ACM-ICPC 2013
Asia Regional Contest
Kharagpur Site

IIT Kharagpur

December 15, 2013

You get:

10 Problems, 24 pages, 300 Minutes

This page intentionally left blank

<u>Rules for ACM-ICPC 2013 Asia Regional Kharagpur Site</u>

1. Solutions to problems submitted for judging are called runs. Each run is judged as accepted or rejected by the judge, and the team is notified of the results. Submitted codes should not contain team or University name and the file name should not have any white space.
2. The public rank list will not be updated in the last one hour. However, notification of accepted/ rejected runs will **NOT** be suspended at the last one hour of the contest time.
3. A contestant may submit a clarification request to judges. If the judges agree that an ambiguity or error exists, a clarification will be issued to all contestants.
4. Contestants are not to converse with anyone except members of their team and personnel designated by the organizing committee while seated at the team desk. **But they cannot even talk with their team members when they are walking around the contest floor to have food or any other purpose.** Systems support staff may advise contestants on system-related problems such as explaining system error messages.
5. While the contest is scheduled for a particular time length (five hours), the contest director has the authority to alter the length of the contest in the event of unforeseen difficulties. Should the contest duration be altered, every attempt will be made to notify contestants in a timely and uniform manner.
6. **A team may be disqualified by the Contest Director** for any activity that jeopardizes the contest such as dislodging extension cords, unauthorized modification of contest materials, distracting behavior or communicating with other teams.
7. Team can bring up to 25 pages of printed materials with them. But they are not allowed to bring calculators or any machine-readable devices like mobile phone, CD, DVD, Pen-drive, IPOD, MP3/MP4 players, floppy disks etc. Blank papers will be provided, if needed.
8. With the help of the volunteers and external judges, the contestants can have printouts of their codes for debugging purposes. Passing of printed codes to other teams is strictly prohibited.
9. **The decision of the judges is final.**
10. **Teams should inform the volunteers if they do not get reply from the judges within 10 minutes of submission. Volunteers will inform the External Judges and the external judge will take further action. Teams should also notify the volunteers if they cannot log in into the PC^2 system. This sort of complains will not be entertained after the contest.**
11. Ten problems will be posed. So far as possible, problems will avoid dependence on detailed knowledge of a particular applications area or particular contest language. All problems require you to **read test data from the standard input** as specified in the problem and **write results to the standard output**.

12. You can use any of the standard library functions that your chosen programming language provides. In addition, you can use the math library in C/C++. You cannot use any other library that requires an extra flag to be passed to the compiler command. If you do this, the judges will probably find a code "compilation error" in your program.
13. Your program is not permitted to invoke any external programs. For example, you cannot use in C the system call ("grep xyz abc") to determine whether the file abc contains an occurrence of the string xyz. *Violation of this rule may lead to disqualification from the contest.*
14. **Output must correspond exactly to the provided sample output format, including (mis)spelling and spacing.** Multiple spaces will not be used in any of the judges' output, except where explicitly stated.
15. Your solution to any problem should be submitted for judging using the PC^2 software only. Once you have submitted the solution, it will reach the judges. The time it takes for your problem to be judged will depend, among other things, on how busy the judges are. Once your submission has been judged, you will receive a message through PC^2 indicating the judgment. The judgment may be "Yes", meaning that your submission was judged to be correct. Otherwise you will get a message indicating the problem with your program. For example, the message may be "Incorrect Output", "Output Format Error", "Compilation Error", "Runtime Error", "Run Time Limit Exceeded" etc.
16. **Programming style is not considered in this contest. You are free to code in whatever style you prefer. Documentation is not required. The judges will only test whether the input-output behavior of your program is correct or not.**

Problem A: Election Campaign

Input: Standard Input
Output: Standard Output
Problem Code: Election

A district in a state has **K** villages (numbered from 1 to **K**). In order to provide better road connectivity in the district, the government has connected all the villages with two-lane paved roads, However, to reduce costs, the roads are constructed such that while it is possible to go from any village to any other village using the paved roads, there is exactly one route between any two villages. In addition, **M** of these villages are also connected to the district headquarter directly using two-lane roads.

