2012 ACM Regional Programming Contest
Asia Kaohsiung
2012/11/24

This problem set consists of 10 problems in 22 pages. Each submitted program will be judged as either *accepted* or *rejected* by the Judges. An accepted program must be able to solve all the test cases of its corresponding problem correctly. The *time* consumed for a solved problem is measured from the beginning of the contest to the submission of the accepted program. For each failed submission, however, a penalty of 20 minutes will be added to the time for this problem. Yet, no time penalty will be given for the problems left unsolved at the end of the contest.

Number of problems solved determines team placement in the contest, with ties broken by total time taken. The *total time* is the sum of the time consumed for each solved problem.

Sample input and output data are given in each problem. The sample input may not be, and usually they are not, the same as the test cases used by the Judges. You are allowed to create sample input data of your own to test your program.

It is not necessary to include comments in your programs. Any algorithm can be applied to solve the problems. However, a program whose execution time exceeds the time limit specified in each problem will be considered as *rejected*.

The input file for problem x is `p`x`.in`. You may assume all input data to your program are correct, which means error checking for the input data is not required. The output file must be the standard output which can be redirected to other files, unless it is specified in the problem.

Each input file may contain more than one test case. The description of the input format is based on records which usually contains a list of data. These data may span more than one line, unless otherwise stated. Maximum number of characters in a line is 255. A special record, usually one or more 0s, indicates the end of an input data. This is an indication of the end of input data. It is not a part of the test data, and anything after this special record should also be ignored.

=0

Problem A
Signed Binary Representation of Integers
Input file: `pa.in`   Time limit: 2 seconds

Computing $a^x \bmod n$ for large integers x and n is the most time-consuming process in most public-key cryptography systems. An algorithm to compute $a^x \bmod n$ can be described in a C-like pseudo-code as follows.

Input: positive integers a, x and n,
Output: $y = a^x \bmod n$
Method:

```
convert $x$ into binary $(x_{k-1}x_{k-2}\ldots x_0)_2$;
$y = 1$;
for $(i = k-1;\ i \geq 0;\ i = i-1)$ {
  $y = y^2 \bmod n$;
  if $(x_i == 1)$ $y = y \times a \bmod n$;
}
print $y$;
```

In the above algorithm, first x is converted into k-bit binary. Then the algorithm performs k iterations. In each iteration, a square ($y^2 \bmod n$) is computed. In the i-th iteration, if $x_i = 1$, the algorithm also computes $y = y \times a \bmod n$. Therefore, the computing time depends on the number of 1's in the binary representation of x.

Let a^{-1} be the inverse of a in the group $\mathbf{Z}_n^*$. That is $a \times a^{-1} \equiv 1 \bmod n$. For example, assume that $n = 13$, then the inverse of 2 is 7, $2 \times 7 = 14 \equiv 1 \pmod{13}$. In this problem, you do not need to know how to compute the inverses.

Assume that a^{-1} is known, and x is represented by $-1, 0, 1$, instead of $0, 1$, we can modify the above algorithm as follows.

Input: positive integers a, x and n,
Output: $y = a^x \bmod n$
Method:

```
convert $x$ into signed binary $(x_{k-1}x_{k-2}\ldots x_0)_{-1,0,1}$;
$y = 1$;
for $(i = k-1;\ i \geq 0;\ i = i-1)$ {
  $y = y^2 \bmod n$;
  if $(x_i == 1)$ $y = y \times a \bmod n$;
  if $(x_i == -1)$ $y = y \times a^{-1} \bmod n$;
}
print $y$;
```

In the above algorithm, we need to represent x by using $-1, 0, 1$. This is called *signed binary* representation of x. For convenience, we shall use $\bar{1}$ to denote -1. For example: $15 = (1111)_2 = (1000\bar{1})_{-1,0,1}$.

You can see that it may be more efficient to use signed binary representation in computing $a^x \bmod n$ when the inverse of a ($a^{-1} \bmod n$) is known. For $x = 15$, using binary needs 4 multiplications, while using signed binary needs only 2.

In this problem, you are going to write a program to convert a large integer into signed binary. Signed binary representation of an integer may not be unique. We need a representation with minimum number of non-zero bits to make the computation of $a^x \bmod n$ fast.

A hint of converting x into a signed binary with minimum number of non-zero bits: Make sure that no adjacent two bits are both non-zero.

Input Format

The input to this problem is a file containing a sequence of large integers. Each integer x is no more than 120 decimal digits, and it is written in a line. There will be no spaces in front of x, and no spaces after x. The last line of the input file contains a single 0. You do not need to process this line.

Output Format

The outputs for each test case consists of two parts written in one line. The first part is an integer b, which is the number of non-zero bits of the binary representation of x. The second part is an integer s, which is the number of non-zero bits in the signed binary representation of x. Print exactly one space between these two parts.

