2009 National Collegiate Programming Contest

- Problems: There are 7 problems (13 pages in all, not counting this cover page) in this packet.
- Problem Input: Input to the problems are through the input files. Input filenames are given in the table below. Each input file may contain one or more test cases. Test cases may be separated by any delimiter as specified in the problem statements.
- Problem Output: All output should be directed to standard output (screen output).
- Time Limit: The judges will run each submitted program with certain time limit (given in the table below).

Table 1: Problem Information Sheet

	Problem Name	Input File	Time Limit
Problem A	Fast Computation of Exponentiation	pa.in	3 sec.
Problem B	Color	pb.in	3 secs.
Problem C	Paper Folding Game	pc.in	2 secs.
Problem D	Food Centers	pd.in	10 secs.
Problem E	Merger	pe.in	10 secs.
Problem F	A Renting Problem	pf.in	10 secs.
Problem G	Survivor	pg.in	5 secs.

Problem A
Fast Computation of Exponentiation

Input File: *pa.in*
Time Limit: *3 sec.*

Problem Statement

In cryptography, a fast computation of integer exponentiation in $Z_n = \{0, 1, \ldots, n-1\}$, where n is a positive integer, is very important. Let g, h, n be positive integers, define

$$x \equiv g^h \ mod \ n,$$

where x is the remainder of g^h divided by n.

This problem asks you to write a *fast program* to compute x with given intgers g, h, n, where h and n could be as large as up to $2^{26} - 1$ within 3 seconds for up to 5 cases.

Technical Specification

1. $1 < g < n < 2^{26}$
2. $1 < h < 2^{26}$

Input File Format

The first line of the input file always contains one integer, $K \leq 5$, indicating the number of test cases to come. Each of the following lines contains three integers g, h, n based on which, you need to compute $x \equiv g^h \ mod \ n$.

Output Format

For each given triple of g, h, n, report g, h, n, x, where $x \equiv g^h \ mod \ n$.

Sample Input

```
5
2  7  127
3  4  7
22  1234567  4097
25  4194303  32767
31  67108863  65535
```

Sample Output

```
5
2  7  127  1
3  4  7  4
22  1234567  4097  1863
25  4194303  32767  15625
31  67108863  65535  63421
```

Problem B
Color

Input File: *pb.in*
Time Limit: *3 sec.*

Problem Description

The FunFun Waterpark has installed a new game that is mentally challenging. The game surface is composed of rectangular grid of *mxn* cells. At the onset of the game, water hoses are placed under s distinct cells and s distinct color dyes (numbered from 1, ..., s) are place on top of these distinct cells. The water hoses are turned on at beginning of the game, filling the cell it is on with distinct "colored" water. With each passing unit of time, the "colored" water is spilled over to it's neighboring cells that are immediately to the north, south, east, and west directions. So eventually, the water will flood the entire game surface. Special insulation is placed on the game surface, so that when a cell is filled with water of a particular color, then no water of other color can spill into that cell. Furthermore, if two or more source (distinct color) of water can spill into a cell at the same time, then the cell is filled with water with a lower dye number.

In the example below (which corresponds to Sample Input 1), the game surface is composed of 10x6 cells. Five hoses and five distinct color dyes are placed on five distinct cells as shown in (a). At the first unit of time, the water are spilled over to neighboring cells as shown in (b). After the 5th unit of time, the water will have covered the entire game surface as shown in (c).

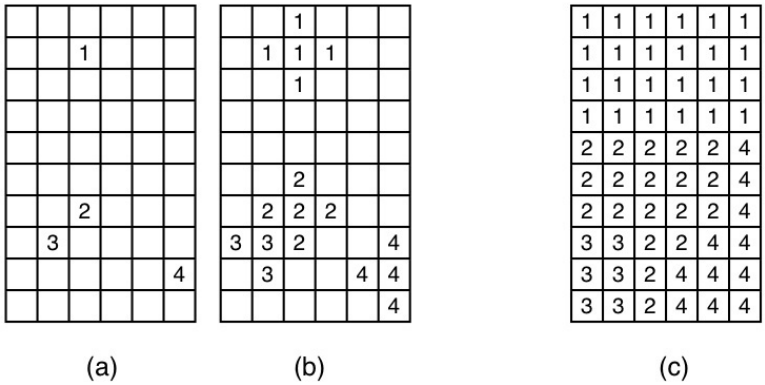

Figure 1: (a) Initial game setup, (b) after one unit of time, (c) after five unit of time.

