Supplying the Suppliers

It's battle time yet again, and now General Frey is worried about the supplies to his units. The capital city of the Empire has two exits, the North Gate and the South Gate, at which several key elements to the army will be stored. Several squads are going to be deployed, and seeing as each one of them has different needs, the General needs to decide on which exit he will deposit each supply type. Storing a supply element on both exits is costly and confusing, so the whole shipment for a resource must be located at a single exit.

The orders that specify which soldier should get which supply should not be disrespected, that is, two soldiers from the same squad cannot swap items if it's more convenient to do so. To solve this issue, the General has decided that a simple rule should be followed: No squad should ever have more than one soldier carrying an element from the opposite exit in which he is located. Given the list of squads and their needs, find out if it's possible to place the items on the exits such that the rule's followed.

Input

The input consists of several test cases. On the first line of each test case will be an integer N ($1 \leq N \leq 100$), the number of squads. Each of the following N lines begins with an integer t, describing how many needs that specific squad has. t integers and characters follow, each of which specify a supply type that squad needs. You may assume a supply type fits in a signed 32-bit integer. If the character is 'S', then the soldier that is going to fetch that item is at the South Gate, if it is 'N' the soldier is at the North Gate. Equal integers specify equal supply types, and no item appears twice on the same squad list. The number of different supply types in a single testcase is at most 100.

The last test case will be followed by a line containing a single zero.

Output

Print a single line for each test case. The line must contain the words "March onward" if it's possible to assign the deposits and follow the General's rules, "Coordination issue" otherwise.

Example

Input:
2
2 1S 2S
2 1N 2N
2
3 1S 2S 3S
3 1N 2N 3N
0

Output:
March onward
Coordination issue