CS1020E | Lab 3 | Exercise 3 (will not be graded)

Restaurant

Objective

The objective of this exercise is to learn to use the `vector` class as well as to practice for the `string` class.

Problem Description

You have been asked to help out in a restaurant. The restaurant has *N* tables. Each of them has a name and a capacity. The name for a table is unique. The names are composed of lowercase English letters (‘a’-‘z’).

During the day, groups of people can come and go. Like the tables, each group has a name, and a number of people. Again, the names are composed of lowercase English letters (‘a’-’z’). Some groups have a favorite table: they only want to sit at their favorite table. For the other groups, we allocate the table that has the lexicographically smallest name among the available tables.

Each table can only have one group at one time. Moreover, the capacity of the table must not be less than the number of people in the group.

During the day, there will be *Q* queries for you to answer. There are 5 types of queries:

`1` *`GROUP_NAME NUM_PEOPLE TABLE_NAME`*
A group with the name *`GROUP_NAME`* and *`NUM_PEOPLE`* persons arrived. They only want to sit in table *`TABLE_NAME`*. Output *`TABLE_NAME`* if you can put the group at that table, and “`not possible`” if you cannot.

`2` *`GROUP_NAME NUM_PEOPLE`*
A group with the name *`GROUP_NAME`* and *`NUM_PEOPLE`* persons arrived. They do not have any favorite table, so you allocate them to the table with the lexicographically smallest name that you can use. Output the name of the table if you can find such a table, and “`not possible`” if you cannot find any table for this group.

`3` *`GROUP_NAME`*
A group with the name *`GROUP_NAME`* leaves the restaurant, if they are still inside. The table that the group occupied can be used after this.

`4` *`GROUP_NAME`*
Output the name of the table the group is assigned to. If there is no group with that name inside the restaurant right now, output “`invalid`”.

`5` *`TABLE_NAME`*

Output the name of the group at the table. If there is no table with that name or no group in that table, output “`invalid`”.

Add your code to the given skeleton. You can use `vector` objects to manage the groups and tables. For details of the `vector` class, please refer to
http://www.cplusplus.com/reference/vector/vector/
http://en.cppreference.com/w/cpp/container/vector

Inputs

The first line contains *N*, the number of tables in the restaurant.
The next *N* lines each contains a string and an integer—the name and capacity of each table.
The next line contains *Q*, the number of queries.
The next *Q* lines each contains a query in the format described above.

Outputs

For each query of type 1, 2, 4, or 5, output a line containing the result of the query.

Sample Run

```
4
a 5
b 4
c 1
d 2
9
1 ann 3 a
2 bob 2
2 charles 2
1 donny 2 c
4 ann
5 a
3 ann
1 eames 5 a
4 random
a
b
d
not possible
a
ann
a
invalid
```

(*User inputs are shown in* ***bold red***.)

Submission

You need to submit all your completed **`Group.cpp`**, **`Group.h`**, **`Table.cpp`**, **`Table.h`**, and **`Restaurant.cpp`** to CodeCrunch (https://codecrunch.comp.nus.edu.sg/).

Your submission for this exercise will not be manually graded, and there is no submission deadline for this exercise.