COCI 2017/2018

Round #1, October 14th, 2017

Tasks

Task	Time limit	Memory limit	Score
Cezar	1 s	64 MB	50
Tetris	1 s	64 MB	80
Lozinke	1 s	64 MB	100
Hokej	1 s	64 MB	120
Deda	1 s	64 MB	140
Plahte	2 s	512 MB	160
Total			650

Little Caesar likes card games. Everytime he comes to Zagreb, he plays blackjack, the famous card game, with his friends.

In this game, the player draws cards while the sum of the cards in his hand is less than or equal to 21 or until he says “DOSTA” (Croatian for “STOP”).
At the beginning of the game, there are 52 cards in the deck, thirteen ranks of each of the four suits. The card ranks are two, three, …, ten, Jack, Queen, King and Ace. The card values are the following: the cards with numbers on them are worth that number (e.g., “nine” is 9), the cards with pictures (Jack, Queen, and King) are worth 10, whereas the Ace is worth 11.

Caesar has found himself in an interesting situation. During the game, he already drew N cards whose sum is less than or equal to 21 and is now having second thoughts about drawing one more card or not. Let’s assume X is the difference from the sum of the cards so far to 21. Everybody knows that you don’t draw a card if the number of the remaining cards in the deck whose value is greater than X is greater than or equal to the number of the remaining cards in the deck whose value is less than or equal to X.

Since Caesar is having a difficult time calculating whether he needs to draw another card or not, he’s asking you to do it for him.

INPUT

The first line of input contains the positive integer N ($1 \le N \le 52$), the number of cards Caesar has drawn so far.
Each of the following N lines contains a single positive integer, the value of the i^{th} card Caesar drew.

OUTPUT

If Caesar should draw another card, output “VUCI” (Croatian for “DRAW”), otherwise output “DOSTA” (Croatian for “STOP”).

SAMPLE TESTS

input

```
6
2
3
2
3
2
3
```

output

```
DOSTA
```

input

```
1
10
```

output

```
VUCI
```

input

```
2
5
6
```

output

```
VUCI
```

Clarification of the first test case:
The sum of the already drawn cards is 15, and the difference X to 21 is 6. The number of cards in the deck with a value greater than 6 is 32 (4 Aces, 4 Kings, 4 Queens, 4 Jacks, 4 tens, 4 nines, 4 eights, and 4 sevens), whereas the number of cards in the deck with a value less than or equal to 6 is 14 (one two, one three, 4 fours, 4 fives, and 4 sixes).

Ivica is a passionate computer scientist. He recently started working on his first computer game: a clone of the popular Tetris. Although he’s far from being finished, his program supports placing five different Tetris figures shown in the image below in a matrix. Before placing it in the Tetris matrix, the figure can be rotated by 90 degrees an arbitrary number of times and coloured. Additionally, the current version of the game doesn’t support placing the figure if that would mean it goes out of the matrix boundaries or overlaps with another existing figure in the matrix.

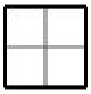


Figure 1

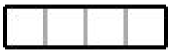


Figure 2

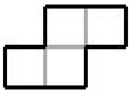


Figure 3

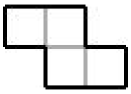


Figure 4

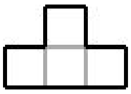


Figure 5

While Ivica was in school, his sister Marica started the game and randomly rotated, coloured and placed the figures in a way that the adjacent figures are coloured differently. Two figures are adjacent if they share a common side or touch in the tip.

When Ivica came back to his computer, he found the game running with the figures his sister placed. He wants to know how many of which figures there are in the Tetris matrix and he is asking you to help him solve this problem while he’s busy with improving the game.

INPUT

The first line of input contains positive integers *N* and *M* (1 ≤ *N*, *M* ≤ 10) that represent the number of rows and columns of the Tetris matrix.
Each of the following *N* lines contains *M* characters that represent the matrix. Each character can be ‘.’ (dot) that represents a blank space or a lowercase letter of the English alphabet that represents a part of the figure. Different letters represent different colours, and the parts of the same figure are coloured the same.

OUTPUT

You must output exactly five rows. The i^{th} line must contain the number of appearances of the i^{th} figure in the game of Tetris.

