2015 Taiwan National Collegiate Programming Contest

Final Round: Oct. 17. 2015

- Problems: There are 11 problems (27 pages in all) in this packet.
- Program Input: Input to the program are through standard input. Program input may contain one or more test cases. Test cases may be separated by any delimiter as specified in the problem statements.
- Program Output: All output should be directed to the standard output (screen output).
- Time Limit: Judges will run each submitted program with certain time limit (given in the table below).

Table 1: Problem Information

	Problem Name	Time Limit
Problem A	Determine a Parabola	3 sec.
Problem B	Detecting Counterfeit	1 sec.
Problem C	2-Geodetic Number	1 sec.
Problem D	Fibonacci Game	3 sec.
Problem E	Route Counting	3 sec.
Problem F	Recover Secret Key	5 sec.
Problem G	A Street Cleaner	3 sec.
Problem H	Critical for Pairing	3 sec.
Problem I	Voting Strategy	3 sec.
Problem J	Matchmaking	1 sec.
Problem K	Pick-6	3 sec.

Problem A
Determine a Parabola
Time Limit: *3 Seconds*

A plane curve called parabola can be expressed as

$$y = ax^2 + bx + c, \quad \textit{where} \;\; a, b, c \;\; \textit{are given parameters}$$

followed by a possible rotation.

It is obvious that given any three distinct points in a plane, (x_1, y_1), (x_2, y_2), (x_3, y_3), a parabola passing these 3 points is uniquely determined. For example, given three plane points $(0, 2)$, $(1, 1)$, $(3, 5)$, the following parabola is uniquely determined.

$$y = x^2 - 2x + 2$$

Given 3 distinct plane points (x_1, y_1), (x_2, y_2), (x_3, y_3). You are asked to write a program to find the parameters a, b, c of the unique parabola

$$y = ax^2 + bx + c$$

Input Format

The first line of the input contains one integer, $K \leq 5$, indicating the number of test cases to come. Each of the next K lines consists of six integers x_1, y_1, x_2, y_2, x_3, y_3, separated by space(s), of the three given plane points as described above that a parabola passes through. The values of x_1, y_1, x_2, y_2, x_3, y_3 and the parameters a. b. c are all integers limited to [-64, 64].

Output Format

The output consists of K lines. Each line consists of three integers in the order of a, b, c and are separated by space(s).

Sample Input

```
4
-1 0 0 -1 1 0
-1 -9 0 -4 2 0
-1 0 0 32 1 0
6 4 8 0 9 1
```

Output for the Sample Input

```
1 0 -1
-1 4 -4
-32 0 32
1 -16 64
```

Problem B
Detecting Counterfeit
Time Limit: *1 Second*

A few oracles were discovered on an archeology site a decade ago. On each oracle, there is an mysterious equation which puzzled the experts for five years. Five years ago, it was determined that the equation is designed to detect counterfeits. The equations can have 3 operators - addition("+"), subtraction("-") and multiplication("*") and the operands are natural numbers. For example, $2 + 3 * 4 = 20$ is one such equation. Since there are only a few oracles discovered and the price sky-rocketed, there are a lot faked oracles. There is an equation on each faked oracle as well, otherwise people can easily tell. Last year, a remarkable discovery determined that the people made those oracles used modular arithmetic and the equation holds by adding appropriate parenthesis to the equation. Given a set of oracles, you are to write a program to decide if there is a way to add parenthesis to make the equation true. Luckily, the discovery also found that the rule of applying parenthesis and its effect are the same as basic arithmetic we learnt in school.

Technical Specification

1. The number of operators in an equation is at most 15.
2. The operands in the equations are natural number between 1 and 2000 so is the modulus.

Input Format

The first line of the input contains an integer, n, which is the number of oracles. The second line of the input contains an integer, p, which is the modulus. The next n lines contain the equations, one equation for each line with the format: $a_0, op_0, a_1, op_1, ..., a_{k-1}, op_{k-1}, a_k = m$. Note that "," does not appear in the input line.

