Cinema

You've been asked to help out in a cinema. The chairs in the cinema are arranged in rows. Each row has a name, and a capacity.

During the day, groups of people will buy tickets. To make it simpler, we're not allowing people to choose their own seats: We'll do it for them. When a group arrive, we'll allocate as many of them as possible to the row with the lexicographically smallest label, and then the next one, and so on. So, a group might need to split to a few different groups if needed.

Input

The first line of the input contains an two integers R, N (R <= 100, N <= 400), representing the number of rows in cinema and the number of operations

R lines follow, each line contains a string and an integer (separated by space) representing the name and the capacity of the i-th row respectively.

The name of the rows is guaranteed to be lexicographically sorted.

N lines follow, each line is either of following:

"1 GROUP_NAME NUM_PEOPLE"

A group with the name GROUP_NAME and NUM_PEOPLE (NUM_PEOPLE <= 500) persons bought tickets. Output the number of groups that this group need to split to, or "not possible" if we can't allocate seats for the group. If we can't allocate seats, all of them will cancel the booking.

"2 ROW_NAME"

Output the number of available seats and the number of different groups in the row ROW_NAME.

"3 GROUP_NAME"

Output the list of rows that the group occupied. If they needed to split to multiple rows, output all of the rows in one line (lexicographically sorted, separated by space). Output empty line if the group didn't occupy any row.

It is guaranteed that

- For the first type of query, there is no two queries have the same GROUP_NAME
- For the second type of query, row ROW_NAME exists
- For the third type of query, group GROUP_NAME exists (has bought tickets before)

Output

For every query, you have to output one line as the problem requested.

Sample input

```
3 6
First 3
Last 1
Second 5
1 Students 2
2 First
1 Teachers 3
2 First
3 Teachers
2 Last
```

Sample output

```
1
1 1
3
0 2
First Last Second
0 1
```

Skeleton program

Group.h

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

class Group {
	private:
		string name;
		int count;

	public:
		Group();
};
```

Row.h

```
#include <vector>
```

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

class Row {
	private:
		string name;
		int capacity;
		vector <Group*> groups;

	public:
		Row();
};
```

```
Cinema.cpp
/*
	Name:
	Matric number:
	Plab account:
*/

#include "Row.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%**