

Assignment 3

Due to your outstanding performance, you have been upgraded from an intermediate developer to a senior developer. The development tasks you are assigned will be a little more complicated than before. Here is a task list:

Question 1

Description

Given some user names of the customers, sorts their names in [dictionary \(lexicographical\) order](#) from **large to small**.
字典序，即字母小放前面，短的放前面

! Warning: Take input using Scanner.

Please do not write `new Scanner` command in the loop when using Scanner in any question, otherwise you will get zero for the corresponding question.

Example: Get n integers and print them.

✓ Do:

```
Scanner scan = new Scanner(System.in); // new Scanner outside loop
while(n-->0){
    System.out.println(scan.nextInt());
}
```

✗ Don't:

```
while(n-->0){
    Scanner scan = new Scanner(System.in); // new Scanner inside loop
    System.out.println(scan.nextInt());
}
```

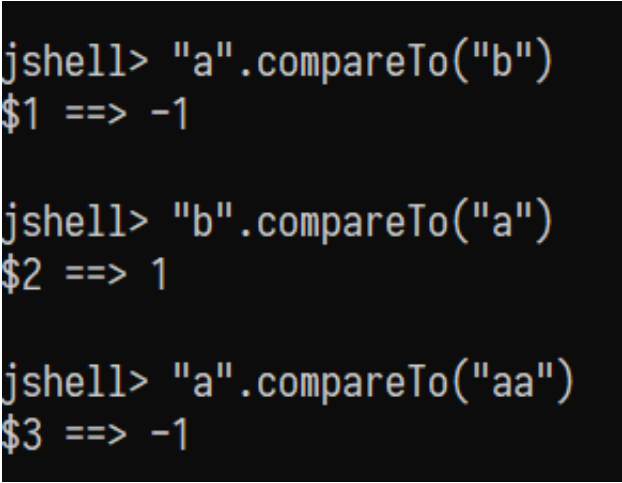
Input

The first line of input is an integer **n** representing the number of consumers ($0 < n < 10^5$), followed by **n** lines of **strings** representing customer's user **name**. (These strings are composed of only English letters and do not contain numbers, special characters and white spaces.)

Output

The output contains **n** lines of strings, in dictionary order from **large to small**.

Notice, the relative large/small is based on the result of Java String's `compareTo()` function. If the result of `str1.compareTo(str2)` is `-1`, then we consider `str1` smaller than `str2`. On the contrary, if the result is `1`, then we consider `str1` larger than `str2`. For example, the following screenshot shows that `a` is smaller than `b`, `b` is larger than `a`, and `a` is smaller than `aa`.



```
jshell> "a".compareTo("b")
$1 ==> -1

jshell> "b".compareTo("a")
$2 ==> 1

jshell> "a".compareTo("aa")
$3 ==> -1
```

Sample

```
[zhaoyao:src zhaoyao$ javac A3Q1.java
[zhaoyao:src zhaoyao$ java A3Q1
```

```
6
Edward
Cindy
Alice
Dyson
Bob
Ariel
```

input

```
Edward
Dyson
Cindy
Bob
Ariel
Alice
```

output

Question 2

Description

As more and more data was stored in the system, the development manager began to experiment with various compression techniques. Now he wants you to implement a string compression algorithm and see how it works. This string compression algorithm is very simple: It count the number of consecutive English letters, and convert the original string into the form of **"character + number"**. Take **"aaaaabbbbbbbbc"** as an example, which is converted to **"a5b7c1"**. (Notice, uppercase and lowercase letters are considered as different. e.g. **AAaa** -> **A2a2**)

Take input using Scanner.

