2016 FHSPS Playoff
March 5, 2016
Cypress Bay High School

Filename	Problem Name
city	Host City
flagpoles	Flagpoles
paper	Ordering Paper
pass	Pass or Fail
rating	Rating Teachers
recovery	Correct Answer Recovery
sorting	Sorting Exams
teach	What to Teach?

Host City

Filename: city

Each June, some of the best AP teachers all gather in a single location to grade all of the AP exams for the year. The AP Board pays for the travel (flight or gas for driving) and hotel room for each out of town participant. (It's expected that in town teachers have lodging and drive a typical daily distance to get to the grading location, so no compensation is necessary.)

Naturally, the AP Board would like to minimize the amount they pay for the travel and hotel room for each out of town grader. They have a policy of always choosing a city with some AP teachers so that at least some teachers get to stay home. If the minimum cost can be achieved by picking different cities, then choose the city that comes first alphabetically.

Write a program to help the AP Board determine which city they should hold their grading session.

The Problem

Given the travel cost between all pairs of cities with AP teachers, the number of AP teachers from each city who need to travel to the grading session, and the cost of a hotel room for a single teacher (for the duration of the grading session) in each of the cities, determine the city that minimizes the amount the AP Board has to pay for the travel and lodging of out of town teachers, breaking ties by choosing the city that comes first alphabetically.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 50$), representing the number of grading sessions to process. The information for each grading session follows. The first line of input for each grading session contains a single positive integer, c ($c \leq 20$), the number of cities that have AP teachers included in the grading session. The next c lines will contain information about each of the cities, with the i^{th} ($1 \leq i \leq c$) line describing city i. Each of these lines will contain 3 pieces of information separated by spaces: s (a string of 1 to 15 uppercase letters), the name of the city, h ($h \leq 1000$), the cost of a hotel room for a single teacher in the city, and t ($t \leq 100$), the number of teachers in that city who will attend the grading session. All of the city names within an input case will be unique. This will be followed by c more lines of c space-separated integers, each. The j^{th} value on the i^{th} of these lines will represent the cost of a single teacher traveling from city i to city j. Note that the triangle inequality will be observed in these costs - it will never be cheaper to travel from city i to an intermediate city k and then from city k to city j, than to travel directly from city i to city j. The travel cost from a city to itself will always be 0 and the travel cost from city i to city j may be different than the travel cost from city j to city i. Each of the travel costs between distinct cities will be positive integers in between 1 and 1000, inclusive.

The Output

For each grading session, output on a line by itself, the name of the city that should host that grading session.

Sample Input

```
2
3
ORLANDO 500 20
MIAMI 800 30
TAMPA 600 25
0 200 110
250 0 300
110 300 0
2
JACKSONVILLE 500 5
FTLAUDERDALE 600 6
0 120
100 0
```

Sample Output

```
ORLANDO
FTLAUDERDALE
```

Sample Explanation

In the first sample case, if we were to host the grading session in Orlando,

it would cost $30 \times \$500 + 30 \times \$250 = \$22500$ for the Miami participants and
it would cost $25 \times \$500 + 25 \times \$110 = \$15250$ for the Tampa participants, for a total of \$37750.

If the session were hosted in Miami, the cost would be

$$20 \times \$800 + 20 \times \$200 + 25 \times \$800 + 25 \times \$300 = \$47{,}500$$

Finally, if the session were hosted in Tampa, the cost would be

$$20 \times \$600 + 20 \times \$110 + 30 \times \$600 + 30 \times \$300 = \$41{,}200.$$

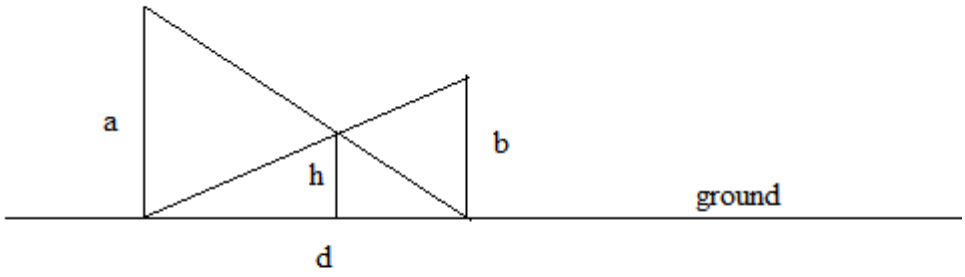
Thus, the session should be hosted in Orlando.

In the second sample case, it would cost $6 \times \$500 + 6 \times \$100 = \$3600$ for the Ft. Lauderdale teachers to go to Jacksonville and it would cost $5 \times \$600 + 5 \times \$120 = \$3600$ for the Jacksonville teachers to go to Ft. Lauderdale, an equal cost. Since there's a tie, the Ft. Lauderdale is chosen because it comes before Jacksonville, alphabetically.

