lines traced the boundaries, distinguishing between **open** pathways, symbolized by an ethereal dot ('.'), and formidable obstacles, manifested as imposing **walls** ('#').

A moment of anticipation engulfed Soren as her gaze fell upon the map's enigmatic markings. There, at the **top row** of the grid, lay the **starting point**, denoted by the letter 'O'. It beckoned her, a symbolic call to embark on a perilous journey, just as her forebearer, Seraphina Seafire, had done centuries ago.

Yet, the true purpose of Soren's quest was revealed by the **final mark**, an 'X' etched with purpose and significance on the **bottom row**. It signified the coveted location of the hidden treasure, a beacon of wealth and wonder that had evaded the grasp of countless adventurers throughout history.

With the ancient map clutched tightly in her hands, Soren's heart quickened, driven by the exhilarating prospect of unearthing her ancestor's fabled bounty. It was time to chart her course, to navigate the treacherous paths and overcome the obstacles that stood between her and the illustrious treasure that lay dormant, awaiting its rightful heir. Alas, duty called, for it was a weekday, and the demands of work compelled her to wait for the grand adventure that awaited her.

Regrettably, amidst the mundane task of disposing of old work documents, Soren's heart sank as she inadvertently fed the map into the shredder, reducing it to a confounding collection of fragmented **horizontal stripes**; each one a mere remnant of the intricate cartography that once held the key to untold riches. Yet, amidst the despair, a flicker of hope emerged, as Soren realized that with careful deliberation, she could still resurrect the original map from this fragmented puzzle.

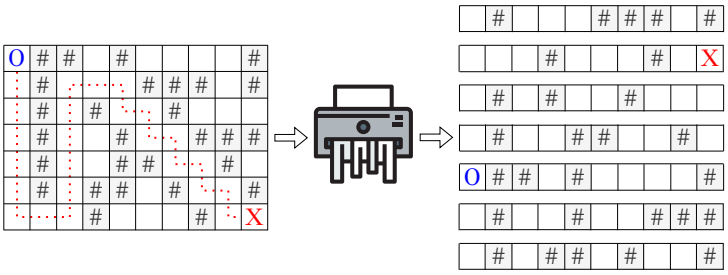


Figure 1: The shredding of a treasure map; an unfortunate disaster!

To piece together the fragmented map, Soren's task was clear: **arrange the stripes**, connecting the top and bottom rows to forge a path from the starting position ('O') to the coveted treasure location ('X'). With each stripe falling into place, the path would gradually reveal itself, charting a course through the puzzle and unveiling the hidden riches that awaited her.

Task

The task at hand is to determine **the only possible configuration** that can be achieved by rearranging the order of the horizontal stripes in the shredded map while adhering to the following criteria:

1. The **starting position** is positioned on the **top row**.
2. The **treasure location** is situated on the **bottom row**.
3. There is **at least one path** connecting the starting position to the treasure.

It is important to note that the paths between cells within the map always traverse **horizontally** or **vertically** (no diagonal connections are possible).

Input

The first line contains two integers separated by a space: the width (W) and the height (H) of the map. The next H lines represent the map stripes. Each line contains a string of W characters, where:

1. A # represents a wall.
2. A . represents an open space.
3. A O represents the starting position.
4. A X represents the treasure location.

There is only one 'O' and one 'X'.

Constraints

$2 \leq W \leq 20$ Width of the map

$2 \leq H \leq 10$ Height of the map

Output

The reconstructed map is represented by H lines, with each line containing a string of W characters, where:

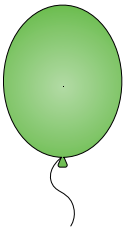
1. A # represents a wall.
2. A . represents an open space.
3. A O represents the starting position.
4. A X represents the treasure location.

Sample Input

```
10 7
.#...###.#
...#...#.X
.#.#..#...
.#..##..#.
O##.#....#
.#..#..###
.#.##.#..#
```

Sample Output

```
O##.#....#
.#...###.#
.#.#..#...
.#..#..###
.#..##..#.
.#.##.#..#
...#...#.X
```



Problem D: This is the Way

You have just arrived at planet Numberlore. While ordering 2 space beers from the Fibonacci bar, the bartender was quick to tell you that you should instead ask for 10 beers. "Always use binary numbers. This is the way"!