Elections are round the corner and a leader of a political party wishes to hold road shows in the villages. Since he does not have time to visit all the villages, he decides to go from the district headquarter to one of the villages that is connected to it, and then follow a path using the roads between the villages only, holding one road show at each village in the path as and when he reaches that village. Since he has to return to the district headquarter, he wishes to hold the last road show in a village that is again connected to the district headquarter. To save time, he also does not want to go through any village more than once.

Obviously, he cannot go through all the villages. To ensure that people in other villages (not on the path) are not deterred from coming to his road shows, he asks his assistant to chalk a route such that the sum total of the distances of any village from its closest village in the path is minimized. The road show being still a week off, he asks his assistant to take all the time he wants, but make sure that he finds the correct path.

The assistant sets off to do this, trying to write a program to try out one-by-one all the different paths the leader can take and calculating the total distances people have to travel if a person in a village goes to the road show nearest to his village. Help him write this program.

Input:

The first line contains the number of test cases **N** (0 < **N** ≤ 3).

For each test case, the first line contains the number of villages **K** (0 < **K** ≤ 50). This is followed by **K** lines, with the j-th line containing the information about village j. The information for each village consists of (in this order): an integer **F** (0 if the village is not connected to the district headquarter, 1 if it is), an integer **D** indicating the number of other villages this village is directly linked to by road, followed by **D** pairs of integers (total

2**D** integers), with each pair containing the id (from 1 to **K**) of a village it is connected to and the distance (positive integer) from that village.

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the sum total of distance of all the villages from their closest village in the path chosen.

Sample Input	Sample Output
2 5 0 3 2 10 3 8 4 15 1 1 1 10 1 1 1 8 1 2 1 15 5 12 0 1 4 12 7 0 2 2 5 6 20 0 3 1 5 4 10 5 5 1 1 6 3 1 1 2 10 0 1 2 5 1 3 1 20 7 10 3 3 1 1 6 10	Case 1: 20 Case 2: 8

Problem B: Olympic Village

Input: Standard Input
Output: Standard Output
Problem Code: Olympic

It is almost Olympic time, but the organizers fear that the entire Olympic village will not be ready in time given the slow progress of construction. The games are going to have **K** events (numbered from 1 to **K**), and **M** countries (numbered from 1 to **M**) have already confirmed sending their sportspersons to participate in one or more of these events, with one person participating in exactly one event. Given the urgency of the situation, the organizers hold an emergency meeting to find the minimum number of total beds that they have to plan for counting all the rooms to accommodate all the people. Given that there will be some moving in and moving out time, it was decided that two people will not be allotted the same bed if the days their events are held overlaps (including if one event starts on the same day the other ends).

The organizers want to get that number and then want to speed up the work on a war-footing to at least get that many beds ready in time. You have to help them calculate this minimum.

Input:

The first line contains the number of test cases **N** (0 < **N** ≤ 3).

For each test case, the first line contains the values of **K** (0 < **K** ≤ 30) and **M** (0 < **M** ≤ 100). This is followed by **K** lines, with the j-th line containing the start and end days of event j (both positive integers, with end day > start day). This is followed by **M** lines, with the j-th line containing the participation information for country j. The participation information for each country consists of (in this order): an integer **D** (0 < **D** ≤ **K**) indicating the number of events the country is participating in, followed by **D** pairs of integers (total 2**D** integers), with each pair containing the id (from 1 to **K**) of an event and the number of participants (positive integer) the country is sending for that event.

Output:

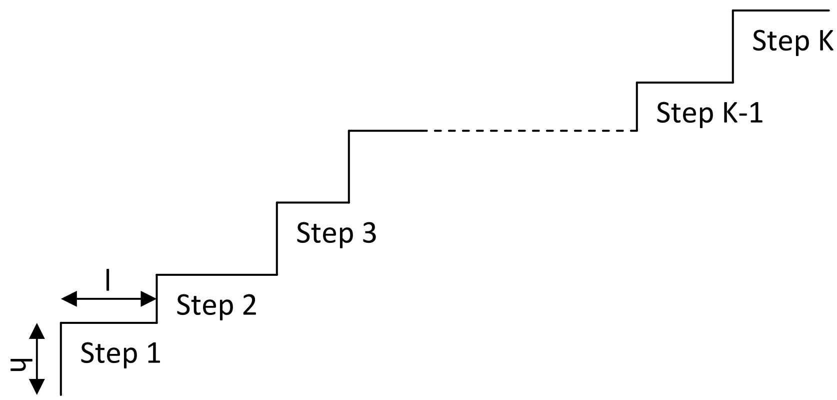
For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the minimum total number of beds needed.