Sample Input

```
15
2514593
113113561845
0
```

Sample Output for the Sample Input

```
4 2
11 9
23 14
```

Problem B
The Shortcut Eight-Puzzle
Input file: `pb.in` **Time limit: 2 seconds**

The shortcut eight-puzzle is a sliding puzzle that consists of a frame of 3×3 spaces where 8 numbered square tiles are placed in a random order with one tile missing. In each sliding move, a numbered square tile can be moved either UP, DOWN, LEFT, RIGHT, or UP-RIGHT to the empty space. (Note that a numbered square tile **cannot** be moved DOWN-LEFT to the empty space) The original object of the puzzle is to change the positions of the tiles from an initial state to a fixed goal state by making sliding moves that uses the empty space. But now, given any initial state and any goal state, your program is requested to output the minimal number of sliding moves that can change the positions of the tiles from the initial state to the goal state. If the goal state is unreachable from the initial state by making sliding moves that uses the empty space, just output -1.

For example, given the initial state in Figure 1 and the goal state in Figure 2, the output should be 3.

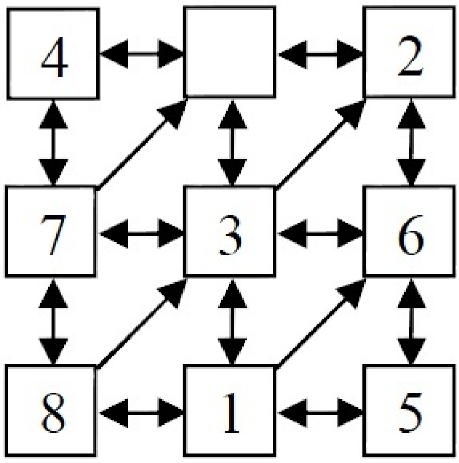


Figure 1: An initial state

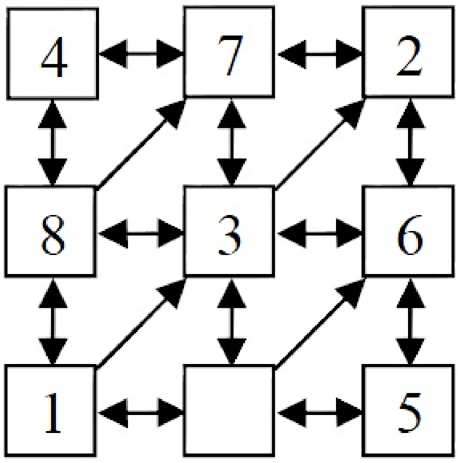


Figure 2: A goal state

Input Format

The number of test cases is given in the first line as an integer not greater than 5. For each test case, there are 2 more lines. The initial state is given in the first line in a row major order where 9 numbers representing the tiles and the empty space (denoted by 0) are separated by one space. The goal state is given in the second line in a similar way.

Output Format

For each test case, if the goal state is reachable from the initial state, please output the minimal number of moves necessary to reach the goal state in a separated line. If the goal state is unreachable from the initial state, please output -1.

Sample Input

```
2
4 0 2 7 3 6 8 1 5
4 7 2 8 3 6 1 0 5
2 6 4 1 3 7 0 5 8
8 1 5 7 3 6 4 0 2
```

Sample Output

```
3
27
```

Problem C
Star Travel
Input file: `pc.in` **Time limit: 1 seconds**

In year 3001, star travel is realized by a device called StarGate. With the StarGate, travelers can hop from a planet to a planet via hyperspace. However, due to some physical constraints on hyperspace, it is not always possible to establish StarGates between every two planets. In addition, the transporting is directional; that is, it is possible to transport from one StarGate to another but not vice versa. So, traveling to a planet is often done by hopping a series of StarGates geometrically.

However, in that era, human colonies are divided into several federations. A planet only belongs to a federation but a traveler can own the citizenship from more than one federation. A planet can operate more than one StarGate to other planets. A traveler can transport between two StarGates free, i.e, without tickets, if he owns the citizenship from the federations that control the two StarGates. Now, a traveler wants to travel from one planet to another. Please write a program to book a sequence of tickets for him, excluding the free tickets. Assume tickets are of the same price. Please plan a tour that has the minimum number of tickets.

Input Format

The test data begins with an integer N, which is the number of test cases. Each test case begins with an integer P, which is the number of planets. Following P is the P lines of planet data. The planet data begins with the ID of a planet. The planet ID is a non-negative integer ranges from 0 to 5000. The second item of planet data is the federation ID. Federation ID is a capital letter ranges from 'A' to 'Z'. Next is a number of outgoing StarGates on that planet, which is a non-negative integer S ranges from 0 to 100. When S is zero, no outgoing StarGates are deployed on that planet. Each outgoing StarGate is described by a ticket name Tx and destination planet ID that can be reached by this StarGate, where x is a non-negative integer from 0 to 99999999, which is not necessarily monotonically increasing in the test data. The ticket names never duplicate.