Later, when entering Hotel 10101, you find out Numberlorians are very superstitious. In fact, this hotel has only 14 rooms, all corresponding to numbers that in binary have more ones (digit 1) than zeros (digit 0). This is the list of rooms with the corresponding decimal number in parentheses:

1 (1)	11 (3)	101 (5)	110 (6)	111 (7)	1011 (11)	1101 (13)
1110 (14)	1111 (15)	10011 (19)	10101 (21)	10110 (22)	10111 (23)	11001 (25)

These ten numbers are all binary numbers smaller or equal to 25 in which the quantity of ones is strictly larger than the quantity of zeros. However, there are many other such numbers bigger than 25, such as 11010 (26) or 101101 (45).

You could not stop thinking about this in your decimal mind, wondering how many such numbers were there for any given range.

Task

Given two integers a and b (in decimal base), your task is to compute the number of numbers in the range $[a, b]$ (inclusive) that have more ones than zeros in their binary representation.

Input

The first line contains an integer N, indicating the number of test cases that follow.

Each of the following N lines contains two integers a b defining a range for which you need to compute the desired quantity.

Constraints

$1 \leq N \leq 100$	Number of test cases (ranges)
$1 \leq a \leq b \leq 10^{18}$	Range limits

Output

The output should contain N lines, each one indicating the number of numbers in the respective range that have more ones than zeroes in their binary representation.

Sample Input

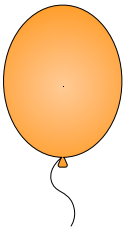
```
5
1 25
10 100
41 42
999 999
1234 5678
```

Sample Output

```
14
48
0
1
2258
```

Sample Explanation

- There are 14 numbers in the range [1,25] that have more ones than zeroes in their binary representation *(the numbers given in the problem statement for the 10101 hotel)*
- There are 48 such numbers in the range [10,100]
- There are no such numbers in the range [41,42]
- There is 1 such number in the range [999,999]
- There are 2258 such numbers in the range [1234,5678]



Problem E: Shipping Containers

Shipping Inc. is a company that specializes in shipping containers across the world. Currently, they transport containers from a main hub to several ports using their ships such that each ship is assigned to a single destination port. It is worth noting that not all ships can transport every container, since the port where the container will arrive cannot be too far away from the final destination of the container, by land.

As the company has many requests they are often not able to ship all containers in time. As such, they would like your help figuring out which containers to ship within a given time period in order to maximize their profit.

Task

You are given the following information:

- The deadline D for the time period. Time starts at 0.
- The number of ships N.
- The capacity q_i of each ship $1 \leq i \leq N$, i.e., the maximum number of containers a ship can carry on a single trip.
- The travel time t_i of each ship $1 \leq i \leq N$ to get from the main hub to its assigned port to make a delivery. The ship takes the same time to come back to the main hub. Note that a ship may be able to perform several deliveries before the deadline.
- The number of containers C.
- The profit p_j of each container $1 \leq j \leq C$ if delivered successfully to a port within the deadline.
- A list of M pairs (i, j) denoting that a ship i can deliver a container j.

Furthermore, you must take into account that each container takes 1 unit of time to be loaded onto a ship, and 1 unit of time to be unloaded onto a port. A container is only considered to be delivered in time if it finishes unloading before or at the deadline.

Your task is to compute the maximum sum of profits of delivered containers.

Input

The first line contains an integer T denoting the number of test cases. T test cases follow.

The first line of a test case contains four space-separated integers, D, N, C, and M.

The second line contains N space-separated integers denoting the values of q_i.

The third line contains N space-separated integers denoting the values of t_i.

The fourth line contains C space-separated integers denoting the values of p_j.

The following M lines contain two space-separated integers (i, j) each, denoting that ship $1 \leq i \leq N$ can deliver container $1 \leq j \leq C$.