Output Format

For each test case, output 0 if it is not possible to add parenthesis to make the equation true otherwise output the parenthesis added equation. There might be many solutions, you only have to output one.

Sample Input

```
3
37
1+2*3=5
5+6*6=29
2*3+4*5=33
```

Output for the Sample Input

```
0
(5+6)*6=29
2*(3+4)*5=33
```

Problem C
The 2-Geodetic Number
Time Limit: *1 Second*

Let $G = (V, E)$ be a graph. A set $S \subseteq V$ is 2-*geodetic* if, for every vertex $u \in V \setminus S$, there exist two nonadjacent vertices $x, y \in S \cap N(u)$. The 2-geodetic number $g_2(G)$ of G is defined as the minimal cardinality of a 2-geodetic set. The 2-*geodetic problem* is to determine $g_2(G)$ for a graph G. For example, see Figure 1. In Figure 1, the set $\{a, b. f, g, h\}$ is a 2-geodetic set and $g_2(T) = 5$. You are asked to design an algorithm for solving the 2-geodetic problem on trees.

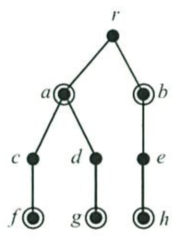

Figure 1: A tree of 9 vertices.

Technical Specification

- The number of vertices in a tree is at most 100.

Input File Format
The first line of the input file contains an integer denoting the number of test cases. There are at most 10 test cases. In each test case, the first line contains an integer n which is the number of vertices in a tree. The second line contains n integers in which the ith integer is the parent of vertex i for $1 \leqslant i \leqslant n$. Note that the parent of each vertex i is smaller than i except that the parent of the root is the root itself. Note also that there is a blank between any two integers.

Output Format
For each test case, output $g_2(T)$ in a line.

Sample Input

```
2
5
1 1 1 1 2
17
1 1 1 2 2 3 3 3 7 7 7 7 8 8 9 9 9
```

Output for the Sample Input

```
4
12
```

Problem D
Fibonacci Game
Time Limit: *3 Seconds*

A math teacher plays the following game with his students to ge likes to play the following game with his students. He first writes n consecutive numbers on the blackboard in clockwise direction from 1 to n. He asks students to cross out numbers one by one according to the numbers in the Fibonnaci sequence F_i and to determine the last remaining number when n-1 numbers are crossed out.

1) Start at number 1 with counting direction set to clockwise. 2) At the i^{th} turn, cross out the ${F_{i-1}}^{th}$ remaining number from the current position in the current counting direction, where F_{i-1} is the i^{th} number in the Fibonnaci sequence. 3) If the number crossed out is a prime number, then the counting direction reverses direction, from clockwise to counter-clockwise and vise versa. For example, if the math teacher writes down 5 numbers as shown in Figure 1. Then the number crossed out are shown as follows.