Following the planet data is the traveler data. The data begins with an integer T, which is the number of tours to be planned for the planets. Following is T lines of tour and traveler data. Each data contains the source planet ID and destination planet ID that he wants to travel. Following the two IDs is a string that contains the citizenship of federations of the traveler. For example, if a traveler owns the citizenship from federation A, B, and C. The string will be "ABC". Every traveler owns at least a citizenship from a federation.

Output Format

For each test case, please output the sequence of ticket names for each tour, excluding the free tickets. Each ticket name is separated by a space. If there are more than one solution, please only output the sequence that begins with the smaller ticket names. For example, suppose two sequences (T1 T3 T4) and (T6 T3 T8) are both valid answers. You only need to output (T1 T3 T4) because $T1 < T6$. If ticket names are equal, the rule applies consecutively until a difference is made.

Sample Input

```
2
5
0 A 2 T0 1 T2 2
1 A 1 T1 2
2 A 1 T3 3
3 B 1 T4 4
4 B 1 T5 1
1
0 4 A
6
0 A 2 T0 1 T2 2
1 A 1 T1 2
2 A 2 T3 3 T6 5
3 B 1 T4 4
4 B 1 T5 1
5 A 1 T7 4
2
0 4 A
5 3 A
```

Sample Output

```
T3 T4
T7
T7 T5 T3
```

Problem D
Species Evolution
Input file: `pd.in`   Time limit: 5 seconds

If a group of genes remain physically close to each other in multiple genomes, often called *gene clusters*, then the genes may be either historically or functionally related. Thus, the study of gene clusters is important for the understanding of evolution, since genomes evolved from a common ancestor tend to share the same varieties of gene clusters.

There are many gene cluster models. Dr. Lee uses the following approach to evaluate the quality of a given model. First, he finds a sequence of n species $(S_1, S_2, ..., S_n)$, where each S_i, $2 \leq i \leq n$, is evolved from S_{i-1}. Let k be a positive integer. Define $[0, k]$ to be an integer set ranging from 0 to k. Based upon the given gene cluster model, Dr. Lee computes a sequence of numbers $A = (a_1, a_2, ..., a_n)$, where each a_i, $1 \leq i \leq n$, is an integer in $[0, k]$ indicating the score of similarity between S_i and S_n. Of course, $a_n = k$. Dr. Lee believes that a perfect gene cluster model should satisfy the following two properties:

P1. The sequence A is digressing (i.e., $a_1 \leq a_2 \leq ... \leq a_n$). This property indicates that S_n is more similar to S_i than to S_{i-1}, $2 \leq i \leq n$.

P2. For $2 \leq i \leq n$, the difference between a_{i-1} and a_i is bounded by some threshold δ (i.e., $a_i - a_{i-1} \leq \delta$). This property indicates that S_i should not deviate far from S_{i-1}, since S_i is evolved from S_{i-1}.

In practice, the sequence A may not satisfy properties P1 and P2. Let R^n be the set of all possible sequences of n integers that satisfy properties P1 and P2. Dr. Lee defines the *distance* between A and a sequence $X = (x_1, x_2, ..., x_n) \in R^n$ to be $d(A, X) = \sum_{1 \leq i \leq n}(x_i - a_i)^2$. (You may consider $d(A, X)$ as the total cost of modifying each a_i to x_i, $1 \leq i \leq n$.) In addition, he defines the *inaccuracy* of the given gene cluster model to be $\varepsilon(A) = \min_{X \in R^n}\{d(A, X)\}$. In other words, he defines the inaccuracy to be the minimum cost of modifying A into a sequence (of n integers) that satisfies properties P1 and P2.

Consider an example of $n = 4$, $k = 10000$, $\delta = 3000$, and $A = (5000, 2000, 6000, 10000)$. Let $B = (5000, 7000, 10000, 12000)$, $C = (3500, 3500, 6500, 9500)$, $D = (5000, 4000, 6000, 8000)$, and $E = (5000, 5500, 6000, 10000)$. Both B and C are in the set R^4. However, D and E are not, since they do not satisfy P1 and P2, respectively. We have $d(A, B) = (5000 - 5000)^2 + (7000 - 2000)^2 + (10000 - 6000)^2 + (12000 - 10000)^2 = 45000000$ and $d(A, C) = (3500 - 5000)^2 + (3500 - 2000)^2 + (6500 - 6000)^2 + (9500 - 10000)^2 = 5000000$. In this example, C has the minimum distance from A among all sequences in R^4. Thus, $\varepsilon(A) = 5000000$.

Please write a program to compute $\varepsilon(A)$.