Input

A string to be compressed

Output

A compressed string

Sample

```
[zhaoyao:src zhaoyao$ javac A3Q2.java
[zhaoyao:src zhaoyao$ java A3Q2
aaaaabbbbbbbbc — input
a5b7c1 — output
```

Question 3

Description

Halloween is coming. The SUSTech hotpot restaurant is going to hold a Halloween party. Customers have formed teams in advance to sign up for the games. If they win, they will get gifts and coupons. Your task is to organize the registration forms. However, the registration information is rather messy, you need to organize into the format of a student id corresponding a team name.

Take input using Scanner.

Input

The first line of input is an integer **n** representing the number of teams ($0 < n < 10^5$), followed by **n** lines which are team's name and members information, separated by **,**. The members information has not an uniform format, but it must contain all members' student id.

Output

Output the student id and his team name of all signed students, separated by **one space**. One student per line.

A valid student id is of 8-digit in length, with the prefix number is from 115 to 119. If a student id is not valid, don't print out it.

Sample

```
[zhaoyao:src zhaoyao$ java A3Q3
```

```
3
```

```
    pumpkin , alice( 1151001 ) bob(11510002)  
    skeleton, Edward"11610001" Dyson"11610003"  
witch , 11710011 11710012 11410013
```

```
11510002 pumpkin  
11610001 skeleton  
11610003 skeleton  
11710011 witch  
11710012 witch
```

input

output

Question 4

Description

On Halloween, the SUSTech hotpot restaurant will hold a lottery. Your task is to complete the lottery program. The lottery rules are as follows:

1. If input a number, output all student id ending with the number as the winning list
2. If input a string of consecutive numbers, output all student id containing the consecutive numbers as the winning list
3. If input a string of consecutive English letters or only one English letter, output all student id which team name containing the input string as the winning list.

Take input using Scanner.

Input

The first line of input is an integer **n** representing the number of students ($0 < n < 10^5$), followed by **n** lines which are student name and their team name, separated by **one space**.

Then followed by **one** line, composing of one of three possible operations:

(1) only a number

(2) a string of consecutive numbers

(3) one English letter or a string of consecutive English letters

Output

Output the winning list, one student per line. For each line, the first one is student id and the second one is his team name, separated by **one space**.

Sample

```
[zhaoyao:src zhaoyao$ java A3Q4
```

```
6
```

```
11510001 pumpkin
```

```
11510002 pumpkin
```

```
11610001 skeleton
```

```
11610003 skeleton
```

```
11710011 witch
```

```
11710012 witch
```

```
i
```

input

```
11510001 pumpkin
```

```
11510002 pumpkin
```

```
11710011 witch
```

```
11710012 witch
```

output

Question 5

Description

1. Design a class named **Staff**. The class contains:

(1) Private data fields:

```
int staffID;  
String name;  
int level;  
double basicSalary;  
double saleAmount;  
double curSalary;
```

(2) Implement the **getter** and **setter** method for each private field of **Staff** except `curSalary`. The field `curSalary` only implements the **getter** method.

(3) Implement a public method named `updateCurSalary()` according the fields: `level`, `basicSalary` and `saleAmount`.

```
public void updateCurSalary()
```

Formulas for calculating `curSalary`

level	curSalary
1	$\text{basicSalary} + \text{saleAmount} * 6\%$
2	$\text{basicSalary} + \text{saleAmount} * 3\%$
3	$\text{basicSalary} + \text{saleAmount} * 2\%$
4	$\text{basicSalary} + \text{saleAmount} * 1\%$
5	$\text{basicSalary} + \text{saleAmount} * 0.5\%$

2. Create a class named **A3Q5**, in which you should write the `main()` method, do the following tasks:

- (1) You should create some **Staff** objects according the input data.
- (2) You should modify the the fields of objects according the operations.
- (3) Finally you should output all `curSalary` of objects.

Take input using Scanner.

Input

The first line of input is an integers **n** ($0 < n < 100$), followed by **n** lines. For each line, the first input is the integer **id** ($0 < id < 100$) representing the staff id, the second input is a string **name** representing the staff's name, the third input is the integer **l** ($1 \leq l \leq 5$) representing the level of the staff, the forth input is the double **s** ($2500 < s < 50000$) representing the basic salary of the staff, the fifth input is the double **a** ($0 < a < 10000000$) representing the sale amount of the staff.