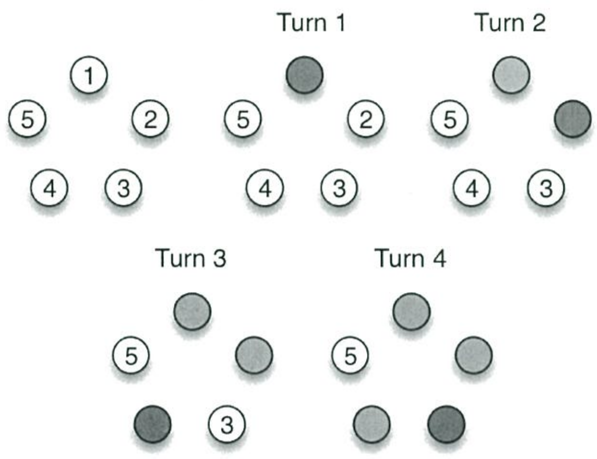

Figure 1: Game with 5 numbers

1. 5 numbers are written down 1 to 5 in clockwise order.
2. Turn 1: $F_0 = 1$, count 1 number in clockwise direction, cross out 1.
3. Turn 2: $F_1 = 1$, count 1 number in clockwise direction, cross out 2. Since 2 is a prime number, the counting direction is changed to counterclockwise.
4. Turn 3: $F_2 = 2$, count 2 numbers in counterclockwise direction, cross out 4.
5. Turn 4: $F_3 = 3$, count 3 numbers in counterclockwise direction, cross out 3. Since 3 is a prime number, the counting direction is changed to clockwise.
6. At this stage, only one number remains, which is the number 5.

Please write a program to help math teacher determine the last remaining number.

Technical Specification

1. A Fibonnaci sequence $F = (F_0, F_1, F_2, F_3, \ldots)$, where $F_0 = 1, F_1 = 1$, and $F_i = F_{i-1} + F_{i-2}$ for all $i > 1$.
2. The total number of numbers written down by the math teacher is n, $2 \leq n \leq 35$.

Input Format

The first line of input contains an integer indicating the number of test cases to follow. For each test case, there is a single integer on a line, indicating the total number of numbers, n, written down by the teacher.

Output Format

For each test case, print the last remaining number not crossed out, after all the others are crossed out.

Sample Input

```
2
5
6
```

Output for the Sample Input

```
5
3
```

Problem E
Route Counting
Time Limit: *3 Seconds*

Napaj is a great country that are visited by many international tourists. To help travelers, Napaj designs a very good public transportation system. There are N tourist spots in Napaj. Each spot is assigned with a unique integer ID from 1 to N. In each spot, there are direct one-way non-stop tour buses going to some other spots every minute. For obvious reasons, there is no bus from a spot i to the same spot i. There is also at most one bus going from spot i to spot j. Due to traffic control, each bus only goes from one spot to another spot without passing through any other spot. When the bus arrives at the destination, it stops serving. For security reasons, once a tourist boards a bus, no disembarkation is allowed until the bus destination is reached. At each spot, there are at most D buses going to different spots.

Mr. Tomato a mathematician arrives at a spot S and wants to go to another spot T. We know that $S \neq T$. In order to kill time in between the time to wait for buses, he wants to count the number of different routes using buses from S to T. A *route* from S to T is a sequence of spots $V_1, \ldots, V_k$ such that (1) $S = V_1$, $T = V_k$, (2) there is a direct bus from V_i to $V_i + 1$, $1 \leq i < k$, and (3) $V_i \neq V_j$ for all i and j. Two routes $U_1, \ldots, U_{k_1}$ and $V_1, \ldots, V_{k_2}$ are *different* if either (1) $k_1 \neq k_2$, or (2) $k_1 = k_2$ and there exists some i, $1 \leq i \leq k_1$, such that $U_i \neq V_i$.

Given the bus network, please help Mr. Tomato find the total number of different routes to go from S to T.

Technical Specification

1. $0 < N \leq 12$.
2. $0 < S \leq N$.
3. $0 < T \leq N$.
4. $S \neq T$.
5. $0 \leq D < N$.

Input Format

The first line of the input contains an integer, which is the number test cases to follow. There are at most 10 test cases. In each test case, the first line has 4 integers N, D, S, T each separated by a space. Lines 2 to $N + 1$ contains the bus tables for the spots. On line $i + 1$, the first integer indicates the number of distinct buses originate from spot i. The remaining numbers on the line are the destinating spot number of the buses originate from spot i.

Output Format

For each test case, output the total number of different routes going from S to T.

