The 2024 ICPC Asia Topi On-line Preliminary Programming Contest

Instructions

- Do not open the booklet unless you are explicitly told to do so. You can only read these instructions below.
- Do not create disturbance or move around unnecessarily in the arena.
- If you have any question regarding the problems, send a clarification from the judges using DOMJudge.
- There would be no internet access and mobile phones are also not allowed.
- Before submitting a run, make sure that it is executable via command line. For Java, it must be executable via "javac" and for GNU C++ via "g++". Java programmers need to remove any "package" statements and source code's file name must be the same as of main class. C++ programmers need to remove any getch() / system("pause") like statements.
- Do not attach input files while submitting a run, only submit/attach source code files, i.e., *.java or *.cpp or *.py.
- Language supported: C/C++, Java and Python3
- Source code file name should not contain white space or special characters.
- You must take input from Console i.e.: Standard Input Stream (stdin in C, cin in C++, System.in in Java, stdin in Python)
- You must print your output to Console i.e.: Standard Output Stream (stdout in C, cout in C++, System.out in Java)
- Please, don't create/open any file for input or output.
- Please strictly meet the output format requirements as described in problem statements, because your program will be auto judged by computer. Your output will be compared with judge's output byte-by-byte and not tolerate even a difference of single byte. So, be aware! **Pay special attention to spaces, commas, dots, newlines, decimal places, case sensitivity etc.**
- Unless mentioned in some problem, all your programs must meet the time constraint of 5 seconds.
- The decision of judges will be absolutely final.
- ***Plagiarism or use of Generative AI tools is strictly prohibited and will result in immediate disqualification.***

Problem 1: Network Planning

Time limit: 3 seconds

The administration of a university is constructing new departments and hostels and plans to deploy a robust network infrastructure to ensure seamless internet connectivity for all students. The campus network will connect multiple hostel buildings through a series of cables, represented by an undirected graph where intersections are connection points, and edges are communication cables.

To ensure optimal performance, the university wants to install a central router at an intersection such that the average communication delay (i.e., distance through cables) from this router to all other hostels is minimized. Your task is to help the university determine the best location for the central router.

Input

The first line contains an integer for number of test cases. For each test case the first line contains two integers, N *(2 ≤ N ≤ 1500)* and M *(1 ≤ M ≤ 5000)*, representing the number of hostel buildings and number of transmission cables, respectively.

The next M lines contain two integers and one real number each, U, V *(1 ≤ U, V ≤ N)*, indicating a cable that connects two buildings U and V, followed by D_{uv} indicating the distance between the buildings U and V.

Output

Output the identifier of the university building where the central router should be placed, minimizing the average communication delay to all other buildings. If multiple buildings have the same delay, select the smallest building identifier.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
1 6 6 1 2 833 1 3 673 2 4 57 3 4 473 4 5 944 4 6 36	4

Problem 2: Architectural Design

Time limit: 1 seconds

A landowner sought out that the most beneficial business would be to develop his land into commercial buildings. Each plaza should hold a few brands and also facilitates the workers by providing them accommodation in those plazas. With that in mind, he held design competitions and architects from all over the world presented their 3D models.

The landowner, being found of art and symmetry in designs, rejected all of the presented designs and asked the architects to have symmetry among buildings. He explained that if a building of a design $\boldsymbol{d}$, is at one end of the road, a building with the same design $\boldsymbol{d}$, should be at the other end of the road. If a design is unique and no other building with the same design exists, move that to the center.

Luckily, all the presented designs had one or more similar designed building in their 3D models but were scattered along the road. Moving 3D models incurs an extra cost. Therefore, the architects are seeking your expertise in developing a solution which may return the count of minimum changes required to make their design symmetrical. If no movement is required, and the design already fulfill the requirements, return 0.

Example:
Input: ccxx
Output: 2
Such that one possible design is xccx which requires minimum of two movements.

Input

The first line of the input consists of $\boldsymbol{t}$, $(\mathbf{1} \leq \boldsymbol{t} \leq \mathbf{100})$ representing the total number of test cases given. The next subsequent t lines contains test cases with n of buildings, $(\mathbf{1} \leq \boldsymbol{n} \leq \mathbf{2000})$, represented as small alphabets of English language.

Output

Output consists of $\boldsymbol{t}$ lines, the number on each line is the minimum number of movements required to make the design symmetrical. If no solution exists, print **-1**.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	**Sample Output**
4 arcacer ppp eggeekgbbeg letelt	4 0 7 2

Problem 3: Library Stack Organization

Time limit: 1 seconds

A librarian needs to organize three stacks of books to ensure that each stack contains books of only one genre. The goal is to minimize the number of moves required to rearrange the stacks in this way.