Then followed by an integer **P** ($1 \leq P \leq 1000000$) shows the total operations to above staffs.

Then followed by **P** lines, composing of one of three possible operations:

- `1 id l` : means to modify the staff's level to **l**;
- `2 id s` : means to modify the staff's basic salary to **s**;
- `3 id a` : means to modify the staff's sale amount to **a**.

(We assure that all the staff id will be in the list inputted before, and you can always find the corresponding staff.)

Output

Output **n** lines representing all staff's final **curSalary** in the order their objects were created, use "%.2f" to keep 2 decimal.

Sample

```
[zhaoyao:src zhaoyao$ javac Staff.java
[zhaoyao:src zhaoyao$ javac A3Q5.java
[zhaoyao:src zhaoyao$ java A3Q5
```

```
5
1 Alice 1 6666.66 10000
2 Bob 2 11000.11 50055
3 Cindy 3 18888.66 100001.05
4 Dyson 5 32506.6 223556.78
5 Edward 4 26066.88 356780.22
10
3 4 333333
2 2 12111.11
1 3 2
1 4 5
3 5 455555.78
2 1 7668.66
3 3 180001.05
3 3 193222.11
2 3 20000
2 5 28066.88
```

input

```
8268.66
13612.76
25796.66
34173.27
32622.44
```

output

Question 6 (Bonus)

Description

After having the hotpot, Mujin Nan had to pay. When calculating the total fee, a beautiful waitress informed Mujin Nan that the hotpot could be free of charge if Mujin Nan could play a game with her. The waitress was so beautiful that he was frozen at a moment. Mujin Nan was deeply attracted by this beautiful girl and decided to play the game in order to get the contact of the beautiful girl. The game is called **super transform**: Given a string and a row number, Mujin Nan is asked to transform the string to a pattern from right to left and down to up. Mujin Nan is too weak to handle this game. So he asked help from you. Could you help him to win the game and get the contact of the beautiful waitress?

```
V   E   I
AA  RT  NL
 J  T  T  I  T
NH  ES  O
 A   B   K
```

For the example above, the input string is **KOTLINISBETTERTHANJAVA** with the input row is **5**.

Input

There are two input, with the first input string `str` and the second input integer `n`. ($2 \leq n \leq 10^5$)

Output

The output is a `n` lines pattern transformed by the input string `str`.

(Notice: Print a new line `\n` after all lines has been printed.)

Sample

```
[zhaoyao:src zhaoyao$ javac A3Q6.java
[zhaoyao:src zhaoyao$ java A3Q6
KOTLINISBETTERTHANJAVA
5
```

input

```

V   E   I
AA  RT  NL
 J  T  T  I  T
NH  ES  O
 A   B   K
```

5 rows
output

Submission Requirements

Submission of Assignment:

- (1) You should submit all the source code files (with an extension **“.java”**).
- (2) The class name of each **“.java”** file should be **A3Q1, A3Q2, ... , A3Q5, A3Q6** respectively to represent these six questions.
- (3) You should submit all source code **directly** into the Sakai system, **and do not compress** them into one file.
- (4) **No Chinese characters** are allowed to appear in your code (to avoid encoding problems.)
- (5) No **package** included.
- (6) The arguments and the output must **strictly** follow the description of each question.
- (7) The assignment should be submitted before the deadline (**Nov.10th 19:00pm**). **Late submissions within 24 hours after the deadline (even a few minutes) will incur a 50% penalty**, meaning that you can only get 50% of the score, which you could get if the assignment was submitted before the latest deadline. **Assignments submitted after the latest deadline will not be graded** (meaning you will get a zero for the assignment).

Wish you to enjoy coding!