Sample Input	Sample Output
2 5 3 2 7 1 4 5 8 3 6 9 10 5 1 1 2 1 3 1 4 1 5 1 4 1 1 2 1 3 2 5 2 3 3 1 5 1 4 1 4 2 1 3 3 6 2 3 4 7 4 1 2 2 2 3 1 4 1 4 1 1 2 1 3 2 4 2	Case 1: 8 Case 2: 9

Problem C: Stairway Reduction

Input: Standard Input
Output: Standard Output
Problem Code: Stairway

A garrison of soldiers is stationed at on old fort built long ago by a king located on the top of a hill. The fort can be accessed only by a cemented stairway with **K** steps (steps are numbered 1 to **K** from the bottommost to the topmost as shown below). The width of all the steps in the stairway is the same (**w**), but the length (**l**) and the height (**h**) of each step is made different to take care of the difference in the slope of the hill at different places along the climb.



Everyday, supplies are delivered to the soldiers stationed at the fort by laborers carrying the supplies on their back. To his utter dismay, the commander finds that the laborers are charging by the number of steps they have to climb, and not by the distance travelled or height climbed. Since the fort is at a strategic location and will be occupied for long, the commander decides to remove some of the steps by filling them up with concrete. Keeping in mind how high the fort is at, the commander decides to convert the stairway to **M** steps (0 < **M** < **K**). However, he knows he has to do this with the limited budget that he has, and wishes to minimize the total extra concrete he has to pour to reduce the steps. Given the common width of all the steps, and the length and height of each step, all in inches, can you help the commander find the minimum amount of concrete (in cubic inches) he has to plan for?

Input:

The first line contains the number of test cases **N** (0 < **N** ≤ 3).

For each test case, the first line contains **K** ($0 < K \leq 500$), **M**, and the common width **w** (positive integer). This is followed by **K** lines, with the j-th line ($1 \leq j \leq K$) containing the length **l** and height **h** of the j-th step (both positive integers).

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the total amount of concrete needed (in cubic inches).

Sample Input	Sample Output
2 3 2 48 18 12 24 14 16 10 7 3 24 12 12 24 24 30 30 36 36 18 22 18 22 32 18	Case 1: 11520 Case 2: 57888

Problem D: Building Township

Input: Standard Input
Output: Standard Output
Problem Code: Township

A real estate company undertakes to build a large township. The work is complex and requires the company to hire different categories of highly skilled laborers at different times for different parts of the work. An analysis of the whole project showed that **K** different groups of highly skilled laborers are needed for **K** subparts of the project, each of which takes a certain number of days. The subparts are numbered from 1 to **K**.

The whole work is supervised by a consultant of the company and he can supervise only one subpart at a time, so only one group of skilled laborers is actually utilized at one time. However, it is very difficult to find skilled labor just when you need them, so in order to avoid delays due to non-availability of required labor when it is needed, the company decides to hire all skilled laborers at the start of the project itself, paying them from the beginning of the project till their subpart is done, at which time, they are let go. Each such group costs the company a certain amount of money per day from the day they are hired till the day they finish their part.

The situation is compounded by the fact that some of these subparts have to be done in order. However, it is seen that any subpart can be dependent on at most one other subpart and can cause at most one other subpart to be dependent on it.

What is the minimum total amount the company has to pay to the laborers?

Input:

The first line contains the number of test cases **N** (0 < **N** ≤ 3).

For each test case, first line contains the number of subparts **K** (0 < **K** ≤ 200). Second line contains a sequence of **K** integers indicating the number of days required to finish each subpart. Third line contains a sequence of **K** integers indicating the hiring cost per day for the labor group needed for each subpart. The fourth line contains an integer **D** indicating the number of pairs of subparts where one subpart is dependent on the other. This is followed by **D** lines, each of which contains two integers **x** and **y** (in that order), where subpart **y** depends on subpart **x** (i.e., **y** can start only after **x** has finished).