Each stack currently contains a mix of three genres:

- ***S***: Science
- ***F***: Fiction
- ***H***: History

You are given the initial count of each genre in each stack. For instance:

- ***S1, F1, H1*** represent the number of Science, Fiction, and History books in Stack ***1***, respectively.
- ***S2, F2, H2*** for Stack ***2***.
- ***S3, F3, H3*** for Stack ***3***.

The goal is to organize the stacks so that:

- One stack contains only Science books.
- Another stack contains only Fiction books.
- The remaining stack contains only History books.

Your task is to find the arrangement that minimizes the total number of moves required to achieve this configuration, where moving one book from one stack to another counts as one move.

Input

First line contains the number of test cases (t) $1<t<=10^5$. For each test case a single line contains nine integers: ***S1 F1 H1 S2 F2 H2 S3 F3 H3***. Each integer represents the number of books in the corresponding genre for each stack $(0 \leq books \leq 100)$.

Output

The final sequence of the stack, followed by the number of minimum moves required to achieve that arrangement of the stack.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
2 1 1 2 3 5 8 10 3 7 4 4 4 4 4 4 4 4 4	HSF 21 FHS 24

Problem 4: Charging your Electric Vehicle in Manhattan

Time limit: 2 seconds

Manhattan is the most densely populated of New York City's five boroughs. It is a famous for its 'grid-like' structure with streets crossing top to bottom and left to right resulting in several blocks. Due to its high density and vertical living resulting from some of the highest skyscrapers in the world, traffic congestion and pollution are some of the major problems of the borough.

The city's new mayor has come up with a brilliant solution to reduce congestion, promote sustainable living, and reduce its carbon footprint. Since all blocks are connected with other blocks on the left, right, top, or bottom, you can use any path to cross Manhattan. The mayor has suggested that all streets linking different blocks be replaced with pressure sensitive material such that when cars drive through these roads, it generates and stores electricity at the center of each block. To discourage pollution, electric vehicles driving on these roads can charge themselves from this stored energy, free of cost when they pass through the center point of that block. However, to discourage more traffic by people trying to leverage this free electricity and roaming around those blocks endlessly, all cars can only make down and right turns.

Your daily commute is such that you need to traverse Manhattan while going to office. You enter the top-left block and exit the bottom-right block. To control traffic, the city adjusts the amount of energy you can receive while going through a particular block and these values are published online. Assuming the borough has $\boldsymbol{N} \times \boldsymbol{M}$ blocks, plan you path through Manhattan such that you benefit from maximal free charging as you traverse the borough according to the published energy for each block at that time.

Input

The first line contains an integer as number of test cases. For each test first line contains two integers, $\boldsymbol{N}$ and $\boldsymbol{M}(\mathbf{1} \leq \boldsymbol{N}, \boldsymbol{M} \leq \mathbf{1000})$, representing the dimensions of Manhattan, i.e., $\boldsymbol{N}$ blocks downwards and $\boldsymbol{M}$ blocks from left to right. The next $\boldsymbol{N}$ lines each contain $\boldsymbol{M}$ integers, where the ***j-th*** integer in the ***i-th*** line, $\boldsymbol{A}[\boldsymbol{i}][\boldsymbol{j}](\mathbf{0} \leq \boldsymbol{i}, \boldsymbol{j} \leq \mathbf{1000})$, represents the charge available while traversing $\boldsymbol{block}(\boldsymbol{i}, \boldsymbol{j})$.

Output

Output a single integer, the maximum charge you can get by the time you exit from the bottom-right corner.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	**Sample Output**
1 3 3 1 3 1 1 5 1 4 2 1	12

Problem 5: Pascal Key Breaker

Time limit: 2 seconds

Binomial Theorem is very useful tool in algebra and statistics. It is believed that this theorem was first introduced by a Persian and Chinese mathematician back in 13th century. The number of terms in the expansion is always $n + 1$ and the value of each term coefficient change with respect to its position in the expansion. For a higher value of n, the binomial expansion can be calculated as

$$(x + y)^n = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} y^k$$

Where $\binom{n}{k}$ represent the combinations and can be calculated as below:

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

Recently, Mr. Wang was on an exploration trip in Bursa, an ancient city in Turkey. He found an interesting cupboard with several small cabinets arranged in a pyramid shape. Each cabin has a special lock and can be open with a preassigned key. Mr. Wang quickly realized that the pyramid shaped cupboard follows the famous Pascal's triangle arrangement, where each row follow the binomial expansion. As a computer programmer, Mr. Wang asked your help to find the key to the corresponding lock for the given values of x, y, and n . Further, as a helping and kind person Mr. Wang also give you a hint that the key to each cabin is the r-th term (1-indexed) of the binomial expansion according to the position of the cabin in the pyramid, calculated as

$$T_r = \binom{n}{r-1} x^{n-(r-1)} y^{r-1}$$

Where ***r-th*** term is 1-indexed, meaning the first term corresponds to ***r*** = 1.