Input Format

There are at most 10 test cases. Each case consists of two lines. The first line contains the integer n ($1 \leq n \leq 100$), the integer $k(10000 \leq k \leq 50000)$, and the integer $\delta(0 \leq \delta \leq 5000)$. The second line contains the sequence $A = (a_1, a_2, ..., a_n)$, where $a_i \in [0, k]$ for $1 \leq i \leq n - 1$ and $a_n = k$. The last test case will be followed by a line containing a single zero.

Output Format

For each test case, output one line for $\varepsilon(A)$.

Sample Input

```
4 10000 3000
5000 2000 6000 10000
3 10000 2000
8200 5900 10000
0
```

Sample Output for the Sample Input

```
5000000
3246667
```

Problem E
Career Planning Problem
Input file: `pe.in` Time limit: 2 seconds

Ace-Career-Mining (ACM) is a renowned software engineering and training institute where monastic principles drive technological development. ACM has been helping many students master programming skills through rigorous on-site training. Every semester there are many students enrolling in various bootcamp training programs. The reason why it becomes so popular is that any one who can pass the program usually will have multiple job offers. Many corporates favor applicants with training records from ACM. Therefore, a certificate from ACM is virtually an IT career guarantee.

Joe is a senior manager of ACM in charge of career planning business. Joe's main task is help qualified students to find suitable jobs, and on the other hand help the corporates recruit employees. For each training program, when the semester ends, Joe will have a list of students who passed the program and a list of job posts from the corporate clients, where each student has a preference list over the posts. Note that each preference list is in strictly deceasing order. We assume that the numbers of qualified students and job posts are equal.

Suppose $A = \{a_1, a_2, \ldots, a_N\}$ is the set of qualified students, and $B = \{b_1, b_2, \ldots, b_N\}$ is the set of job posts, where each member of A has a preference list over some members of B.

For convenience, we use i to index the i-th student and post respectively, for $i = 1$ to N. An arrangement M between the students and posts is defined as: for a qualified student x, $M(x)$ denotes the post arranged for x and $M(x)$ must be in x's preference list. If $M(x)$ is not defined, then it means that x is jobless under M.

For any two arrangements M_0 and M_1, we say a student u prefers M_0 to M_1 if u is better-off in M_0 than in M_1, i.e., u is either assigned a job in M_0 and jobless in M_1 or assigned in both and prefers $M_0(u)$ to $M_1(u)$. We say M_0 is more popular than M_1, if the number of students that prefer M_0 to M_1 exceeds the number of students that prefer M_1 to M_0. We call an arrangement M a *happy* one if there is no arrangement that is more popular than M. Your task is to help Joe write a program to find out whether the preference lists of students admit a happy arrangement or not.

Technical specification

1. All the numbers are positive integers.
2. K: the number of test cases. $K \leq 20$.
3. N: the number of qualified students and job posts, respectively. $N \leq 1000$.
4. m_i: the length of the i-th student's preference list. $0 < m_i \leq N$.
5. $pm_{i,j}$: the i-th student's jth preference, $pm_{i,j} \in B$.

Input Format

The first line of the input file contains an integer $K(\leq 20)$ indicating the number of test cases to follow. Each test case starts with a positive integer $N(\leq 1000)$ indicating the number of students and the number of job posts. Then, N lines follow. In a line with multiple integers, two adjacent ones are separated by at least one space character.

The i-th line of the N following lines, has the format $m_i \;\; pm_{i,1} \;\; pm_{i,2} \ldots pm_{i,m_i}$, where m_i indicates the length of the i-th student's preference list and the following m_i distinct numbers $pm_{i,1} \;\; pm_{i,2} \ldots pm_{i,m_i}$ indicate the indices of preferred job posts with preference in decreasing order, i.e., the i-th student prefers $pm_{i,j}$ to $pm_{i,j+1}$.

Output Format

For each test case, output YES in a separate line, if it admits a happy arrangement; NO otherwise.

Sample Input

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

Sample Output for the Sample Input

```
YES
NO
YES
```

Problem F
Ranking Number
Input file: `pf.in`   Time limit: 5 seconds

There is an International Communication and Pastime Company (ICPC) with k departments. Some important managers of this company plan to propose a cross-department project. They first organize a project committee to handle this project. The project committee members are invited from k departments such that each department has at least one member invited to join the project committee. Note that any two departments can share members iff they are in the project committee. Two members X and Y can *directly* discuss the project, denoted by X$\leftrightarrow$ Y, if and only if they are in the same department or they are both in the project committee. In addition, two members X and Y can *indirectly* discuss the project if and only if they can discuss the project via another $t-2$ members $\mathrm{M}_1(=\mathrm{X}), \mathrm{M}_2, \ldots, \mathrm{M}_{t-1}, \mathrm{M}_t(=\mathrm{Y})$ such that $\mathrm{M}_1 \leftrightarrow \mathrm{M}_2 \leftrightarrow \mathrm{M}_3 \leftrightarrow \cdots \leftrightarrow \mathrm{M}_{t-1} \leftrightarrow \mathrm{M}_t$, where $t \geq 3$. We call M_i for $2 \leq i \leq t-1$ *key persons* for X and Y.