Output:

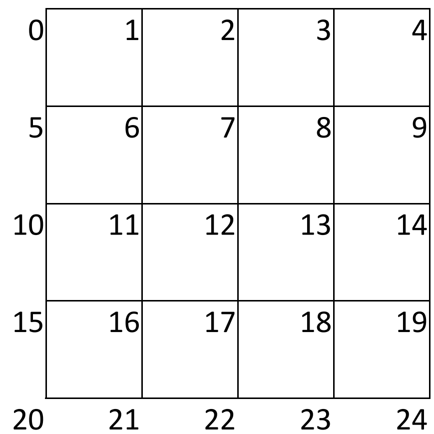
For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the total labor cost paid by the company.

Sample Input	Sample Output
2 5 21 27 12 33 5 100 200 50 25 75 3 1 2 3 5 5 4 4 10 10 10 10 40 30 20 10 0	Case 1: 21125 Case 2: 2000

Problem E: One-Way Roads

Input: Standard Input
Output: Standard Output
Problem Code: Road

A city has a road network with all roads arranged in a **K** x **K** square grid, with intersections (total (**K**+1) x (**K**+1)) numbered as shown below (for an example 4 x 4 grid).



There are no one-way roads in the city now. For ease of movement of cars, the traffic department plans to convert all roads to one-way, while ensuring that any intersection can be reached from any other intersection. However, the fire brigade is located beside intersection **A**, and a set of factories with high fire risks are located in a set of intersections **J**. It is known that no intersection in **J** can be reached from **A** without making a turn somewhere, irrespective of which roads are made one-way or the route followed. The fire brigade has requested the traffic department to convert the roads to one-way in such a manner such that fire trucks starting from **A** can reach all of the intersections in **J** passing through the minimum total number of intersections possible to avoid delay. How should the traffic department decide which road to set which way?

Input:

The first line contains the number of test cases **N** ($0 < N \leq 3$).

For each test case, the first line contains **K** ($1 < K \leq 100$), the dimension of the grid. The second line contains the intersection number **A** (an integer indicating the intersection number in the numbering scheme shown). The third line contains the number of elements in **J** ($1 \leq |J| \leq K^2$), followed by the intersection numbers of the elements in **J** (|**J**| integers indicating the intersection numbers in the numbering scheme shown).

Output:

For each test case, in the first line, print the case number, followed by a colon, followed by a single space, followed by the total number of intersections passed to reach all elements of **J** from **A**, **not counting intersection A or the intersections in J**. Then print 2**K**(**K**+1) line, one for each possible edge in the grid. Each of these lines should contain (in this order): an integer **x**, an integer **y**, and an integer **d** from the set {0,1} where **x** and **y** are valid intersection numbers, and **d**=0 if the one-way is set from **x** to **y** or **d**=1 if it is set from **y** to **x**. The two nodes in an edge can be printed in any order as long as the correct d is printed. Similarly, the edges can be printed in any order.

Sample Input	Sample Output
1 2 4 2 6 2	Case 1: 2 0 1 1 4 5 0 2 1 1 6 7 1 1 4 1 2 5 0 3 4 0 7 4 1 0 3 0 5 8 0 3 6 1 7 8 1

Problem F: Ant Colony

Input: Standard Input
Output: Standard Output
Problem Code: Ant

In an ants' colony spread over a flat surface, the ants can reside at integral coordinates as close as unit distance apart from each other (but no closer). The colony has the shape of a quadrilateral **ABCD** where one ant must reside at each of **A**, **B**, **C**, and **D**, and all other ants can be on any integral coordinate on its perimeter and its interior. Given the integral positions of **A**, **B**, **C**, and **D**, print the maximum number of ants that can reside in the colony (including the ants that can reside on the four corners and the perimeter).

Input:

The first line contains the number of test cases, **N** (0 < **N** ≤ 3).

For each test case, a single line contains the **x** and **y** coordinates of the four corners of the quadrilateral in clockwise order starting with the left-most, bottom-most point.

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the maximum number of ants.