Sample Input

```
2
5 2 1 3
2 2 4
1 3
1 5
1 3
0
7 2 2 7
2 2 3
2 4 5
2 4 5
2 6 7
1 7
0
0
```

Output for the Sample Input

```
2
2
```

Problem F
Recover Secret Key
Time Limit: *5 Seconds*

Encryption and decryption are important tools to protect sensitive data from unauthorized access. Encryption and decryption need secret keys. In some critical situation, the secret key should not be kept by only one person, they should be shared by many people.

Let k be the secret key to be shared. A good method to share the secret key by n people in a way that any subgroup of m people, $m \leq n$, can retrieve the key is to use a polynomial of degree $m-1$.

First each person p_i is assigned a randomly chosen integer x_i, $i = 1, 2, \ldots, n$. Then randomly select $m-1$ integers $a_1, a_2, \ldots, a_{m-1}$, and construct a polynomial $f(x)$ of degree $m-1$ by using the key and the randomly selected integers:

$$f(x) = k + a_1 x + a_2 x^2 + \cdots + a_{m-1} x^{m-1}.$$

Each person p_i can now be assigned a point on $f(x)$: (x_i, y_i), where $y_i = f(x_i)$. Note that this step must be done in a secure way. Only p_i knows its own y_i.

It is well-known that a polynomial of degree $m-1$ can be represented by m points on the polynomial. The polynomial can also be retrieved by using a set of m distinct points on the polynomial. Thus, the secret key can be computed by using any m points. Write a program to compute the secret key from some m points of a polynomial.

To avoid large integers and the loss of precision in floating point computation, all calculations should be done in a finite field $\mathbf{GF}(p)$, where p is a prime larger than the secret key k. All addition and multiplication operations should take the remainder after dividing by p. For example, let $p = 19$. In $\mathbf{GF}(19)$, $10 + 15 = 25 \equiv 6$ and $3 \times 7 = 21 \equiv 2$.

For subtraction $a - b$, you need to do it by adding the *additive inverse* of b, $a + b = a + (-b)$. For example, $6 - 10 \equiv 6 + (19 - 10) = 15$. Division a/b is done by multiplying the *multiplicative inverse* of b, $a/b = a \times b^{-1}$.

If $a \times b \equiv 1$, then the inverse of a is b. For example, $4 \times 5 = 20 \equiv 1 \pmod{19}$, therefore, the inverse of 4 is 5 in $\mathbf{GF}(19)$. There exist efficient algorithms to compute the multiplicative inverse, even if p is very large. In this problem, you can try all possible integers to find the multiplicative inverses, since the value of p is not too large.

Technical Specification

1. $p < 2^{20}$,
2. $n < 100$,
3. k and each a_i is no more then p.

Input Format

The first line of input contains a prime p. This prime will be used in all test cases. Each test case consists of 2 lines. The first line of a test case contains an integer m. The second line contains m pairs of integers x_i, y_i, $i = 1, 2, \ldots, m$. The last test case is followed by a line containing 0.

Output Format

For each test case, print out the secret key.

Sample Input

```
31
3
25 9 3 19 18 13
5
3 5 15 22 21 15 11 30 8 21
0
```

Output for the Sample Input

```
23
5
```

Problem G
A Street Cleaner
Time Limit: *3 seconds*

A street cleaner, Raymond, is in charge of keeping the streets clean at X city. There are several *supply stations* (*stations* for brevity) for providing supply materials to street cleaners, some of which are connected by streets such that any two stations can be reached each other by traversing at least one street. Note that each street is bidirectional. For convenience, we can view the supply stations and streets as a graph in which supply stations are represented by vertices and streets are represented by edges. A supply station is *odd station* if there are odd number of streets incident to it; otherwise, it is an *even station*. Each edge (u, v) is associated with a positive rational number representing the cleaning cost $c(u, v) = \frac{q}{p}$, where p and q are two positive integers. Given two odd stations S and T ($S \neq T$), Raymond would like to clean all the streets, starting from S and ending at T. Your task is to help Raymond seek an S-T walk that minimizes the total cost (the sum of the edges' costs in the walk) such that all the streets are cleaned at least once. We call such a walk as a minimum S-T walk, and call the cost of this walk the *minimum cost*. Note that a walk is a list $v_0 \to v_1 \to \cdots \to v_{k-1} \to v_k$ such that, for $1 \leq i \leq k$, vertices v_{i-1} and v_i are adjacent. Note that a walk may repeat vertices and edges. A walk is *open* if its endpoints are different.

