OOM Assignment 4

1. The mid-semester copies need evaluation. The questions consist of 3 types: Objective (MCQ with text question string and answer as a letter A/B/C or D), Subjective (Text string question and text string answer) and specialized (which is same as subjective with answers consisting of a list of references). The instructor evaluates each copy and gives marks. Give the total marks obtained by all the student roll-no wise.

Input Format
The first input is the number of test cases. Each test case starts with *q* (the number of questions), *n* (the number of students). Thereafter there are *q* lines, each denoting a question, consisting of ID, question type, text and maximum marks. Thereafter there are *n* lines, one for each student, denoting student roll number, name and number of questions attempted (*a*). The next *a* lines mention question ID and answer as per format. The specialized answers have *r*, the list of references followed by *r* strings in addition. The instructor's evaluation are separately available. The number of lines are sum of number of questions attempted. Each line mentions the student roll number, question ID and marks given by the instructor. Marks over the maximum should be limited to maximum and negative marks should be made 0.

Output Format
n lines, each printing the roll number, name and marks obtained by the student, roll number wise.

Sample Input

Number of Test Cases	1
Number of questions and students	3 2
Question 1	Q1 objective text1 5
Question 2	Q2 subjective text2 10
Question 3	Q3 specialized text2 15
Student 1	R1 S1 2
Student 1, questions attempted	Q1 A Q2 Answer1
Student 2	R2 S2 2
Student 2, questions attempted	Q2 Answer2 Q3 Answer3 3 Ref1 Ref2 Ref3
Instructor's marks	R1 Q2 5 R2 Q2 3 R1 Q1 6 R2 Q3 8

Sample Output
R1 S1 10
R2 S2 11

(**Note:** Storing all information given is mandatory)

2. In Q1, there is a confusion over the use of LinkedList, Array, SortedArray, Tree and HashMap. This is given as an input in the question.

Input Format and Sample Input: Add either of LinkedList, Array, SortedArray, Tree and HashMap in the first line of every question (say 3 2 LinkedList).

Output Format and Sample Output: Same as above

3. The evaluation is done with the help of TAs. The TAs are of 3 types

- *beginner*, who can evaluate only objective questions. They have a roll number and name.
- *novice,* who can evaluate only objective and subjective questions. They have a roll number, name and a course of study (string) relevant to the exam.
- *advanced,* who can evaluate subjective and specialized questions. They have a roll number, name and a list of projects (each string) relevant to the exam.

The instructor divides the students and questions among the TAs. The instructor takes students roll number wise (primary) and questions as per ID (secondary) one after the other, and out of all eligible TAs, assigns the question to the TA with the least number of questions so far. In case of a tie, a TA with a smaller roll number is preferred.

First print the marks for each student roll number wise. Then for each TA print the roll number, name and the list of copies evaluated by the TA in the order of evaluation. The list of TA must be printed roll number wise.

Sample Input

Number of Test Cases	1
Number of questions, students and TAs	3 2 3
Question 1	Q1 objective text1 5
Question 2	Q2 subjective text2 10
Question 3	Q3 specialized text2 15
Student 1	R1 S1 2
Student 1, questions attempted	Q1 A Q2 Answer1
Student 2	R2 S2 2
Student 2, questions attempted	Q2 Answer2 Q3 Answer3 3 Ref1 Ref2 Ref3
TA 1	beginner TA1 Name1
TA 2	novice TA2 Name2 IOOM
TA 3	advanced TA3 Name3 3 GUIProject1 OOMProject2

	Project3
TA marks in the order of assignment	6 8 4 8

Sample Output

R1 S1 13
R2 S2 12
TA1 Name1
R1 Q1
TA2 Name2
R1 Q2
TA3 Name3
R2 Q2
R2 Q3

Explanation

TA assignment

Student	Question	Eligible TAs	Chosen TA	Marks	Actual Marks
R1	Q1	TA1, TA2	TA1	6	5
R1	Q2	TA2, TA3	TA2	8	8
R2	Q2	TA2, TA3	TA3	4	4
R2	Q3	TA3	TA3	8	8

(**Note:** The logic behind which TA type can evaluate which question type should be handled using interfaces. Storing all information given is mandatory)