Constraints

$1 \leq T \leq 100$	Number of test cases
$1 \leq D \leq 1000$	Deadline
$1 \leq N \leq 20$	Number of ships
$1 \leq C \leq 2500$	Number of containers
$C \leq M \leq \min(N \times C, 2 \times 10^4)$	Number of ship/container pairs
$1 \leq q_i \leq 10$	Ship's capacity
$1 \leq t_i \leq D$	Ship's travel time
$1 \leq p_j \leq 100$	Container's profit

The sum of M over all test cases does not exceed 2×10^4.

Output

Output a single integer denoting the maximum sum of the profits for delivered containers within the deadline.

Sample Input

```
1
20 3 10 15
2 3 1
7 4 6
8 9 8 1 3 10 4 2 4 2
1 1
1 2
1 3
1 7
1 8
1 10
2 4
2 5
2 6
2 9
```

```
2 10
3 4
3 6
3 8
3 10
```

Sample Output

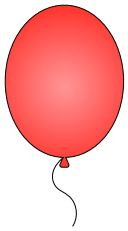
```
38
```

Explanation

There is only a single test case. One possible shipping allocation to get a profit of 38 is:

- Ship 1 loads containers 1 and 2, finishes the trip and finishes unloading the containers at time 11. It arrives back at the main hub at time 18, which means there is no time left to deliver another container given the trip time of the ship.
- Ship 2 loads containers 5, 6 and 9, finishes the trip and unloads all containers by time 10. It arrives back at the main hub at time 14. Then, it loads container 10, finishes the trip and unloads it at time 20. Note that if this ship tried to take another container on the second trip, there would be no time left at the destination to unload either container.
- Ship 3 loads container 8, finishes the trip and unloads it at time 8. It gets back to the main hub at time 14. There is no time left for another delivery.

The delivered containers were 1, 2, 5, 6, 8, 9, and 10, for a total profit of 38.



Problem F: Stickers

Little Kevin is an 8-year kid that loves to collect stickers. Every week he manages to convince his parents to buy him 30 packets of stickers, but he is still far from finishing the album. The problem with stickers is that they keep coming repeated in every new packet he gets, while some are still missing. To solve this problem, Kevin will attempt to use an aggressive approach at school by offering several stickers for the missing ones. For this, Kevin needs to determine how many copies he has from a specific set of stickers. As a start, Kevin has already sorted all his stickers ascendingly based on the sticker number. Help him find how many stickers he has from a set of specific stickers.

Input

The first line of the input contains an integer N, denoting the number of stickers Kevin has. The second line has N sticker identifiers (B), in ascending order, separated by one whitespace. The third line contains the amount of sticker identifiers Kevin is interested in finding (M). The last line contains M distinct sticker identifiers (B) that Kevin wants to know how many copies he has, separated by one whitespace.

Constraints

$1 \leq N \leq 500,000$	Number of stickers
$1 \leq M \leq 100,000$	Number of sticker IDs to find
$1 \leq B \leq 100,000$	A Sticker ID

Output

The output should be a single line with one integer representing how many stickers there are in Kevin's collection whose identifier is one of those indicated in the last line of the input.

Sample Input 1

```
4
5 9 9 9
2
8 9
```

Sample Output 1

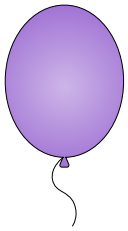
```
3
```

Sample Input 2

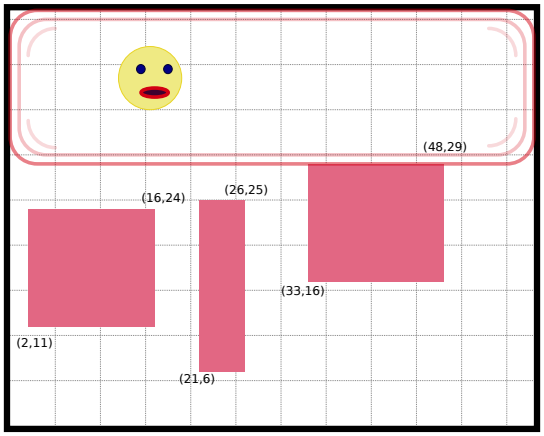
```
8
1 5 7 7 9 9 9 14
3
5 9 1
```

Sample Output 2

```
5
```



Problem G: Snake Oil Salesperson



In the current media landscape, all sorts of experts, pundits, and salesperson are eager to reach out to as many listeners as possible. Snake oil salesperson are particularly interested in tricking as many listeners as possible. They want their message to spread out as far as possible. Obstacles prevent a message from propagating further in any of the four orthogonal directions. As such they have to calculate the best spot to transmit their message.