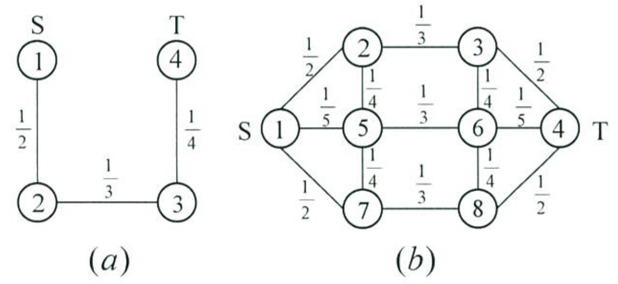

Figure 1: Two examples

For example, "$S = 1 \to 2 \to 3 \to 4 = T$" is only one way to find the minimum S-T walk in Figure 1(a). The sum of those edges' costs (the minimum cost) is 13/12. In Figure 1(b), there are many ways to find S-T walks. For example, "$S = 1 \to 2 \to 5 \to 7 \to 1 \to 5 \to 6 \to 4 \to 3 \to 2 \to 5 \to 7 \to 8 \to 6 \to 3 \to 6 \to 8 \to 4 = T$" and "$S = 1 \to 2 \to 3 \to 4 \to 8 \to 7 \to 1 \to 5 \to 2 \to 3 \to 6 \to 8 \to 7 \to 5 \to 6 \to 4 = T$" are two S-T walks such that all the streets are cleaned at least once. The former is not a minimum S-T walk (with total cost 27/5), but the latter one is a minimum S-T walk with the minimum cost 76/15.

Given a graph $G = (V, E)$ and two vertices S and T, we use $1, 2, \ldots, |V|$ to represent the nodes. Each edge (u, v) is associated with the cost $c(u, v)$ that is a positive rational number. Your task is to write a computer program to compute the cost $\frac{q}{p}$ of a minimum S-T walk that traverses all the edges at least once, where $\frac{q}{p}$ is an irreducible fraction (i.e., p and q have no common factor).

Technical Specification

1. $G = (V, E)$ is connected.

2. $2 \leq |V| \leq 100$.
3. $1 \leq |E| \leq \frac{|V|(|V|-1)}{2}$.
4. For each edge (u, v) with $c(u, v) = \frac{q}{p}$, $1 \leq p \leq 10$ and $1 \leq q \leq 100$.

Input Format

The first line of the input contains an integer indicating the number of test cases to follow. The input consists of a number of test cases. Each test case consists of a graph $G = (V, E)$, which has the following format: the first line contains four numbers, $n(= |V|)$, $m(= |E|)$, S, and T such that any two consecutive numbers separated by a single space. The next m lines contain the description of m edges and the corresponding costs such that one line contains two endpoints of an edge and the two positive integers p and q, where p and q are denominator and numerator of the cost of the corresponding edge. Each line is represented by four positive numbers separated by a single space; the first number representing one endpoint, the second representing the other endpoint, and the third representing the denominator of the cost, and the fourth representing the numerator of the cost.

Output Format

The output contains one line for each test case. Each line contains two positive integers p and q, where p and q are denominator and numerator of the cost of a minimum S-T walk that traverses all the edges at least once. Note that $\frac{q}{p}$ is an irreducible fraction.