Input

The first line contains the number of test cases. And for each test case the first line gets the integer term $\boldsymbol{n}$, $(\mathbf{0 \le n \le 50})$, the power of binomial. The next line contains an integer $\boldsymbol{r}$, $(\mathbf{1 \le r \le n+1})$, the position of the term. The third line contain two integer numbers $\boldsymbol{x}$ and $\boldsymbol{y}$, $(\mathbf{1 \le x, y \le 100})$, the components of the binomial expression, separated by single space.

Output

A single line prints the value of the ***r-th*** term.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
2 5 3 2 3 4 2 1 2	240 8

Problem 6: The Quest of Mango Tree Network in Pakistan

Time limit: 2 seconds

In the heart of Pakistan, nestled within the lush expanses of Multan's Mango Fields, lies a complex network of interconnected mango trees, where nature's intricate web has remained unchanged for centuries. This forest is home to a series of pathways, hidden and ancient, known only to the most seasoned adventurers. Over time, the inhabitants of nearby villages have passed down tales of this ancient network, yet no one has fully mapped its extent. Some say it connects two distant mango trees at the farthest ends of the network, but no one has returned with definitive proof.

The Sufi elders of Sindh have left a riddle to guide your journey: "To find the paths of the Great Mango Network, begin at the first tree you see. Traverse through all the trees until you reach the farthest ones. Only then will you uncover the truth of the Great Mango Path and its companion." At the same time, it is important to note the density of the two most connected trees, as this offers insight into the characteristics of a healthy tree and its potential for expansion.

Remember, true power lies not just in finding a single path, but in understanding the complexity and interconnectedness of the entire network. Your journey is about uncovering secrets, testing hypotheses, and perhaps discovering new insights along the way.

Input

The first line will be the number of test cases. In the second line the number of mango trees should be entered. Followed by lines for each edge ***(Vertex_1   Vertex_2)***.

- **Number of Vertices (*V*):** $2 <= V <= 1000$
- **Number of Edges (*E*):** $E = V - 1$
- **Edge Constraints**:
  - Each edge connects two different vertices u and v where $0 <= u, v < V$.
  - There are no duplicate edges, and all edges are undirected.

Output

The output in one line shows the three integers

- Depth of the forest
- First and second highest connected trees in the forest.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
1 5 0 1 0 2 0 3 0 4	2 4 1

Problem 7: Survive the Zombie

Time limit: 2 seconds

Mustafa is gathering an army to fight a demon in a videogame. The army consists of two parts: the heroes and the defensive artifacts. Each hero has his health, and each defensive artifact has its durability to consider in this game. Before the battle begins, Mustafa distributes artifacts to the heroes so that each hero receives at most one artifact. To simulate the game, the battle consists of rounds that proceed as follows:

- First, the dragon deals damage equal to $\frac{1}{a+b}$ (a real number with precision of two floating points) to each hero, where a is the number of heroes alive and b is the number of active artifacts.
- After that, all heroes with health 0 or less die. An artifact with durability is deactivated when one of the following occurs:
  - The hero holding the artifact either dies or receives x total damage (from the start of the battle).
  - If an artifact is not held by any hero, it is inactive from the beginning of new round.
  - The battle ends when there are no heroes left alive.

Initially, the army is empty. There are q queries: add a hero with health x or an artifact with durability y to the army. After each query, determine the maximum number of rounds that Mustafa can survive if he distributes the artifacts optimally.

Input

The first line contains one integer q where $1 \le q \le 3.10^5$ — the number of queries. In the following q lines, there are two integers t_i and v_i ($t_i \in \{1, 2\}; 1 \le vi \le 10^9)$ — the type of the query and the value of the query parameter. If the type is 1, a hero with health v_i is added. If the type is 2, an artifact with durability v_i is added.

