Restaurant

You've 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 will be unique. The names will be composed of lowercase English letters (a-z).

During the day, groups of people can come and go. Like the tables, each group will have a name, and a number of people. Again, the names will be composed of lowercase English letters ('a'-'z'). Some groups will have a favorite table: they only want to sit in their favorite table. For the other groups, we will allocate the table with the lexicographically smallest name that we can use.

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 4 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 on that table, and "not possible" if you can't.

2 GROUP_NAME NUM_PEOPLE

A group with the name GROUP_NAME and NUM_PEOPLE persons arrived.
They don't 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 can't 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 "".

5 TABLE_NAME

Output the name of the group on the table. If there's no table with that name or no group in that table, output "invalid".

Input Format:

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 the size of the table.
The next line contains Q, the number of queries.
The next Q lines each contains a query in the format described above.

Output format:

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

Sample Input:

```
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
```

Sample Output:

```
a
b
d
not possible
a
ann
a
invalid
```

Skeleton:

Group.h

```
#include <string>
using namespace std;

class Group {
private:
	string name;
	int size;

public:
	Group();
};
```

Table.h

```
#include <string>
#include "Group.h"
using namespace std;

class Table {
private:
	string name;
	int capacity;
	Group* group;
public:
	Table();
};
```

Restaurant.cpp

```
/*
	Name:
	Matric number:
	Plab account:
*/

#include "Table.h"
using namespace std;
int main() {
}
```

Note:

1. You should develop your program in the subdirectory, ex1 and use the cpp file provided. You should not create new file or rename the file provided.
2. You have to use OOP in this sit in lab. You are allowed to add more methods inside each class.
3. If your algorithm is different from the given skeleton, you can development according to your own algorithm.
4. Please be reminded that the marking scheme is
   Input:10%, Output:10%, Programming Style:30% and Correctness: 50%