Sample Input

```
2
4 3 1 4
1 2 2 1
2 3 3 1
3 4 4 1
8 13 1 4
1 2 2 1
1 5 5 1
1 7 2 1
2 5 4 1
2 3 3 1
5 6 3 1
5 7 4 1
7 8 3 1
3 6 4 1
3 4 2 1
6 4 5 1
6 8 4 1
8 4 2 1
```

Output for the Sample Input

```
12 13
15 76
```

Problem H
Critical for Pairing
Time Limit: *3 Seconds*

There are n people in a company, and each person has exactly one leader except the CEO has no leader. There are some jobs to do, and each job needs a team of exactly two persons to complete. Furthermore, one of the persons in a team must be the leader of the other. Therefore, we want to pair as many teams as possible.

In Fig. 1, the top leftmost tree shows an example of a company. Each vertex represents a person in the company, and there are six persons with labels from 0 to 5. Each edge connects a person and his/her leader, with the upper one being the leader. For example, person 2 is the leader of 4 and 5, and person 0 is the CEO.

In this six-person organization, there are serveral ways to organize two-person teams. However, at most two teams can be formed. A pairing achieves the maximized number of teams is a "maximum pairing". In fact, there are five different maximum pairings, as shown in Fig. 1.

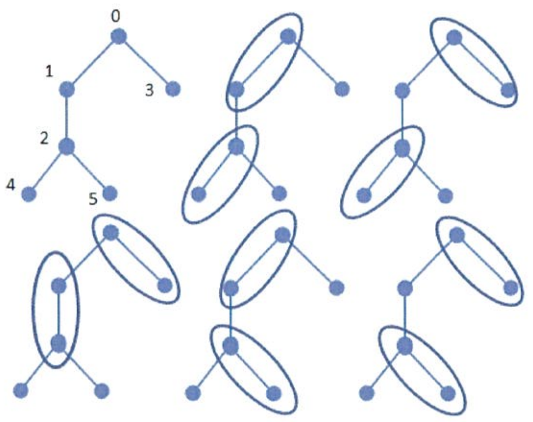

Figure 1: An example of the company and all the possible maximum pairings.

By examining all five maximum pairings, we find that persons 0 and 2 are more critical than the others, since all the maximum pairings need these two persons. On the other hand, person 1 is not so critical, i.e., even if he/she does not join any team, we can also organize two teams. Thus, we define that a person is critical if all the maximum pairings contain the person. In other words, if a critical person is removed, the maximized number of pairs decreases. In this task, you are asked to write a program for finding the critical persons.

Technical Specification

1. The number of persons n satisfies $2 \leq n \leq 100000$.

Input Format

The first line of the input contains an integer which indicates the number of test cases. The input of a test case consists of two lines. The first line contains the integer n, which is the number of persons in the company. Each person has a unique ID between 0 and $n-1$. For each person, his/her ID must be larger than the ID of his/her leader. The CEO has ID 0. The second line contains $n-1$ integers $t(1), t(2), , t(n-1)$ such that $t(i)$ is the leader of i for all i. Any two consecutive numbers in the same line are separated by a space.

Output Format

For each test case, output two lines. In the first line, output the maximum number of pairs and then the number of critical persons. In the second line, output the ID of the critical persons from large to small. If there are more than three critical persons, output the largest three. Any two consecutive numbers in the same line are separated by a space.

Sample Input

```
2
6
0 1 0 2 2
7
0 1 2 3 4 5
```

Output for the Sample Input

```
2 2
2 0
3 3
5 3 1
```

Problem I
Voting Strategy
Time Limit: *3 seconds*

Joe is working for a political consulting company. Joe has a client looking for help to win a national election. The situation is that there are three political parties A, B, C and with m equal-sized voting districts. In the i-th district there are a_i, b_i, c_i voters for A, B, C, respectively. Since all districts have equal size s, we have $a_i + b_i + c_i = s$, for $1 \leq i \leq m$. For each district, the party that receives the relative majority of votes wins the district. To simplify the rules, we assume that ties are always broken to the *disadvantage* of party A. Therefore, party A wins a district if and only if $a_i > \max\{b_i, c_i\}$ holds. In other words, if $a_i = b_i$ or $a_i = c_i$, then party A loses the i-th district.