Sample Input	Sample Output
2 1 1 3 4 5 3 6 1 1 3 5 6 6 3 3 1	Case 1: 14 Case 2: 16

This page intentionally left blank

Problem G: Message Coding

Input: Standard Input
Output: Standard Output
Problem Code: Message

You have a mini number pad which allows you a shortcut mode of typing messages using the digits 0 to 9. Each digit represents a subset of English letters (lowercase only). No letter is represented by more than one digit. An example keypad is given below along with the corresponding letters. Unfortunately there is no representation for blank or any other separator for words.

0: a e u	5: f p z
1: b l v	6: g q
2: c m w	7: h r
3: d n x	8: k s
4: i o y	9: j t

You have a complete dictionary of words (with lowercase letters only). You may use the keyboard to type a sequence of words. For example, the words "code competition final" is entered as "2 4 3 0 2 4 2 5 0 9 4 9 4 4 3 5 4 3 0 1". This sequence may correspond to several different word sequences. Some digit sequences may not correspond fully to any word sequence.

You are required to print the minimum number of words in sequence from the dictionary that covers the given input. If the given digit sequence does not correspond to any complete word sequence in the dictionary, output -1.

Input:

The first line contains the number of test cases, **N** (0 < **N** ≤ 3).

For each test case, first line contains an integer **K** (0 < **K** ≤ 10000) indicating the number of words in the dictionary. Second line contains **K** words, each with length less than 20. This is followed by 10 lines, with each line containing the following: a digit **d** (0 ≤ **d** ≤ 9), followed by an integer **m** (0 < **m** ≤ 15), followed by **m** lowercase alphabets (with one space between two characters) indicating the English alphabets represented by the digit **d**. This is followed by a single line containing an integer **n** (0 < **n** ≤ 1000), followed by **n** integers (each within 0 and 9) denoting an n-digit input sequence to check.

Output:

For each test case, in the first line, print the case number, followed by a colon, followed by a single space, followed by a single integer representing the minimum number of words in the dictionary that can cover the given digit sequence (the integer should be -1 if the given digit sequence does not correspond to any complete word sequence).

Sample Input	Sample Output
2 18 am axe exam boy colour dam dot donkey fox new prim prime primeval pry the this theory van 0 3 a b c 1 3 d e f 2 2 g h 3 3 i j y 4 3 l m n 5 3 o p q 6 2 r s 7 2 t u 8 3 v w x 9 2 k z 11 7 2 1 5 6 3 4 1 8 0 4 20 am ant axe boy colour dam dot donkey exam exact fox new prim prime primeval pry the this theory van 0 3 a b c 1 3 d e f 2 2 g h 3 3 i j y 4 3 l m n 5 3 o p q 6 2 r s 7 2 t u 8 3 v w x 9 2 k z 5 2 4 3 0 2	Case 1: 2 Case 2: -1

Problem H: Toy Exchange

Input: Standard Input
Output: Standard Output
Problem Code: Toy

During the pre-puja bonanza, the Khelnacompany has put **K** different types (numbered from 1 to **K**) of Disney figurines in their hot-selling RC cars. You can find the Disney figurine only after you have bought a RC car and opened the box. Sumana just loves Khelnacompany's cars and the Disney characters, and wants to collect as many different Disney figurines as possible.

So Sumana tries to trade some of the figurines she has with her **M** friends to get figurines that she does not have. Any of Sumana's friends (say X) will trade a figurine with Sumana if and only if X has a duplicate of that figurine, and Sumana gives X a figurine X does not have. Moreover, Sumana's friends are not as crazy as Sumana about collecting figurines, so they don't exchange any figurines among themselves; they just exchange with Sumana to make her happy.

Sumana is clever and has realized that in some cases it might be good to exchange a Disney figurine she has for another figurine of a type she already possesses, or trade off a figurine even if she has only one copy of it. Given that Sumana knows exactly how many figurines of each type each of her friends has, can you help her find the maximum number of different Disney figurines that she can get by trading figurines with her friends?

Input:

The first line contains the number of tests cases **N** ($0 < N \le 3$).