Task

The media landscape is represented by a rectangle that may contain rectangular obstacles. Your task is to compute the location of the largest rectangular area(s) that the snake oil salesperson message can reach out to.

Input

The first line contains two integers, W, H representing the width and height of the media landscape, respectively. The second line contains an integer N. The following N lines contain the description of the rectangular obstacles. Each line r contains four integers, $X_1^r, Y_1^r, X_2^r, Y_2^r$, where (X_1^r, Y_1^r) represents the coordinate of the left bottom corner of the rectangle and (X_2^r, Y_2^r) represents the coordinate of the right upper corner of the rectangle. Note that rectangles may overlap. Overlaped space is considered an obstacle.

Constraints

$10 \leq W \leq 2 \cdot 10^5$	Media landscape width
$10 \leq H \leq 2 \cdot 10^5$	Media landscape height
$1 \leq N \leq 100$	Number of rectangular obstacles
$0 \leq X_1^r < X_2^r \leq W$	Horizontal coordinates of the corners of the obstacle
$0 \leq Y_1^r < Y_2^r \leq H$	Vertical coordinates of the corners of the obstacle

Output

The first line of the output should consist of one number, A, representing the area of the largest rectangle that the snake oil salesperson's message can reach out. The following lines should consist of four integers, $X_1^*, Y_1^*, X_2^*, Y_2^*$, representing the coordinates of the left bottom corner and right upper corner of the rectangular areas. If there are multiple such areas, they should be presented in lexicographic order.

Sample Input – 1

```
58 51
3
2 11 16 24
21 6 26 25
33 16 48 29
```

Sample Output – 1

```
1276
0 29 58 51
```

Sample Explanation – 1

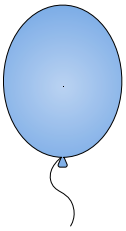
The upper area in the media landscape has the largest area. His message will reach out the upper, left and right boundaries of the landscape. In the bottom, his message is blocked by the obstacle located at $(33, 16)$ $(48, 29)$.

Sample Input – 2

```
100 100
2
0 10 20 15
10 0 15 20
```

Sample Output – 2

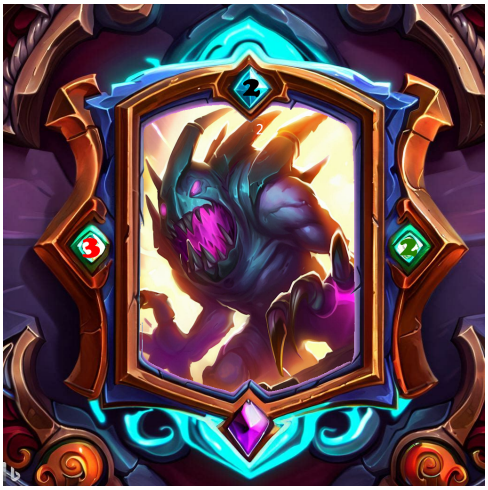
```
8000
0 20 100 100
20 0 100 100
```



Problem H: MagicStone Deckbuilder

Bob Tiger is developing a digital Tradeable Card Game called *MagicStone*. One core feature of the game is a simple AI that is able to automatically create a deck to play against a player, created from the same set of cards the player has at his disposal. A creature card in *MagicStone* has 3 main stats: *Cost*, *Attack*, and *Defense*. The *Power* of a card is considered to be the sum of its *Attack* and *Defense*.

Given a set of available cards, the deckbuilder selects a specified number of cards that maximize the total power while having a total cost lower or equal to a specified maximum total cost. Each card can only be selected once.