Joe's client wants to know if it is possible that parties B and C can repartition their votes such that they reach the relative majority in at least k districts.

More specifically, do there exist non-negative integers b'_i, c'_i, $i = 1, \ldots, m$, with $\sum_{i=1}^{m} b'_i = \sum_{i=1}^{m} b_i$, and $\sum_{i=1}^{m} c'_i = \sum_{i=1}^{m} c_i$ and $b'_i + c'_i = b_i + c_i$ for $1 \leq i \leq m$, such that there exists an index set $I \subseteq \{1, \ldots, m\}$ with $|I| = k$ and $a_i \leq \max\{b'_i, c'_i\}$ for all $i \in I$?

For example, suppose there are 2 districts, each district has 8 voters and parties B and C want to win 2 district, i.e, $m = 2, s = 8$ and $k = 2$. Assume in both districts there are 4, 3, 1 voters for parties A, B and C, respectively. Then there is no way for parties B and C to trade votes and win the election in both districts. However, if in the second district there are 4, 1, 3 voters for A, B, C, respectively. Then it is possible for B and C to trade 1 vote in both districts, such that B can win the first district and C wins the second district.

Technical Specification

1. n: the number of test case, $n \leq 30$.
2. m: the number of districts, $m \leq 100$.
3. s: the total number of voters in each district, $s \leq 30000$.
4. k: the number of districts to win, $k \leq m$.
5. a_i, b_i, c_i: the number of voters of A, B, C, respectively, $a_i, b_i, c_i \leq 10000$, $i = 1, \ldots, m$.

Input Format

The first line of the input contains an integer n indicating the number test cases, $n \leq 30$. Each test case consist of $m + 1$ lines. For each test case, the first line has three positive integers m s k (separated by space(s)) indicating the number of voting districts, the total number of voters in a district and the number of districts to win, and then in the following m lines, the i-th line has three non-negative integers (separated by space(s)) a_i b_i c_i, indicating the number of voters of partites A, B, C, respectively. For convenience, we insert a blank line between adjacent cases.

Output Format

For each test case, output YES, if it is possible to repartition the votes of B and C such that they can win the election in at least k district; otherwise output NO.

Sample Input

```
3
3 6 2
1 3 2
2 2 2
4 0 2

2 8 2
4 3 1
4 3 1

2 6 2
3 2 1
3 1 2
```

Output for the Sample Input

```
YES
NO
YES
```

Problem J
Matchmaking
Time Limit: *1 Second*

National Computer Programming Company (NCPC) is a well-established corporation that attracts many young talents every year. However, stress and anxiety from the workplace and metropolitan lifestyle restrict the motivation and opportunities for meeting new people, building new relationships, and ultimately finding a life partner, which has been an important factor behind the company's high attrition rate. In search for a stable workforce, NCPC plans to hold a large matchmaking event, and to enhance the likelihood of further engagement, the host asks each participant to select and rank one or more names that they would be interested in dating from the full list of participants beforehand. To ensure the efficiency of the event, the host banned the practice of ranking multiple names in tie.

The level of a participant's satisfaction is dependent on the person he/she is paired with under two different pairing methods. If participant A is more satisfied with pairing method M than M', this indicates that the person A is paired with under M ranks higher in his/her personal ranking than the person A is paired with under M', or that A finds a paired person under M but not under M'. We say a pairing method M is a *popular matching* if there does not exist a pairing method M' such that the number of participants satisfied M' is more than the number of participants satisfied M. The goal is to develop a popular matching system that paired the maximum number of participants.