For each test case, the first line contains the value of **K** ($0 < K \le 100$) and **M** ($0 < M \le 100$). The second line contains **K** integers, with the j-th integer indicating the number of figurines of type j that Sumana has. This is followed by **M** lines, one for each of Sumana's **M** friends. Each of these lines contains **K** integers, with the j-th integer indicating the number of figurines of type j possessed by that friend. Note that Sumana or a friend may not possess any figurine of a particular type, in which case that number will be 0.

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the maximum number of Disney figurines Sumana can collect.

Sample Input	Sample Output
2 3 1 3 0 0 1 3 3 4 2 3 3 0 0 0 0 3 0 1 1 0 2	Case 1: 1 Case 2: 4

Problem I: Sequence Alignment

Input: Standard Input
Output: Standard Output
Problem Code: Alignment

Consider two sequences **A** and **B** comprising of digits 0 to 9, of length **K** and **L** respectively. The distance from **A** to **B** is the total minimum penalty incurred in converting **A** to **B** using the following operations, each of which incurs a penalty as described below.

1. Replacing a digit of **A** with another digit: incurs a penalty of 1
2. Inserting a digit in **A**: incurs a penalty of 1 for each digit inserted. In addition, there is also a fixed penalty of **x** for every sequence of digits inserted (one-time penalty for starting the sequence, not per-digit). However, the total penalty charged for a sequence of digits inserted never exceeds **y** (**y** > **x**). For example, if **x** = 2 and **y** = 5, then inserting a one digit sequence (say "5") will incur a penalty of 3 (=**x**+1), a 2-digit sequence (say "24") will incur a penalty of 4 (=**x**+2), and inserting any n-digit sequence (n > 2) (say "132", 2543", "57241" etc.) will all incur a penalty of 5 (=**y**).
3. Deleting a digit from A: incurs a penalty of 1 for each digit deleted. In addition, there is also a fixed penalty of **x** for every sequence of digits deleted (one-time penalty for starting the sequence, not per-digit).

What is the minimum penalty with which **A** can be converted to **B**?

Input:

The first line contains the number of tests cases **N** (0 < **N** ≤ 3).

For each test case, the first line contains the integers **K** (0 < **K** ≤ 100) and **L** (0 < **L** ≤ 100). The second line contains the sequence **A** (**K** digits with no space in between). The third line contains the sequence **B** (**L** digits with no space in between). The fourth line contains the integers **x** and **y** (**x** > 0, **y** > **x**).

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer representing the minimum penalty.

Sample Input	Sample Output
3 9 1 123456789 1 10 15 1 9 1 123456789 10 15 11 4 12345678912 1212 10 15	Case 1: 18 Case 2: 15 Case 3: 17

Problem J: Nomad Camp

Input: Standard Input
Output: Standard Output
Problem Code: Camp

A nomadic group wants to make camp inside a large forest having thousands of trees - all very tall, slick, and upright, with leaves and branches much above man-height. They will pitch a single tent, with the canvas of the tent conical in shape, and the tent has three long ropes fitted with the canvas-base that needs to be tied up with the bases of three trees so that the tree-bases form an acute-angled triangle (T). The circumcircle (C) of the points in T form the camp-base within which the nomads will set up all their other activities, so they naturally want to make their camp-base as large as possible and free of any other trees, although there may be some trees standing on the circumference of C. However, the nomads are all deadly against cutting any tree. Imagine that you are the nomad leader and have got the map of the forest. From this map, you find that the forest belongs to a planar land surface, and you also get the coordinates of all the tree bases. How can you compute the area of the largest camp-base that can be made?

Input:

The first line contains the number of test cases **N** ($0 < \mathbf{N} \le 3$).

For each test case, the first line contains the number of trees **X** ($0 < \mathbf{X} \le 5000$) followed by **X** lines, each containing the **x** and **y**-coordinate (in that order) of one tree (both positive integers).

Output:

For each test case, print the case number, followed by a colon, followed by a single space, followed by a single integer indicating the area of the largest camp base. The integer should be the floor of the area computed (for example, if the area computed is 123.7, output 123). If no camp can be made satisfying the given conditions, print -1 for the area.

Sample Input	Sample Output
1 4 1 1 10 1 9 5 5 4	Case 1: 26

END OF PROBLEMS