Flagpoles

Filename: flagpoles

The AP teachers happen to be gathering near a meeting of the state mathematics teachers. On the exam the math teachers are grading, two of the teachers got in a heated argument about one of the problems. Realizing the superiority of the Advanced Programming teachers, the two math teachers came to the AP grading room in an effort to have the AP teachers settle the debate. The problem they wanted to solve involves two flag poles and two pieces of rope tied from the top of each flag pole to the base of the other flag pole. The following diagram illustrates the situation



In the problem, the heights of both flagpoles, a and b are given, as well as the distance on the ground between them, d. The ropes are the two diagonal lines in the illustration, that meet at a point somewhere in between. The argument the math teachers are having is calculating the height above the ground, h, at which the two ropes meet.

Write a computer program to solve their query correctly so that they can settle their dispute.

The Problem

Given the heights of two flagpoles and the distance between them on the ground, determine the height at which the two ropes illustrated above meet.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 10000$), representing the number of flag pole problems to solve. Each of the problems will follow, one per line. Each of these lines will contain three space-separated positive integers, a ($a \leq 1000$), b ($b \leq 1000$), and d ($d \leq 10000$), representing the heights of the two flagpoles and the distance between them, in feet, respectively.

The Output

For each problem, output the height, in feet, that the two ropes meet, rounded to three decimal places.

Sample Input

```
3
10 10 10
10 20 30
15 5 10
```

Sample Output

```
5.000
6.667
3.750
```

Ordering Paper

Filename: paper

Because the AP teachers are old school, they still insist on administering all of their exams on paper. Exam booklets consist of 17" x 11" in sheets of paper folded vertically down the middle. A booklet made of a single sheet can have writing on up to four pages, since each sheet folds into two parts and each part has two sides. The state would like to know how many sheets of the 17" x 11" paper to order based on the size of each examination booklet (in pages) and the number of students taking each exam.

The Problem

Given the number of different exams, the number of students taking each exam, and the number of pages necessary for each exam, calculate the fewest number of sheets of 17" x 11" paper the state can order to make all of their exams.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 20$), representing the number of years of data to process (number of input cases). The first line in each year of data will consist of a single positive integer, e ($e \leq 100$), representing the number of different exams given for that year. The following e lines will contain information about one exam each. Each of these lines will have two space-separated positive integers, s ($s \leq 10000$), representing the number of students taking that exam, and p ($p \leq 100$), the number of pages in a single exam booklet, for that exam.

The Output

For each year, output the total number of sheets of 17" x 11" paper that are needed, on a line by itself.

Sample Input

```
2
3
100 25
50 8
100 10
2
1000 100
2000 95
```

Sample Output

```
1100
73000
```

Pass or Fail

Filename: pass

A group of Florida teachers has decided that they need a round of exams to rate the state's programming students. After much deliberation, they've decided to call all of the exams Advanced Programming (AP) exams. The exam will be given in several different categories and scored from 1 to 10. It has been determined that a score of 6 or higher is a passing score. The teachers would like you to write a program that determines whether or not a student has passed an exam based on their score.

The Problem

Given a score a student earned on an Advanced Programming exam, determine whether or not they passed.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 20$), representing the number of exam scores to process. Each of the exam scores will follow, one per line. Each exam score will be a positive integer, in between 1 and 10, inclusive.

The Output

For each exam score, output either "PASS" or "FAIL" on a line by itself indicating whether or not the input score was a passing score or not.

Sample Input

```
5
2
10
6
5
8
```

Sample Output

```
FAIL
PASS
PASS
FAIL
PASS
```

Rating Teachers

Filename: rating

Rating teachers and performance pay is all the rage. While the Florida Association of Advanced Programming teachers doesn't have an answer for all secondary teacher woes, they believe they've come up with a great way to rate the performance of computer science teachers. As part of their initiative, they'd like you to produce a ranked list of teachers for each county in three categories of class size based on the exam performance of their students.

We define a small class to be 10 or fewer students, a medium class to be in between 11 and 30 students, inclusive, and a large class to be greater than 30 students. We only rank teachers who have small classes with other teachers who have small classes, and so on. To compare two teachers, we first compare their students' pass rates. If the pass rates aren't equal, the teacher whose students have the higher pass rate is ranked higher. Recall that AP exams are scored from 1 to 10, with scores of 6 or higher being considered passing scores. In the cases where the pass rates are equal, ties are broken by average exam score of the students, with the teacher whose students have the higher average exam score being ranked higher. The data will be such that no two teachers' students will have identical pass rates AND exam average scores. For the purposes of this problem we assume that each teacher has precisely one class.