SCORING

In test cases worth 20% of total points, only the first figure will appear.
In test cases worth an additional 20% of total points, only the first two figures will appear.
In test cases worth an additional 20% of total points, only the first three figures will appear.
In test cases worth an additional 20% of total points, only the first four figures will appear.

SAMPLE TESTS

input

```
4 5
aaaa.
.bb..
.bbxx
...xx
```

output

```
2
1
0
0
0
```

input

```
4 5
.aab.
aabb.
.cbaa
cccaa
```

output

```
1
0
1
1
1
```

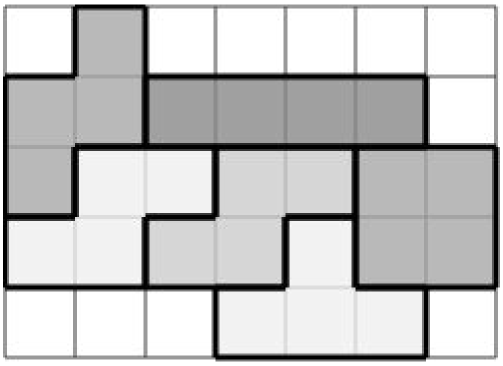
input

```
5 7
.c.....
ccdddd.
caabbcc
aabbacc
...aaa.
```

output

```
1
1
2
1
1
```

Clarification of the third test case:
The following image depicts the Tetris matrix when Ivica came back to his computer.



Recently, there has been a breach of user information from the mega-popular social network *Secret Network*. Among the confidential information are the passwords of all users.

Mihael, a young student who has been exploring computer security lately, found the whole thing really interesting. While experimenting with the social network, he found another security breach! When you input any string of characters that contains a substring equal to the actual password, the login will be successful. For example, if the user whose password is *abc* inputs one of the strings *abc*, *abcd* or *imaabcnema*, the system will successfully log him in, whereas the login will fail for *axbc*.

Mihael wants to know how many ordered pairs of different users exist such that the first user, using their own password, can login as the second user.

INPUT

The first line of input contains the positive integer *N* (1 ≤ *N* ≤ 20 000), the number of users. Each of the following *N* lines contains the user passwords. The passwords consist of at least one and at most 10 characters of the English alphabet.

OUTPUT

The first and only line of output must contains the number of ordered pairs from the task.

SCORING

In test cases worth 40 points total, it will hold 1 ≤ *N* ≤ 2000.

SAMPLE TESTS

input

```
3
aaa
aa
abb
```

output

```
1
```

input

```
3
x
x
xy
```

output

```
4
```

input

```
5
mir
mirta
ta
ir
t
```

output

```
6
```

Clarification of the second test case:
The first user can login as the second user, the second user can login as the first, and the third user can login as both the first and the second user.

The date of a major marathon ice hockey tournament is approaching. As it is often the case in marathon ice hockey, the game is *M* minutes long. On the field (ice) are, as in regular ice hockey, at each given moment, **six players** from each team. However, a game of marathon ice hockey can last a very long time, so the coaches brought a bunch of players in buses and planes so they can perform substitutions when the players get tired.

One of the coaches is the hero of our story, and his name is Ante. Ante brought *N* players to the tournament. For each player, he knows two things: the player quality - *K*, and the player endurance - *I*. The player endurance is the total time in minutes that a player can spend playing in the game without getting tired. If the player played for *X* minutes, then rested on the bench, then played again for *Y* minutes, his total playing time is *X* + *Y*. When a player plays a total number of minutes equal to their endurance, he gets tired and can't continue playing, so in that moment someone needs to substitute for him, or he will faint on the ice and end up in the hospital (marathon ice hockey is a dangerous game).

The quality of the team in a given minute of the game is the sum of the quality of that team's players that are currently playing. Ante is not a great coach, so he asked you to come up with the initial six players and the substitute schedule so he can achieve **the maximum possible** sum of the quality of the team for all the minutes in the game - *Z*. It is guaranteed that it will always be possible to come up with a schedule such that there are six players in the game in each minute.
For example, if the game lasted for 3 minutes, and if in the first minute the quailty of the team was 15, in the second 12, and in the third 14, Z will be equal to 15 + 12 + 14 = 41.

Please note: in marathon ice hockey, there is no goalie, because the game must be interesting!