For some security reason, the chief executive officer of this company asks the Human Resources to assign a "*ranking number*" to each member X using $r(\mathrm{X}) \in RN = \{1, 2, \ldots, \alpha\}$ for some $1 \leq \alpha \leq 10001$ such that the following *ranking conditions* hold:

1. Two persons from the same department or from the project committee must receive different ranking numbers.

2. If there exist two persons, X and Y, with the same ranking number q, then they can indirectly discuss the project and there exists a key person Z for X and Y such that the ranking number of Z is higher than q (i.e., $r(\mathrm{Z}) > r(\mathrm{X}) = r(\mathrm{Y}) = q$).

Given a company with k departments $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$, where $1 \leq k \leq 1000$, and the project committee $\mathcal{D}_0$, let $|\mathcal{D}_j|$ for $0 \leq j \leq k$ denote the number of members in $\mathcal{D}_j$, where $1 \leq |\mathcal{D}_0| \leq 10000$ and $2 \leq |\mathcal{D}_i| \leq 10001$ for $1 \leq i \leq k$. Your task is to provide a computer program to compute the minimum α in RN such that the ranking conditions hold. We call such an α *optimal ranking number.*

For example, assume that there are three departments $\mathcal{D}_1, \mathcal{D}_2$, and $\mathcal{D}_3$ in the company. $\mathcal{D}_1$ has three members: A, B, and C; $\mathcal{D}_2$ has two members: D and E; and $\mathcal{D}_3$ has four members: F, G, H, and I. The members of project committee $\mathcal{D}_0$ are A, D, and F. Then, we can assign the ranking numbers to the members with $r(\mathrm{A}) = 3$, $r(\mathrm{B}) = 1$, $r(\mathrm{C}) = 2$, $r(\mathrm{D}) = 4$, $r(\mathrm{E}) = 1$, $r(\mathrm{F}) = 5$, $r(\mathrm{G}) = 3$, $r(\mathrm{H}) = 1$, and $r(\mathrm{I}) = 2$. In this assignment, although A and G have been assigned the same ranking number 3, they can indirectly discuss the project via F with A$\leftrightarrow$F$\leftrightarrow$G, where $r(\mathrm{F}) = 5 > r(\mathrm{A}) = r(\mathrm{G}) = 3$. The similar reasons can apply to members C and I, and any two members in {B,E,H}. Hence, the ranking conditions hold. In this case, $\alpha = 5$. However, there exists another ranking with $r(\mathrm{A}) = 3$, $r(\mathrm{B}) = 1$, $r(\mathrm{C}) = 2$, $r(\mathrm{D}) = 2$, $r(\mathrm{E}) = 1$, $r(\mathrm{F}) = 4$, $r(\mathrm{G}) = 3$, $r(\mathrm{H}) = 2$, and $r(\mathrm{I}) = 1$. In this assignment, $\alpha = 4$. In fact, the optimal ranking number equals 4.

Input Format

For convenience, we use $\{1, 2, \ldots, i\}$ to denote i project committee members.

The first line of the input file contains an integer, denoting the number of test cases to follow. For each case, the k departments and $|\mathcal{D}_0|$ project committee members are given the following format: The first line contains two positive integers, k and $|\mathcal{D}_0|$, separated by a space. In the following k lines, each line contains $m + 2$ positive integers and any two consecutive integers are separated by a space. The first positive integer m indicates the number of members invited from the corresponding department to join the project committee; the second positive integer indicates the number of members in the department; and the following m positive integers represent the project committee members.

Output Format

For each case, output the optimal ranking number.

Sample Input

```
3
3 3
1 3 2
1 2 1
1 4 3
4 3
1 4 3
1 2 1
1 3 1
1 3 2
2 4
2 9 1 3
2 12 2 4
```

Sample Output for the Sample Input

```
4
5
12
```

Problem G
Smartphone Manufacturing
Input file: `pg.in`   Time limit: 5 seconds

In a highly competitive industry, such as the information technology industry, finding the right manufacturing partners is critical to ensure successful product release. Therefore, the *ACM* (Advanced Communication Machinery Corp.) is having difficulty choosing between one of three possible manufacturing companies to produce it's next generation smartphones. The three companies, namely company A, B and C, all have a number of workers. For each company, the workers are split into two groups, working in two shifts: one group will work in one week; the other group work the following week; and then back to the first group for the third week, and so on. *ACM* wants to ensure the weekly phone manufacturing is as consistent as possible, so it has developed a complicated, yet accurate, formulation to predict each worker's weekly throughput. Let t_{Ai}, t_{Bi}, t_{Ci} denote the throughputs of *i-th* worker in company A, B, C, respectively. The criteria used for choosing manufacturing partner is as follows:

- Criteria 1: The entire workforce of the company must be used. No one can be left out.
- Criteria 2: The variation between weekly total phone output must be as small as possible.
- Criteria 3: The total phone output over a year (52 weeks) must be as large as possible.

