Midterm Exam

Sessions Given

- Tuesday, March 30, 8:00am - 9:15am EST
- Tuesday, March 30, 12:30 - 1:45pm EST

This exam is:

- open notes/books/any printed resources
- open laptop (anything that you have on your own computer, not online storage)

Online resources that you are allowed to access:

- Gradescope
- course website
- language documentation:
  - Java: https://docs.oracle.com/javase/10/docs/api/
  - C++: https://www.cplusplus.com/reference/ and/or https://en.cppreference.com/w/

Instructions:

Solve three out of the four problems given on the next pages. You will **not** get extra credit for solving all four problems. If you do attempt four problems, we will pick the three with highest scores. For each problem, your last submission counts.

Grading:

Every exam problem is graded out of 10 points. The total exam grade is the weighted sum computed as follows (assume *scoreN* is a score for a particular problem with *score1 >= score2 >= score3*):

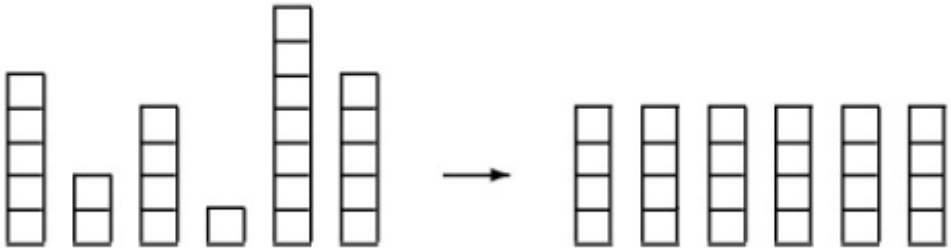
*exam = 5 * score1 + 3 * score2 + 2 * score3*

The total score for a problem is determined by the maximum between zero and the sum of scores for individual tests based on their results. The maximum score for each test is determined by *max_score = 10/number_of_tests*.

test outcome	test score
passed test	*max_score*
wrong answer	*- 0.5 max_score*
runtime error	*- 0.5 max_score*
timeout error	*- 0.5 max_score*
presentation error	*0.75 max_score*

Bricks

Little Bob likes to play with his toy bricks. He puts the bricks one upon the other and builds stacks of different heights. His sister Annie challenges him to convert this stacks of random heights into a wall in which each stack is the same height. She wants him to take a long time, so the challenge is for him to do so by moving as few bricks as possible.



Bob secretly asks you to help him figure out the smallest number of moves he needs to make to make each stack of bricks the same height.

Input

The first line of input contains an integer N (1 <= N <= 50) representing the number of stacks that Bob already has. The next line contains N numbers, the height (1 <= #boxes <= 100) of each stack.

The total number of bricks Bob used is divisible by the number of stacks and thus it is always possible to rearrange the bricks such that all stacks have the same height.

Output

You should print the line `The minimum number of moves is k.` , where k is the minimum number of bricks to be moved in order for all the stacks to have the same height.

Example 1

```
Input:
6
5 2 4 1 7 5

Output:
The minimum number of moves is 5.
```

Game Enthusiast

Kou loves playing video games and wants to share her games with classmates. She has a USB stick but it is too small to hold all of her games. Kou decided to copy only some of her games to that USB stick but struggled to determine which games to put and turned to you for help.

You are given the size of N games and the capacity of the USB stick. You need to find subset of games to store on the memory stick such that the leftover unused space is minimum.

Input

The first line of the input contains two integers V (1 <= V <= 10,000) and N (1 <= N <= 20), representing the capacity of the USB stick and the number of games.

The second line contains N integers, indicating the size of Kou's games. All of these integers are between 1 and 10,000.

Output

Print one line containing a single integer W, the total space occupied by the games copied onto the USB stick such that the USB stick has the least unused space left.

Example 1

```
Input:
5 3
1 3 4

Output:
5
```

Example 2

```
Input:
10 4
9 8 4 2

Output:
10
```

Example 3

```
Input:
20 4
10 5 7 4

Output:
19
```

Example 4

```
Input:
45 8
4 10 44 43 12 9 8 2

Output:
```

45

Time Master

Eto manages all timers in the Light Kingdom. People usually need her to repeatedly alert them at a fixed interval. They can register a reminder with her using the following instruction:

Register *id interval*

where *id* is the ID of that reminder and *interval* is the interval between two consecutive alerts. Eto will first alert the user after *interval* hours after the register request and then alert that user every *interval* hours.

All the requests are registered at once. Your task is to perform the first N alerts. If there is more than one alert occurring at the same time, you should complete them in the ascending order of *id*'s.

Input

The input consists of two parts. Each line of the first part contains a single register request as described above where 1 <= #requests <= 1,000, 1 <= *id* <= 3000 and 1 <= *interval* <= 3000. It is guaranteed that there are no duplicated *id*'s.

The second part consists of only one line, containing a single integer N (1 <= N <= 10000), representing the number of alerts you have to perform for Eto (assuming no alerts happened before).

The two parts are separated by a single # symbol on a line by itself.

Output

You should print the *id*'s of the first N alerts. One *id* per line.

Example 1

```
Input:
Register 2004 200
Register 2005 300
#
5

Output:
2004
2005
2004
2004
2005
```

Connected or Not

Alice is a network administrator. She is keeping a log of all connections between the computers in the network.

Each connection is bi–directional. Two computers are interconnected if they are directly connected or if they are interconnected with the same computer (of course, any computer is directly connected with itself).

Alice is wondering if two given computers are interconnected at a particular moment, according to the log.

Write a program that counts the number of *yes* and *no* answers to the questions of the kind: *is computer c_i interconnected with computer c_j?*

Input

The 1st line contains a single integer n, 1 <= n <= 10^6 , the number of computers in the network.

At the beginning, there is no direct connection between any two computers.

The following lines contain commands in the form:

- c c_i c_j - which means *connect c_i to c_j*, i.e., a new direct link between c_i and c_j is created. This instruction may appear any number of times for a pair of computers, but only the first occurrence of it changes the state of the network; repeated occurrences do not matter.
- q c_i c_j - which means *query: is c_i interconnected with c_j* (given the sequence of *connect* instructions that came before).

You can assume 1 <= c_i, c_j <= n. And there will be no more than 10^6 commands.

Output

Each of the queries can be answered either *yes* or *no*. The output should be a pair of integers that indicates the total number of the *yes* answers and *no* answers, respectively. The two numbers should be on a single line and be separated by a comma (no space).

Example 1

```
Input:
10
c 1 3
c 2 4
q 6 1
c 3 4
c 9 3
q 2 9

Output:
1,1
```

Example 2

```
Input:
1
q 1 1
c 1 1
q 1 1

Output:
2,0
```