Problem S
Not Another Committee!
Problem ID: committees

Your solution to this problem must be implemented using an object-oriented approach by defining the classes presented below. Any other solution will receive zero points, even if the testing system judges it correctly.

If there's one thing that universities love to do, it's to create committees. Need to design a new degree? Have a committee figure it out! Dealing with a problem? A committee will sort it out! However, given the large number of committees, the university has realized it needs software to manage the composition of these committees.

More specifically, a committee is composed of several professors, each of which can belong to one or more fields (e.g., a professor could do research in both Computer Science and Mathematics). Furthermore, professors can either be tenured or not. In this problem, we will determine if a "*bad*" committee has been formed. A committee is "*bad*" if both of these conditions are true:

- There are only one or two unique fields represented in the committee.
- There are not enough tenured professors in the committee. If the committee has N professors, we need at least $(N//2) + 1$ of them to be tenured. The operator $(//)$ means floor division (i.e., dividing and rounding down to the nearest integer.)

You **must** implement the following class definitions along with the specified constructor, attributes (also known as instance variables) and methods. You must think about the type for each attribute and the type signature for each method. For this problem, the *Input* section helps clarify these types. Implement the following classes:

A class named `Profesor` that represents a single professor at a university.

- **Constructor**
  - Implement a constructor that initializes a `Profesor` instance and takes in three arguments: the professor's name, a boolean flag that indicates whether the professor is tenured or not, and a list of the professors' fields of study.
- **Attribute(s)**
  - Define an attribute that represents the name of a professor.
  - Define an attribute that represents whether the professor is tenured or not.
  - Define an attribute that keeps track of the fields of study for the professor.
- **Method(s)**
  - No other methods are required to be implemented.

A class named `Committee` that represents a group of professors.

- **Constructor**
  - Implement a constructor that initializes a `Committee` instance and takes in two arguments: the committee's name, and a list of `Professor` objects that are part of the committee.
- **Attribute(s)**
  - Define an attribute that represents the name of the committee
  - Define an attribute that keeps track of the `Professor` objects (i.e., the professors that are part of the committee).
- **Method(s)**
  - Implement a method called `size` that returns the number of professors in the committee.
  - Implement a method called `unique_fields` the determines the unique fields represented in the committee (i.e., the fields of all the professors in the committee, but without any duplicate fields). This method returns the unique fields as a list.
  - Implement a method called `num_tenured` that returns the number of tenured professors in the committee.

You may write additional helper methods and attributes; however, the above class definitions, constructor, attributes, and methods are **required** to be implemented and used in your solution.

As a remainder, your solution to this problem must be implemented using an object-oriented approach by defining the classes presented above. Any other solution will receive zero points, even if the testing system judges it correctly.

Input

The first input line is two integers separated by a single space; The first integer is P ($1 \leq P \leq 50$), which represents the number of professors at the university. The second integer is C ($1 \leq C \leq 100$) that represents the number of committees at the university.

The input is followed by P lines, each corresponding to a single professor. A professor line contains the professor's name, a single letter string that is either `"T"`, which means the professor is tenured or `"F"`, which means the professor is not tenured. The remainder of the line are F ($1 \leq F \leq 20$) number of strings that represent the fields of study for the professor. Each name and field contains only lowercase letters, no whitespace, and has a maximum length of 50 characters. You can assume the professor name will be unique for this problem and there are no duplicated field strings for a given professor. Each line component is separated by a single space.

The remaining input is C lines, each corresponding to a single committee. A committee line contains the committee's name followed by N ($1 \leq N \leq P$) number of strings that are professor names. Each committee name, professor name contains only lowercase letters, no whitespace, and has a maximum length of 50 characters. You can assume the committee name will be unique for this problem. Each line component is separated by a single space.

Output

The output is K lines of the committee names that represent a *bad* committee, as described above. The K lines must be sorted by committee name.

If there are zero *bad* committees then output `NONE`.

Sample Input 1

```
3 1
bob T systems theory
sally T theory
ally T ai
c1 bob sally ally
```

Sample Output 1

```
NONE
```

Sample Input 2

```
7 3
bob T systems theory
sally T theory
ally T ai
rob F systems
jenny F systems ai
ben F systems
stephanie F theory
c9 bob sally ally ben
c12 bob sally rob stephanie
c2 bob rob jenny stephanie
```

Sample Output 2

```
c2
```

Sample Input 3

```
7 4
bob T systems theory
sally T theory
ally T ai
rob F systems
ben F systems
jenny F systems ai
stephanie F theory
c5 ally rob ben jenny
c4 bob sally rob
c6 bob sally rob ben
c3 bob rob jenny stephanie
```

Sample Output 3

```
c5
c6
```