For example, let m_1, m_2, m_3 be three male participants and f_1, f_2, f_3 be three female participants. Their wish list ranking from high to low are as follows:

m_1:	1 2 3	f_1:	1 2 3
m_2:	1 2	f_2:	1 2
m_3:	1	f_3:	1

Consider the following three matchings where (i, j) indicates pairing m_i with f_j.

M_1: (1,1), (2,2)
M_2: (1,2), (2,1)
M_3: (1,3), (2,2), (3,1)

Comparing M_1 with M_2, we know that m_1 and f_1 prefer M_1 than M_2, m_2 and f_2 prefer M_2 than M_1, and m_3 and f_3 has no preference between them.

Comparing M_1 with M_3, we have m_1 and f_1 prefer M_1 than M_3, m_3 and f_3 prefer M_3 than M_1, and m_2 and f_2 has no preference between them.

Comparing M_2 with M_3, we find that m_1, f_1, m_2 and f_2 prefer M_2 than M_3, and m_3 and f_3 prefer M_3 than M_2.

Since there are more participants satisfied M_2 than M_3, M_3 is not a popular matching while M_1 and M_2 are both popular.

Eligible participants in the matchmaking event must be single employees of the company, thus the total number of possible participants is within 500 for male and female participants respectively. Since current legal practices require marriage as the joining of opposite sexes, each participant's ranking must also be based on names from the opposite sex. Because the number of people interested in the event and the number of names in participants' rankings are still unknown, the host does not guarantee successful pairing for all participants, however successfully paired participants would be paired with only one other participant.

Technical Specification

1. The number of male participants m satisfies $1 \leq m \leq 500$, each participant is assigned a number i such that $1 \leq i \leq m$.
2. The number of female participants f satisfies $1 \leq f \leq 500$, each participant is assigned a number j such that $1 \leq j \leq f$.
3. Every one only rank the person in opposite sex that he/she wishes to date with, and there is no tie in the rank.

Input Format

The first line of the input contains an integer n ($n \leq 10$) indicates the number of test cases that follows. The first line of each test case consists two integers m and f which are the number of male and female participants respectively. Then it immediately follows by $m + f$ lines, each line contains several integers separated by a space. Line i, for $1 \leq i \leq m$, is the male i's preference list of females he is interested to dating in decreasing order. Similarly, line j for $m + 1 \leq j \leq m + f$, is the female j's preference list of males she is interested to dating in decreasing order. Then it immediately follows by next test case.

Output Format

For each test case, output the maximum cardinality of the popular matchings in one line.

Sample Input

```
3
2 2
1 2
1
1 2
1
3 3
1 2 3
1 2
1
1 2 3
1 2
1
6 6
1
1 5 2
4 2 3
4
5 4
5 6
2 1
2 3
2 3
```

```
5 3 4
2 6 5
6
```

Output for the Sample Input

```
2
2
6
```

Problem K
Pick-6
Time Limit: *3 Seconds*

Pick-6 is a game played by two players. Each player first pick six digits from the set {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. The digits chosen can be in any order and can also repeat. For example, a player can pick 7, 9, 0, 5, 9, 3 or 1, 1, 1, 1, 1, 1 if so desired.

The two players then take turn to try to score points. Player A first arrange a number using the digit(s) he has picked (at least a one-digit number and at most a six-digit number). If player B cannot match that number by using the digit(s) he picked, then player A scores a point. Player B then arrange a number for player A to match. The game continues until neither player can generate any more numbers.

Please write a program to determine the score of each player.

Input Format
The first line of the input contains an integer n ($n \leq 10$) indicating the number of test cases that follows. The first line of each test case consists of the six digits picked by player A. The second line of each test case consists of six digits picked by player B. Digits are separated by space(s).

Output Format
For each test case, output two integers separated by a space on a single line indicating the score of player A and B, respectively.

Sample Input

```
2
1 1 1 9 9 9
8 3 3 8 8 3
1 2 3 4 5 6
0 0 0 0 0 1
```

Output for the Sample Input

```
68 68
1955 5
```