Task

Write a program that receives the number of cards to select for the deck, the maximum total cost of the deck, and a list of cards ordered by cost. Each card contains the name of the card, and its *Cost*, *Attack* and *Defense* values.

Your program should print the total *Power* value of the deck and its total *Cost* together with a list of the names of the selected cards. If it is not possible to generate a deck that satisfies the restrictions, "No Solution" should be printed instead.

If there is a tie between two potential decks, the one with the highest total *Cost* wins. If there is a tie on *Cost*, the order in which cards appear in the input is used to untie, by giving preference to cards that appear first. If the first card of deck A appears earlier than the first card of deck B, then deck A wins. If there is a tie in the first card, then it is broken on the second card, and so forth.

Input

The first line specifies the deck size, S, representing the number of cards to be selected for the deck. The second line contains an int value, C, representing the maximum total cost of the deck for it to be considered. The third line contains the number of cards to consider when building the deck, N. The following N lines contain cards data: a string of lowercase letters (and the character '_') representing the name of the card, n_c, an integer representing the cost of the card, c_c, an integer representing the attack value, a_c, and another integer representing the defense of the card, d_c.

Constraints

$1 \leq S \leq 30$	deck size
$1 \leq C \leq 300$	maximum cost
$1 \leq N \leq 50$	number of cards to consider
$S \leq N$	deck size lesser or equal to number of cards
$1 \leq \operatorname{len} n_c \leq 20$	card name length
$1 \leq c_c \leq 10$	card cost value
$0 \leq a_c \leq 20$	card attack value
$0 \leq d_c \leq 20$	card defense value

Output

The first line of the output should contain the number of selected cards in the deck, the total *Power* value of the deck, and the total *Cost* of the deck. Each of the following lines, should contain the selected card names, by the order in which they appeared in the input.

Sample Input

```
4
13
10
righteous_protector 1 1 1
sanguine_warrior 1 2 1
blood_liadrin 2 3 2
light_blade 2 2 3
buckthorn_servant 3 2 4
stout_defender 4 3 6
bold_attacker 4 8 2
the_curator 5 4 6
tontochronos 6 7 7
leviathan 6 9 5
```

Sample Output

```
4 32 13
sanguine_warrior
blood_liadrin
bold_attacker
tontochronos
```

Sample Explanation

In this example, we have to select 4 cards from a set of 10 cards, maximizing power, but with the restriction that their total cost cannot be greater than 13. The resulting deck has 4 cards, with 32 total combined power (14 + 10 + 5 + 3) and 13 total combined cost (6 + 4 + 2 + 1).

There is no deck satisfying the restrictions with higher power than the one presented in the output. There are other decks with the same power, for instance by replacing tontochronos with leviathan, or replacing blood liadrin with light blade. However, tontochronos and blood liadrin are preferred because they appear earlier in the given card list.

Sample Input

```
7
13
10
righteous_protector 1 1 1
sanguine_warrior 1 2 1
blood_liadrin 2 3 2
```

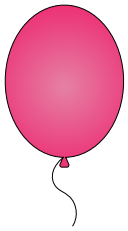
```
light_blade 2 2 3
buckthorn_servant 3 2 4
stout_defender 4 3 6
bold_attacker 4 8 2
the_curator 5 4 6
tontochronos 6 7 7
leviathan 6 9 5
```

Sample Output

```
No Solution
```

Sample Explanation

In this example it is not possible to select 7 cards with cost 13 or less. The six cheapest cards have a combined cost of 13 and adding any other card would make us go over this limit.



Problem I: Talking with Aliens

Albert is a young physicist with a keen interest in astronomy and space exploration. His greatest passion is extraterrestrial life and he dreams of becoming the first Human to establish contact with our outer-space friends.