Note that *Criteria 1* takes precedence over *Criteria 2* and *Criteria 2* takes precedence over *Criteria 3*.

Given information about the workforce in manufacturing companies A, B, and C. Please decide which company should *ACM* choose to partner to manufacture its latest smartphone.

Technical Specifications

1. There are at least 2 and at most 1,000 workers in each of the three manufacturing companies.
2. The worker throughput is an integer between 1 and 500.

As an example, in the first sample input below, company A has two workers (throughput of 100 and 80). Company B has eight workers, which can be divided up into two shifts of equal phone output (11) per week. Company C has four workers which can be divided up into two shifts of equal phone output (3) per week. Therefore, company B will be chosen by ACM because if has the least variation (actually no variation) between weekly phone output, yet produces more phone in 52 week period than company C.

Input Format

The first line of the input file contains an integer n, denoting the number of test cases to follow. For each test case, there are 3 lines of integers, denoting the throughput of the workers in Company A, B and C, respectively. For each line, the first integer m denotes the number of workers in that company, followed by m integers denoting throughput of the m workers for that company.

Output Format

For each test case, output either A, B, or C, denoting the company that best meets *ACM*'s selection criteria, followed by the weekly throughput for the two shifts of workers, smaller throughput should be printed first.

Sample Input

```
2
2 100 80
8 1 3 2 5 7 1 2 1
4 2 1 1 2
2 100 80
3 20 80 40
10 1 1 1 1 1 1 1 1 1 30
```

Sample Output for the Sample Input

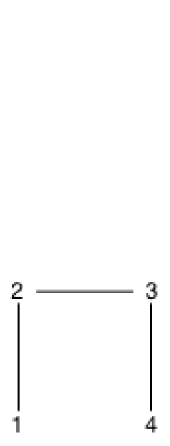
```
B 11 11
A 80 100
```

Problem H
Grass
Input file: `ph.in` Time limit: 2 seconds

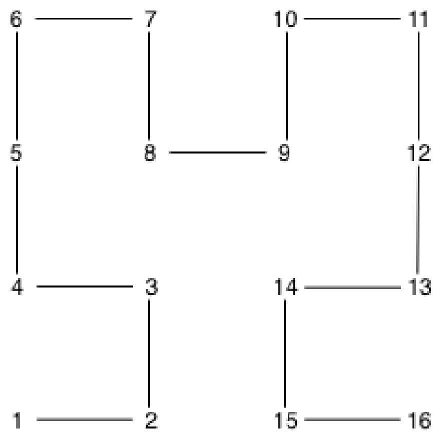
Owen is a college student, and he earns his tuition fee by cutting grass for his neighbors in the weekends. Unlike the other weeding service, Owen's service is charged by the number of turns made when cutting grass, and he has to weed at least one cell of grass, which is a one meter by one meter square area, between every two turns. Given a 2^n meter by 2^n meter grass area, Owen develops a cutting path planning algorithm as follows.

1. First, he divides the grass area to four equal-sized square subarea (i.e., each one is a 2^{n-1} meter by 2^{n-1} meter square).
2. He applies the solution of 2^{n-1} meter by 2^{n-1} meter grass area to both of the *northwest* and *northeast* subareas.
3. He rotates the solution of 2^{n-1} meter by 2^{n-1} meter grass area by 90 degree clockwise, and applies the rotated results to the *southwest* subarea.
4. He rotates the solution of 2^{n-1} meter by 2^{n-1} meter grass area by 90 degree counterclockwise, and applies the rotated results to the *southeast* subarea.
5. He always starts from the most southwest corner and ends at the most southeast corner of the grass area. Moreover, he always visits the *southwest* subarea first, followed by the *northwest* subarea, the *northeast* subarea, and the *southeast* subarea.

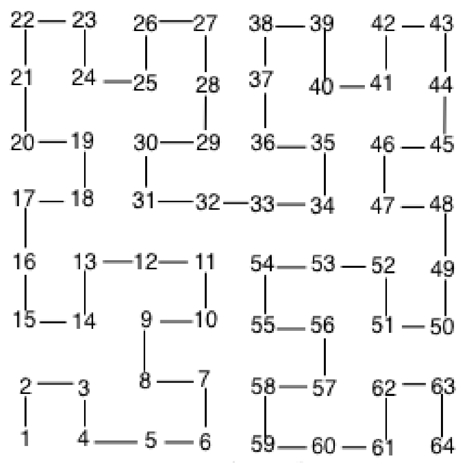
The figures below show Owen's moving sequence in the grass area, when n = 1, 2, and 3, respectively.