The Problem

Given the scores of all of the students for each teacher in a county, create three ranked lists for the county, for the small classes, medium classes and large classes.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 100$), representing the number of counties for which to process exam scores.

The first line for the input for each county will have an uppercase string with the name of the county, followed by a space, followed by a positive integer, t ($3 \leq t \leq 20$), representing the number of AP teachers in the county.

The following t lines will contain input about each teacher in the county, with one line per teacher. Each of these lines will have 11 space separated pieces of information on them. The first will be the last name of the teacher. The next ten will be non-negative integers, a_1, a_2, ..., a_{10}, representing the number of students of that teacher who got a 1, 2, ..., 10, respectively on the AP exam. It's guaranteed that $1 \leq \sum_{i=1}^{10} a_i \leq 100$, namely, the number of students in each class will be in between 1 and 100, inclusive. It is guaranteed that the input for each county will contain at least one small class, one medium class and one large class. Finally, it is guaranteed that within a county, all of the teachers last names will be distinct and will be comprised of uppercase letters only.

The Output

For each county, output a header line with the name of the county followed by a space, followed by the string "COUNTY". On the second line of output for each county, put the header "SMALL

CLASS RANKING". On the following lines, put the last names of all the small class teachers of the county, one per line, in ranked order based on the criteria described in the problem description. Follow this with a blank line and then the header "MEDIUM CLASS RANKING". On the following lines put the last names of all the medium class teachers of the county, one per line, in ranked order. Follow this with a blank line and then the header "LARGE CLASS RANKING". On the following lines put the last names of all the large class teachers of the county, one per line, in ranked order.

Place one blank line after the output for each county.

Sample Input

```
2
LEON 3
SMITH 2 3 0 0 0 0 0 4 0 1
JONES 0 0 2 5 8 9 10 12 13 0
MURRAY 0 0 0 5 5 5 0 0 0 0
OSCEOLA 6
ADAMS 1 2 3 4 5 6 7 8 9 10
BARKER 10 9 8 7 6 5 4 3 2 1
CANADA 1 2 3 5 3 7 7 8 9 10
DAVIS 0 0 0 0 0 0 0 0 5 0
ELLIS 0 0 0 0 6 0 0 0 0 6
FRINKLE 0 0 0 0 10 0 0 0 1 9
```

Sample Output

```
LEON COUNTY
SMALL CLASS RANKING
SMITH

MEDIUM CLASS RANKING
MURRAY

LARGE CLASS RANKING
JONES

OSCEOLA COUNTY
SMALL CLASS RANKING
DAVIS

MEDIUM CLASS RANKING
ELLIS
FRINKLE

LARGE CLASS RANKING
CANADA
ADAMS
BARKER
```

Correct Answer Recovery

Filename: recovery

You've taken over for a former AP teacher who left haphazard records. You've got a copy of her students' True/False answers on last year's AP test, as well as the distribution of the scores of the students. Unfortunately, you don't know which student got which score. Based on this data, you want to see if you can reconstruct the correct answers to the test. Since it's likely that based on the data you have more than one possible set of answers could have been correct, you'll just settle for counting the number of different possible sets of answers that could have been correct.

The Problem

Given all of the students answers on a True/False test, as well as a distribution of their scores, determine the number of possible sets of answers that could have been the correct set of answers.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 100$), representing the number of classes to process. The information for each class follows. The first line of input for each class contains two space separated positive integers: q ($q \leq 15$), the number of questions on the test and s ($s \leq 40$), the number of students in the class. The next s lines will each contain a single string of length q, representing the answers a student in the class gave for the test. Each of these strings will only contain the characters 'T' and 'F', for true and false answers, respectively. The last line of each test case will contain q+1 space separated integers, $f_0, f_1, f_2, \ldots, f_q$, where f_i represents the number of students in the class who got exactly i questions correct, for $0 \leq i \leq q$. It is guaranteed that for all $0 \leq i \leq q$, $f_i \geq 0$ and $\sum_{i=0}^{q} f_i = s$. The input will be consistent, namely, there will exist at least one possible set of correct answers for the test for each test case.

The Output

For each class, output on a line by itself, the number of possible sets of answers that could have been the correct ones for the test.

Sample Input

```
2
4 3
FFFF
FTFT
TTTT
0 1 0 2 0
3 4
FFF
FFF
FTF
TTF
2 1 1 0
```

Sample Output

```
4
1
```

Sample Explanation

In the first test case, the correct answers could have been FFFT, FTFF, FTTT or TTFT. If the correct answers were either of the first two, the first two students listed would have gotten 3 correct and the last student would have gotten 1 correct. If FTTT were correct, then students 2 and 3 would have gotten 3 correct answers and the first student would have gotten 1 correct answer. Finally, if TTFT were the correct answers, then students 1 and 3 would have gotten 3 questions correct while student 2 would have gotten 1 answer correct.