Output

Print q integers. After each entry of either artifact or hero, output the maximum number of rounds that Mustafa can survive if he distributes the artifacts optimally.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
3 2 5 1 4 1 10	0 8 19

Problem 8: Monkey Election

Time limit: 2 seconds

The Monkey House has n enclosures, numbered with integers from 1 to n. Enclosure i is home to b_i monkeys and g_i gorillas. Some enclosures are connected by walkways. Two enclosures are said to be adjacent if there is a walkway between them. A simple path from enclosure x to y is a sequence $s_0,\ldots,s_p$ of distinct enclosures, where p is a non-negative integer, $s_0=x$, $s_p=y$, and each pair s_{i-1} and s_i is adjacent for all $i=1,\ldots,p$. Because the walkways are set up in such a way that for any two enclosures x and y, there is exactly one simple path from x to y, it is guaranteed that the network of enclosures forms a tree.

A section is a non-empty set S of enclosures such that for any two x and y in S, there is a simple path from x to y that passes only through the enclosures in S. A collection of sections P is called a partition if each of the n enclosures is part of exactly one section in P. This means no two sections in P share any enclosures, and together they account for all n enclosures.

The Monkey House holds its annual **Mister Jungle** competition. This year's contestants are **Grumpy Gorilla** and **Playful Monkey**. The winner of the competition is determined by a voting system, which we will now explain. Suppose P is a partition of the enclosures into m sections $S_1,\ldots, S_m$. Each village holds its own election. The monkeys vote for their favorite contestant (Playful Monkey), and the gorillas vote for theirs (Grumpy Gorilla). No one abstains from voting.

Mayor Gorang and his assistant, Zangorang, want **Grumpy Gorilla** to win. They can choose how to partition the Monkey House into exactly m sections. If they make the partition optimally, we want to know the maximum number of villages in which **Grumpy Gorilla** can win.

Input

The first line of input contains a single integer t $(1 \le t \le 100)$ representing the number of test cases. Each test case is described as follows:

- The first line contains two space-separated integers n and m $(1 \le m \le n \le 3000)$.
- The second line contains n space-separated integers $b_1, b_2,\ldots, b_n$ $(0 \le b_i \le 109)$, where bi represents the number of monkeys in enclosure i.
- The third line contains n space-separated integers $g_1, g_2,\ldots, g_n$ $(0 \le g_i \le 109)$, where g_i represents the number of gorillas in enclosure i.
- The next $n-1$ lines describe the pairs of adjacent enclosures. Each of these lines contains two space-separated integers x_i and y_i $(1 \le x_i, y_i \le n, x_i \neq y_i)$, indicating that enclosures x_i and y_i are adjacent.

It is guaranteed that these pairs form a tree. It is also guaranteed that the total number of enclosures across all test cases does not exceed 105.

Output

For each test case, output a single line containing a single integer: the maximum number of villages in which Grumpy Gorilla wins, among all possible ways to partition the Monkey House into m villages.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

<table>
<tr><th>Sample input</th><th>Sample Output</th></tr>
<tr><td>2
4 3
10 160 70 50</td><td>2
0</td></tr>
</table>

70 111 111 0 1 2 2 3 3 4 2 1 143 420 214 349 2 1	

Problem 9: Factory Assembly Line Optimization

Time limit: 3 seconds

In the bustling city of Optimizia, factories operate interconnected assembly lines where each workstation processes parts. Transferring parts between two workstations consumes energy, and the factory's management wants to minimize the total energy consumption required to produce high-tech gadgets.

The factory has N workstations, numbered from 1 to N. Each workstation has specific input requirements and output production:

1. Workstation i requires $p[i-1]$ parts as input.
2. Workstation i produces $p[i]$ parts as output.

Energy consumption depends on the transfer between groups of workstations. Specifically:

- If workstation ii processes a batch of x parts and passes it to a group ending at workstation j, the **energy consumed** for this transfer is proportional to:

$$Energy = p[i-1] \times p[i] \times p[j]$$

where:

- $p[i-1]$: Input parts to workstation i,
- $p[i]$: Output parts from workstation i,
- $p[j]$: Input parts required by workstation j.

Your task is to determine the minimum total energy consumption needed to transfer all parts across N workstations by optimizing the grouping and processing sequence.

Input

The first line contains the number of test cases, and each test case consists of:

1. An integer N $(2 \le N \le 100)$, representing the number of workstations in the factory.
2. An array of $N+1$ integers $p[0], p[1], \ldots, p[N]$, where $(1 \le p[i] \le 1000)$:
   - $p[i-1]$ is the number of parts required at the input of workstation i,
   - $p[i]$ is the number of parts produced at the output of workstation i.

Output

For each test case, output a single integer: representing the minimum total energy required to transfer all parts across the workstations.

Sample input & output

The following is an example of a sample input and corresponding correct outputs.

Sample input	Sample Output
2 4 10 20 30 40 50 3 5 10 3 12	38000 330