INPUT

The first line of input contains the positive integers *M* and *N* ($1 \le M \le 500\ 000$, $6 \le N \le 500\ 000$), the duration of the game in minutes, and the number of players Ante brought.
Each of the following *N* lines contains two positive integers per player, *K* ($1 \le K \le 100\ 000$) and I ($1 \le I \le M$), the quality and the endurance.
The players are numbered from 1 to *N*, in the order given in the input (the first player's number is 1, the second is 2, etc.)

OUTPUT

The first line of output must contain the maximum possible *Z* from the task.
The second line of output must contain exactly six numbers - the numbers of the six players that start the game.
The third line of output must contain the number of substitutions *B*, that must not exceed **3 * N**.
Each of the following *B* lines must contain, in order from the earliest to the latest substitute, the information about the substitutes, three numbers per substitute, *X* ($1 \le X < M$), *A* and *B*, where *X* is the number of minutes from the start of the game when the substitute is being

done, *A* is the number of the player **exiting** the game and going to the bench, and *B* is the number of the player that is **entering** the game as their substitute.

Multiple substitutions in the same minute are permitted, but **a player cannot enter the game, and then exit in the same moment, or vice versa**.

If multiple solutions exist, output any.

SAMPLE TESTS

input

```
200 6
3 200
4 200
5 200
6 200
7 200
8 200
```

output

```
6600
1 2 3 4 5 6
0
```

input

```
9 9
10 3
9 3
13 9
5 3
15 9
100 9
3 6
2 6
1 6
```

output

```
1260
1 2 3 4 5 6
3
3 1 7
3 2 8
3 4 9
```

input

```
3 9
100 3
100 3
100 3
100 3
100 2
100 1
50 1
30 2
1 1
```

output

```
1610
1 2 3 4 5 6
2
1 6 8
2 5 7
```

Clarification of the first test case:
No substitutes were needed, the entire team will play the game from start to end.

Little Marica is making up a ~~nonsensical~~ unusual fairy tale and is telling to her grandfather who keeps interrupting her and asking her ~~stupid~~ intriguing questions.

In Marica's fairy tale, *N* children, denoted with numbers from 1 to *N* by their age (from the youngest denoted with 1, to the oldest denoted with *N*), embarked on a train ride. The train leaves from the station 0 and stops in order at stations 1, 2, 3 … to infinity.

Each of the following Marica's statements is of the form: **"At stop *X*, child *A* got out"**, where the order of these statements is completely arbitrary. In other words, it does not depend on the station order. Her grandfather sometimes asks a question of the form: **"Based on the statements so far, of the children denoted with a number greater than or equal to *B*, who is the youngest child that rode for *Y* or less stops?"** If at the moment the grandfather asks the question it hasn't been said so far that a child is getting off the train, we assume that the child is riding for an infinite amount of stops.

Marica must give a correct answer to each of her grandfather's questions, otherwise the grandfather will get mad and go to sleep. The answer must be correct in the moment when the grandfather asks the question, while it can change later given Marica's new statements, but that doesn't matter. Write a program that tracks Marica's statements and answers her grandfather's questions.

INPUT

The first line of input contains the positive integers *N* and *Q* (2 ≤ *N*, *Q* ≤ 200 000), the number of children and the number of statements. Each of the following Q lines describes:

- either Marica's statement of the form "M" *X A*, where "M" denotes Marica, and *X* and *A* are positive integers (1 ≤ *X* ≤ 1 000 000 000, 1 ≤ *A* ≤ *N*) from the task,
- or her grandfather's question of the form "D" *Y B*, where "D" denotes the grandfather, and *Y* and *B* are positive integers (1 ≤ *Y* ≤ 1 000 000 000, 1 ≤ *B* ≤ *N*) from the task.

All of Marica's statements correspond to different children and at least one line in the input is her grandfather's question.

OUTPUT

For each grandfather's question, output the number of the required child in its own line. If no such child exists, output -1.