In the second test case, all three answers must have been true. This is the only possibility that makes 2 students get all the questions wrong, 1 student get 1 question correct and 1 student get 2 questions correct.

Sorting Exams

Filename: sorting

In June of every year, a select group of the AP teachers get together to grade all of the exams. Since there are so many exams, the teachers are split into many groups and each group grades just some of the exams. After each group finishes, for record keeping purposes, the physical exams must all be sorted in alphabetical order. Naturally, each group already sorts their exams, thus, the last remaining step is to take several stacks of sorted exams and put them together into one sorted stack.

As part of the AP curriculum, you've learned how to merge two sorted stacks into one sorted stack. If the sizes of the two sorted stacks are x and y, respectively, you can produce the one sorted stack in $x+y$ steps. The AP teachers have asked you to devise an algorithm that repeatedly uses this technique to merge all of their stacks together into one sorted stack. Natually, they would like your algorithm to take as few steps as possible.

For example, imagine there are four stacks of 10 papers each. One method would be to merge the first two stacks, creating a stack of 20 papers. This takes 20 steps. Then, you could merge this newly created stack with one of the other stack, creating a stack of 30 papers. This takes 30 steps. Finally, you can merge the last two remaining stacks of sizes 30 and 10 into one sorted stack of size 40. This takes 40 steps. The total number of steps taken here is 20+30+40 = 90. Consider a different alternative for picking stacks to be merged in this example. Merge the first two stacks of size 10 into one stack of size 20. This takes 20 steps. Then, merge the last two stacks of size 10 into one stack of size 20. This takes 20 steps. Finally, merge the two stacks of size 20 together to form one sorted stack of size 40. This takes 40 steps. The total work here is 20+20+40 = 80 steps. Thus, for this particular set of papers, the optimal cost for merging them into one sorted stack is 80 steps.

The Problem

Given a list of the sizes of the stacks of sorted papers after the AP teachers have finished grading in their groups, calculate the minimum number of steps it will take to merge all of the stacks into one sorted stack, using the method described above where at each step, any two stacks are chosen to merge into one, over and over again.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 50$), representing the number of exam grading sessions to process. The information for each grading session follows. The first line of input for each grading session contains a single positive integer, g ($g \leq 500$), representing the number of grading groups for that grading session. The second line of input for each grading session will contain g space-separated positive integers, each less than or equal to 500, representing the number of papers graded by each group in the grading session.

The Output

For each grading session, output a single integer repsenting the minimum number of steps necessary to merge all of the sorted stacks of papers into a single sorted stack of papers.

Sample Input

```
3
3
1 2 3
4
1 2 3 4
3
1 3 2
```

Sample Output

```
9
19
9
```

What to Teach?

Filename: teach

It's the middle of the year and you've taken over for an AP teacher who wasn't terribly effective. You'd like to maximize your students scores, but when you teach a topic, you want to fully teach it. Luckily, after being on committee who creates the AP test for many years, you have a very good idea of how many points a student will get if they understand a particular topic. The only problem, of course, is that you have limited classroom time. Thus, you'll have to skip some topics, knowing that your students won't get those questions, only teaching the topics that will earn them the most points on the test. Write a program to help you determine which topics to teach and which topics to skip, so as to maximize your students' possible exam score. For the purposes of this question we assume that every AP test questions fits into precisely one topic.

The Problem

Given a list of how long (in minutes) it takes to teach each topic and how many points of questions pertain to that topic, as well as the total number of minutes left to teach in the semester, determine the maximum possible score of the students in the class if you choose to teach the appropriate topics.

The Input

The first line of the input file will contain a single positive integer, n ($n \leq 50$), representing the number of classes to process. The information for each class follows. The first line of input for each class contains two space separated positive integers: t ($t \leq 100$), the number of topics on the test and m ($m \leq 5000$), the number of minutes left to teach. The next t lines will each contain two space-separated positive integers: m_i ($m_i \leq 1000$), the number of minutes to teach topic i ($1 \leq i \leq t$), and p_i ($p_i \leq 100000$), the number of points on the exam pertaining to topic i ($1 \leq i \leq t$).

The Output

For each class, output on a line by itself, the maximum number of points the students can get on the exam, based on the best set of topics to be taught.

Sample Input

```
2
3 100
30 50
50 90
60 80
4 10
20 100000
3 22
6 17
100 1
```

Sample Output

```
140
39
```

Sample Explanation

In the first class, it takes 80 minutes to teach topics 1 and 2, which are collectively worth 140 points. In the second class, only topics 2 and 3 can be taught, worth 39 points total.