n = 1



n = 2



n = 3 (scaled)

We let the coordinate $(0, 0)$ represents the most southwest cell and $(n-1, n-1)$ be the most northeast cell in the 2^n meter by 2^n meter grass area. Your task is to find the coordinate of the k-th visited cell, as well as the coordinate of its neighbouring cell that has the greatest distance (i.e., the difference of the visiting sequence number) to the k-th visited cell, based on

Owen's approach. For instance, when $n = 2$ and $k = 14$, the coordinate of the k-th visited cell is $(2, 1)$. The neighbouring cells of $(2, 1)$ are $(2, 0)$, $(1, 1)$, $(2, 2)$, and $(3, 1)$; and their visiting sequence numbers are 15, 3, 9, and 13 respectively. Therefore, the greatest distance is 11 because $14 - 3 = 11$, and the cell that results in the greatest distance is $(1, 1)$.

Input Format

There are multiple test cases in the input file. For each test case, there are two integers separated by a single white space in a line. The first integer is n indicating the grass area is a 2^n meter by 2^n meter square, and the second integer is k. In this problem, we assume that $1 \leq n \leq 15$ and $1 \leq k \leq 2^{2n}$. A test case starting with 0 denotes the end of the input file.

Output Format

For each test case, output the coordinate of the k-th visited cell, followed by the coordinate of its neighbouring cell of the greatest distance, in a line. All the coordinates are output in the order of the west-east direction/axis first and the south-north direction/axis next; and all output numbers are separated by a single white space. If there are multiple neighbouring cells that have the greatest distance to the k-th cell, please output them in ascending order with preference to the west-east direction/axis first and the south-north direction/axis next.

Sample Input

```
1 3
2 14
3 23
0
```

Sample Output for the Sample Input

```
1 1 0 1 1 0
2 1 1 1
1 7 2 7
```

Problem I
Airline Companies
Input file: `pi.in` **Time limit: 2 seconds**

A country named *Acmland* is very famous because of its convenient air traveling. Acmland has n cities, namely $c_1, \dots, c_n$. Two different cities may be connected by an *air corridor*. For example, we use $e_{i,j}$ to denote the air corridor between city c_i and c_j. Because an air corridor is bi-directional, we can go from city c_i to c_j, and from city c_i to c_j, if the air corridor $e_{i,j}$ exists. Acmland has m air corridor, and for ease of notation we always assume that $i < j$ when we describe an air corridor $e_{i,j}$.

The air corridors in Acmland form an *air corridor network*. An air corridor network has the following characteristic – if there is a cycle of length greater than three, then there will be a *shortcut* air corridor connecting two cities with this cycle. For example, Figure 3 illustrates an air corridor network of 16 cities and 20 air corridors. We find a cycle of length 5 in this air corridor network, namely c_3, c_5, c_8, c_7, and c_{16}, then we can find a shortcut air corridor, e.g. $e_{5,7}$, to connect c_5 and c_7. Now consider a cycle of length 4, c_3, c_5, c_7, and c_{16}. Again we find a shortcut air corridor, $e_{5,16}$ that connects c_5, c_{16}.

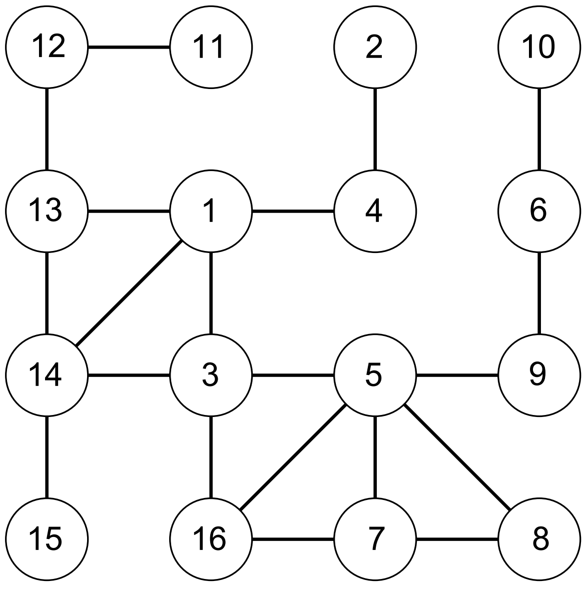


Figure 3: An example of air corridor network

Acmland has k airline companies. Each airline companies is located in a unique city. That is, no two airline companies are located in the same city.

Now a financial crisis is hitting Acmland and all airline companies are in trouble. The Acmland government wants to help airline companies by providing financial aids to them. Although the government wants to help as many airline companies as possible, the government regulation says that the government can only help *independent* airline companies. Two airline companies are *independent* if and only if they are located in two cities that are *not* connected by a air corridor. For example airline company A, B, and C are located at cities 3, 9, and 6. Then airline company A and B are independent, because air corridor $e_{3,9}$ does *not* exist. On the other hand, airline company B and C are *not* independent because air corridor $e_{6,9}$ does exist.