SAMPLE TESTS

input

```
3 4
M 10 3
M 5 1
D 20 2
D 5 1
```

output

```
3
1
```

input

```
10 10
M 20 10
D 1 9
M 2 3
D 17 10
M 20 2
D 8 2
M 40 1
D 25 2
M 33 9
D 37 9
```

output

```
-1
-1
3
2
9
```

Little Donald decided to wash all *N* of his white sheets one day. After washing them, he put them to dry on the ground in his backyard. Donald placed the sheets so that **none of them touch on the tips or the sides and that none of their sides intersect**, but it is possible that he placed smaller sheets on top of bigger ones, or that a sheet is completely covering another sheet. After doing this, Donald went to bed.

Donald's friend Kim somehow got the information that Donald is drying his sheets and decided to mess with him. He found a paintball gun from his father in the attic. Along with the gun, there were *M* paintball balls in different colours. As soon as Donald fell asleep, Kim walked into his backyard and started shooting the sheets with his paintball gun. We all know that sheets bleed, so when Kim shoots the topmost sheet, that sheet would bleed the color of the ball down to all of the sheets beneath it. After Kim used all the balls, he happily left Donald's backyard.

When Donald woke up and went to get his sheets, shock ensued. On most of Donald's sheets, there was a number of new colours. Since Donald is very interested in correct data, and he's in shock and not able to think, he is asking you to tell him the number of new colours on each sheet.

We can represent Donald's backyard as an infinite coordinate system, and the sheets as rectangles parallel to the coordinate axes. Kim's shots can be represented as points in that system.

Please note: it is possible that Kim's shot missed all the sheets, but coordinates of each shot are unique.

INPUT

The first line of input contains the positive integers *N* ($1 \le N \le 80\ 000$), the number of sheets, and *M* ($1 \le M \le 80\ 000$), the number of paintball balls.
The i^th^ of the following *N* lines contains four numbers: the coordinates of the lower left corner A_i, B_i ($1 \le A_i, B_i \le 10^9$) and the upper right corner C_i and D_i, ($1 \le C_i, D_i \le 10^9$) of the i^th^ sheet.
The j^th^ of the following *M* lines contains the coordinates where Kim's j^th^ shot landed X_i, Y_i ($1 \le X_j, Y_j \le 10^9$), and K_j, the colour label of the j^th^ ball.

OUTPUT

The i^th^ of *N* lines must contain the number of new colours on the i^th^ sheet.

SAMPLE TESTS

input

```
2 2
1 1 3 3
5 6 10 10
3 3 1
5 1 2
```

output

```
1
0
```

input

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

output

```
3
2
1
```

input

```
1 3
1 1 7 7
2 6 2
4 7 3
4 4 1
```

output

```
3
```

Clarification of the test cases:

The number of the shot denotes the colour of the ball from that shot.

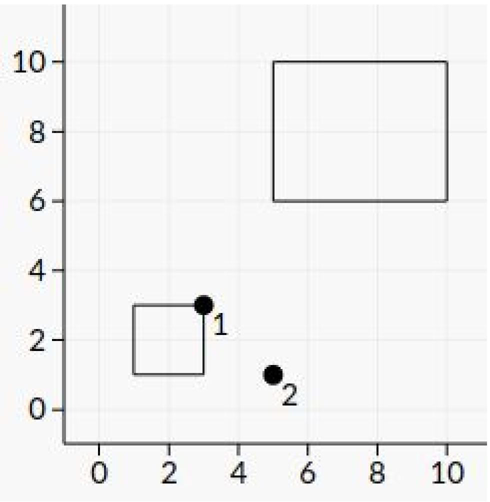


Image from the first test case

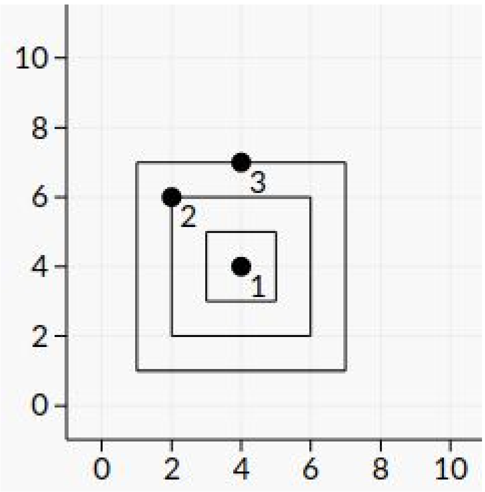


Image from the second test case