During the past few months, Albert has been busy building in his garage a machine that allows him to send messages into the big, cold void of space. Having limited resources, Albert's machine only allows him to send a message once per day. So, he must chose carefully what message to send. His machine works as follows:

1. Albert must input an initial String composed of characters from 'A' to 'Z' and numbers from '0' to '9'.

2. Afterwards, Albert produces a set of intermediate Strings using very simple operations. The machine instruction set features only two opcodes:

   - `JOIN k m`, which creates a new intermediate String by concatenating (a copy of) the characters of k-th generated String with (a copy of) the characters of the m-th generated String.
   - `CUT k l u`, which creates a new intermediate String composed of (a copy of) the characters of the k-th String from index `l` (inclusive) to index `u` (exclusive).

3. The final generated String is the actual message to be sent into the Cosmic abyss.

For example, if Albert types the following into his machine

```
PROGRAMMINGISFUN
CUT 0 6 7
CUT 0 8 9
JOIN 1 2
CUT 0 0 1
CUT 0 14 15
JOIN 5 4
JOIN 3 6
```

the final message is "MIUP". The first two `CUT` instructions generate the singleton Strings "M" and "I", respectively. The following `JOIN` command builds "MI", which is now the third intermediate String on the machine stack. Then, the next two `CUT` generate "P" and "U", respectively. The subsequent `JOIN` builds the String "UP", the sixth intermediate String. Finally, the last `JOIN` command is used to produce "MIUP".

Task

Given the initial String and the set of instructions Albert typed on his machine, your task is to compute the final String and print it.

Input

The input starts with a line with a single integer N. The second line contains S_0, the initial String typed by Albert. Then, N lines follow. The i-th line, where $1 \leq i \leq N$, contains an instruction to build String S_i . Each line can be of one of two patterns:

- "`JOIN` k m", where $0 \leq k, m < i$, means String S_i is built by concatenating a copy of the String S_k with a copy of String S_m.
- "`CUT` k l u", where $0 \leq k < i$, means String S_i is built as the sub-string of S_k containing characters from index l (inclusive) to u (exclusive). Both l and u are within the bounds of S_k. If $l == u$, then S_i is the empty String.

Constraints

$1 \leq N \leq 2500$

$1 \leq \texttt{length}(S_0) \leq 1000$

$1 \leq \texttt{length}(S_N) \leq 1000$

Output

The final computed String. This String never exceeds 1000 characters.

Sample Input 1

```
2
HELLOWORLD
CUT 0 0 5
JOIN 1 1
```

Sample Input 2

```
7
PROGRAMMINGISFUN
CUT 0 6 7
CUT 0 8 9
JOIN 1 2
CUT 0 0 1
CUT 0 14 15
JOIN 5 4
JOIN 3 6
```

Sample Output 1

```
HELLOHELLO
```

Sample Output 2

```
MIUP
```

Problem J: The Enigmatic Connection

(Scene: Poirot's Study. Poirot and Hastings sit at a table, engrossed in their latest puzzle.)

Poirot: Ah, Hastings, we find ourselves faced with a most intriguing conundrum today. Two enigmatic characters have crossed our path. I think their connection, my dear friend, lies hidden in plain sight.

Hastings: Quite right, Poirot. But how do we unravel this mysterious bond? What *is* their connection?

Poirot: Ah, my friend, that is the question. We must carefully examine the intricate threads that intertwine their movements. Let us delve into the depths of this puzzle and uncover the truth.

Hastings: (Duly puzzled) But, Poirot, even if we know the places that each one visited, and the order by which they visited them, their relation seems too vague to be helpful.

Poirot: You surely see, *mon ami*, that we must look at the commonalities, both at their number and at their order? We must strive to find the longest possible chain of places common to both characters' routes, while respecting their order.

Hastings: (Eagerly) What about the distinct ways of pairing the commonalities, Poirot? Is the multiplicity of ways an element of this hidden connection?

Poirot: (Nodding) An astute observation, Hastings. Indeed, we must consider the distinct ways that may lead to our desired result. We shall not overlook a single possibility.

(Poirot takes out two folded papers from his pocket, both bearing sequences of pairs of numbers, and they proceed to study them.)

Task

Poirot: (Addressing you) Your task, my friend, given the route each of our enigmatic characters took through London, is to help us determine the longest order-respecting chains of commonalities between them. Furthermore, we also need to know in how many different ways such chains of commonalities may be formed.