Now given the air corridor network and the cities all airline companies are located, please compute the *maximum* number of airline companies that the Acmland government can provide financial aids.

Input Format

An instance of the problem consists of

1. the number of cities n,
2. the number of air corridors m,
3. the number of airline companies k,
4. the indexes of cities of every air corridor, i and j, where $1 \leq i, j \leq n$ and an air corridor will appear exactly once,
5. and the indexes of cities all airline companies are located.

These data are stored in $2 + m$ lines in the input file.

1. The first line has integer n, m, and k.
2. The following m lines are the the indexes of cities of every air corridor, i and j, where $1 \leq i < j \leq n$.
3. The next line has the indexes of cities where all airline companies are located.

In this problem, we assume that $1 < n, m \leq 100000$, and $1 \leq k \leq 10000$. Note that a test data file may contain more than one instance. The last instance is followed by a line containing a single 0.

Output Format

The output for each instance is one integers N, which is the maximum number of airline companies that the Acmland government can provide financial aids.

Sample Input

```
16 20 3
11 12
12 13
13 14
14 15
3 14
1 13
1 14
1 3
1 4
```

```
2 4
3 5
3 16
5 16
5 7
7 16
7 8
5 8
5 9
6 9
6 10
3 6 9
0
```

Sample Output for the Sample Input

```
2
```

Problem J
Balanced Community
Input file: `pj.in` **Time limit: 5 seconds**

A social network can be modeled as a graph in which a node represents an actor and an edge represents some social relation between the two actors. For some applications, the edges may be marked with *positive* or *negative*. Usually a positive edge means a positive emotion while a negative edge is for a negative emotion, such as "liking" or "disliking" respectively.

Imbalance-Compromise-People-Club, ICPC for short, is an Internet social platform providing communications for its members. However, unlike FaceBook, people in ICPC either like or dislike each other. In other words, any two members in ICPC are either friends or enemies. Therefore the relationship can be modeled by a complete graph in which each edge is either positive or negative.

Nana, a member of ICPC, found that these relations are unstable. From time to time, some positive relations change to negative and vice versa. Nana wondered why it happened, so she asked experts the reason. According to their knowledge in social network analysis, a relation changes because there exists some *imbalance*. For any three actors, there are four possible graphs (triples) as shown in Figure 4. The left two triples in the figure are considered as *balanced* triples while the right two triples are *unbalanced*. Furthermore, a group of actors is balanced if and only if any triple in the group is balanced.

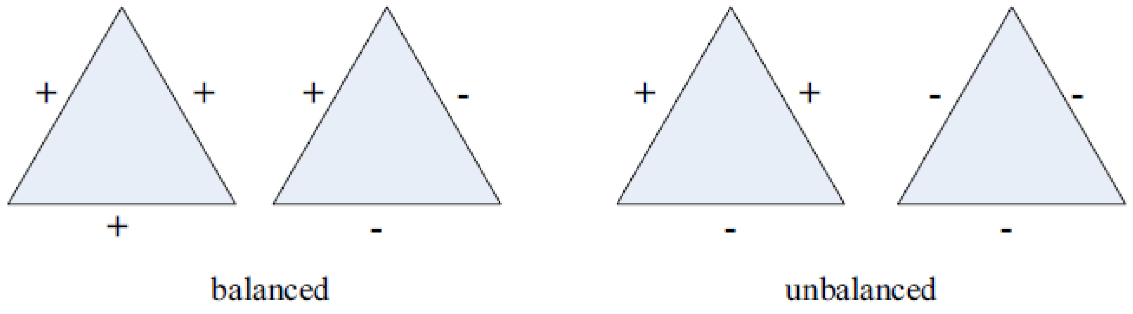


Figure 4: Balanced and unbalanced triples.

Now, after knowing the reason, Nana also wants to find out the largest balanced group including herself. Your task is to write a program for finding such a group. Let n denote the number of members in ICPC. For simplicity, all the members have a unique ID which is an integer from 0 to $n-1$. The ID of Nana is 0.

Input Format

The input contains several test cases. For each test case, the first line contains two integers n and m, in which n is the number of members in the club and m is the number of positive edges. It is assumed that $n \leq 122$. In the following m lines, each line contains two integers separated by a space, which are the endpoints of a positive edge. Remember that there is an edge between any pair of nodes. Therefore if two nodes are not linked by a positive edge, there is a negative edge. A case with $m = n = 0$ indicates the end of the input and you don't need to process it.

Output Format

For each test case, your program should output the maximum number of members of a balanced group which contains Nana. The result of each test case should be in an individual line.

Sample Input

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

Sample Output for the Sample Input

```
4
4
```