Poirot: (Humbly, as only he can) Even Hercule Poirot can be defeated by the volume of data and must recognise when to resort to *le numérique*.

Input

Hastings: (Patiently) I'll say. On the first line, you will find an integer M, denoting the length of the route of the first character. Each of the following M lines holds two integers constituting a pair (x, y) of grid coordinates, corresponding to the places he visited, in the order by which he visited them.

Poirot: (Becoming impatient) On the following line, there will be an integer N, followed by N lines, similarly describing the route of the second character.

Constraints

$0 \leq M, N \leq 4000$ The lengths of the characters' routes

$1 \leq x, y \leq 100$ The values of the coordinates

All values will be smaller than 2^{31}.

Output

Chief Inspector Japp: (Commanding) Now, there! Once you have the length of the longest order-respecting chains of commonalities between the two routes, and the number of ways a chain of commonalities of that length may be formed, just write the former, followed by the latter, on the same line. If you find nothing in common between the two routes, write "`no connection found`" and you're done. Now, get to work!

Sample Input 1

```
7
10 25
40 20
85 15
40 20
70 45
10 25
40 20
6
40 20
70 45
85 15
10 25
40 20
10 25
```

Sample Output 1

```
4 4
```

Sample Explanation 1

Miss Lemon: (Sternly) There are three length 4 order-respecting chains of commonalities between the two routes:

1. $(40, 20)$ $(85, 15)$ $(10, 25)$ $(40, 20)$
2. $(40, 20)$ $(85, 15)$ $(40, 20)$ $(10, 25)$
3. $(40, 20)$ $(70, 45)$ $(10, 25)$ $(40, 20)$

While there is only one way to form the first and the second chains, the third chain may be formed in two ways: either by first matching the first $(40, 20)$ pair from the first route with the first $(40, 20)$ pair from the second route; or by first matching the second $(40, 20)$ pair from the first route with the first $(40, 20)$ from the second route. The remaining three pairs may only be matched in one way.

There is no length 5 (or longer) chain of order-respecting commonalities between the two routes.

Sample Input 2

```
1
10 25
1
25 10
```

Sample Output 2

```
no connection found
```

Sample Explanation 2

Miss Lemon: (Sympathetically) I hardly think an explanation is necessary, here.

Sample Input 3

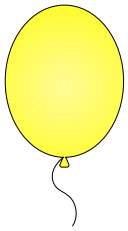
```
3
10 10
20 20
30 30
3
30 30
20 20
10 10
```

Sample Output 3

```
1 3
```

Sample Explanation 3

Miss Lemon: (Confidently) You can figure this one out, as well, surely!



Problem K: Periodic Quadratic Equations

To solve a quadratic equation we usually use an off-the-shelf formula. For example to find out which x solves the equation $x^2 - 2x - 1 = 0$ we may use this formula to obtain $x = 1 + \sqrt{2}$, after some simplification. There is also another solution but in this question we are always interested in the largest one. Moreover this solution will always be a positive number greater than 1 in our test set.

We can consider the problem in another way and search for a solution that does not contain a square root. First note that 0 is not a solution to this equation. Hence we can swap terms and divide by x to obtain $x = 2 + 1/x$. We can now replace the x on the denominator by using the information on this equation to obtain the following sequence:

$$\begin{aligned} x &= 2 + \frac{1}{2 + \frac{1}{x}} \\ x &= 2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{x}}} \\ x &= 2 + \frac{1}{2 + \frac{1}{2 + \frac{1}{\ddots}}} \end{aligned}$$

By repeating this process we end up with a succession known as an infinite regular continued fraction. The regular attribute refers to the fact that the fraction numerators are always 1. This is fairly well behaved as it is a Cauchy sequence and therefore converges. Hence it is valid to claim that $1 + \sqrt{2} = 2 + 1/(2 + 1/(2 + 1/(2 + \ldots)))$. Using the regular property we can define a specific notation for continuous fractions and write the fraction as $[2; 2, 2, 2, 2, 2, \ldots]$.

The question now is how to find this kind of solution for other quadratic equations. Let us consider the equation $3x^2 - 18x + 26 = 0$. In this case rewriting the equation to $x = 18 - 26/x$ is not helpful because the numerator is -26 and we need a value of 1. Moreover we wish to have an expression where $x = a + 1/x_1$ and not $x = a - 1/x_1$, for some number $x_1 > 1$ and an integer $a > 0$. For simplicity, in our test set, the value of a is always an integer from 1 to 9. Hence for this particular equation we need to choose a value for a. Using the trusty quadratic formula we conclude that the numerical value of x is around 3.577350. Hence we will choose a to be the integer part of this number, i.e., $a = 3$. Note that there is also the root with a value of around 2.42265 but, as mentioned before, we are only interested in the largest root. Hence at each step the value of a corresponds to the integer part of the largest root of the polynomial.

It is possible to find the value of a using only integer calculations. Notice that for the polynomial $p(x) = 3x^2 - 18x + 26$ we have that $p(3) = -1 < 0$ and that $p(4) = 2 > 0$. Hence a 0 must be between 3 and 4, because polynomials are continuous functions[1]. We

[1]Note that this analysis is not enough for every polynomial, as for example the polynomial $331x^2 - 2146x + 3477$ always yields positive values when calculated with integer arguments. In this case recall that these polynomials are parabolas and therefore they reach extreme values at the symmetry axis, which in this case is at $x = 3 + (80/331)$. Therefore both these zeros are also between 3 and 4.

can now use $a = 3$ as the integer part of the largest root to obtain the substitution $x = 3 + 1/x_1$. This yields the equation $x_1^2 - 3 = 0$. Again rewriting this equation does not yield a suitable fraction, so instead we repeat the substitution process. In this case we have $x_1 = 1 + 1/x_2$. The following table shows the full process, starting with $x = x_0$.

$$\begin{aligned}
3x_0^2 - 18x_0 + 26 &= 0, & x_0 &= 3 + 1/x_1 \\
x_1^2 + 0x_1 - 3 &= 0, & x_1 &= 1 + 1/x_2 \\
2x_2^2 - 2x_2 - 1 &= 0, & x_2 &= 1 + 1/x_3 \\
x_3^2 - 2x_3 - 2 &= 0, & x_3 &= 2 + 1/x_4 \\
2x_4^2 - 2x_4 - 1 &= 0, & x_4 &= x_2
\end{aligned}$$

We can collect the desired continuous fraction from the substitutions on the right of the equations. Notice an interesting property, the equation for x_2 is the same as the equation for x_4. This implies that the process repeats from this point onward. Hence the continued fraction for x_0 is $[3; 1, 1, 2, 1, 2, 1, 2, \ldots]$, more precisely we can use parentheses to indicate the repeating pattern, i.e., $x_0 = [3; 1, (1, 2)]$. It turns out that for any quadratic equation the process described will always end up repeating, hence yielding periodic continued fractions. In this problem the goal is to determine the size of these repeating patterns.

Input

The first line contains the value `N` that indicates the number of polynomials to consider. Each of the following lines contains three numbers, each one followed by a space. These correspond to the coefficients of the initial polynomial. For a line containing `A B C` the corresponding equation is $\mathtt{A}x^2 + \mathtt{B}x + \mathtt{C} = 0$.

Output

N lines each one containing the size of the repeating pattern of the corresponding simple continued fraction.

Constraints

$1 \leq \mathtt{N} \leq 1000000$	Number of lines
$1 \leq \mathtt{A} \leq 9223372036854775807$	Coefficient of x^2
$-9223372036854775807 \leq \mathtt{B} \leq 9223372036854775807$	Coefficient of x
$-9223372036854775807 \leq \mathtt{C} \leq 9223372036854775807$	Constant Coefficient
$1 \leq a \leq 9$	value in continued fraction.

All intermediate necessary calculations fit in integers in the interval $[-9223372036854775807, 9223372036854775807]$.

Sample Input

```
5
1 -2 -1
3 -18 26
331 -2146 3477
2378 -4756 -2378
84387471324008644 -273083635876967356 220929336469426939
```

Sample Output

```
1
2
7
1
42
```