Please write a program to determine which color of water will have occupied the most game cells. If there is a tie, print all that tied from the smallest color number to the largest colored number.

Technical Specification

1. The grid size is $1 <= m, n <= 1,000$. The cells are labelled from (1, 1) ... (m, n) in a two-dimensional grid.

2. The initial number of hoses and dyes is $s, 1 \leq s \leq 20$.

Input File Format

The first line of the input contains an integer t, indicating the number of test cases to follow. The first line of each test case contains four integers, m, n, and s separated by a space. The next s lines each contains two integers row_i and col_i, indicating the location of the hose and color dye i.

Output Format

For each game played, output color dye number(s) that would occupy the most number of cells when the entire game surface is filled with water. If more then two colored water covered equal number of cells, then output all in the order of their color number from smallest to largest.

Sample Input

```
2
10 6 4
2 3
7 3
8 2
9 6
3 6 5
1 6
2 4
2 3
2 2
3 1
```

Output for the Sample Input

```
1
2 4
```

Problem C
A Paper Folding Game

Input File: *pc.in*
Time Limit: *2 sec.*

Problem Description

A paper strip as seen in Fig. 1 is composed of many squares on both sides (called side A and side B). These squares are indexed from 0, 1, 2, and so on from one end of the paper strip to the other end. We use (i, A) to refer to the square with index i at side A and (i, B) to refer to the square with index i at side B. Each square has a score ranging from -10 to 10.

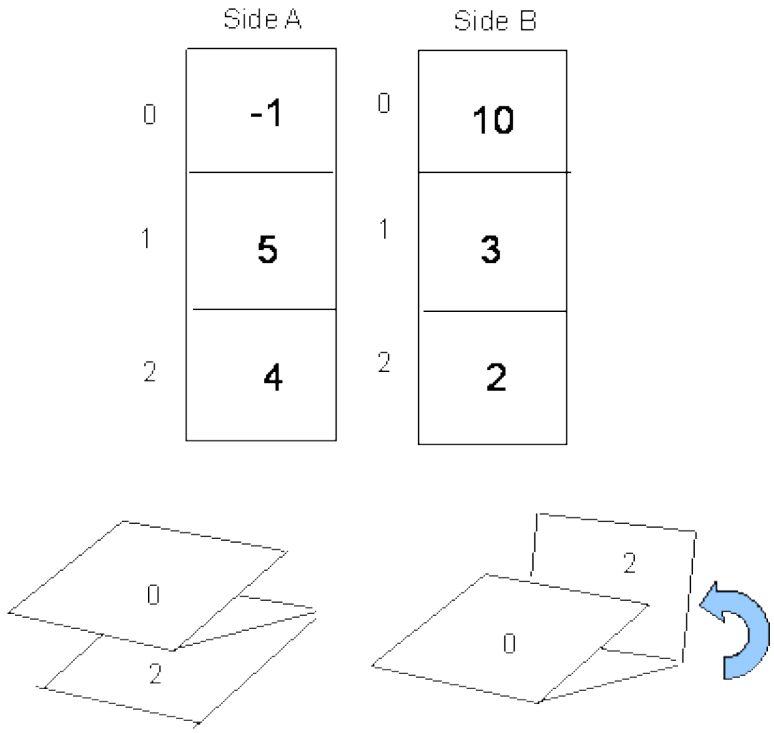

Figure 2: An example to fold a 3-square paper strip.

The game begins by folding square 0 onto square 1. There are two choices to fold the paper, either you fold $(0, A)$ onto $(1, A)$ or $(0, B)$ onto $(1, B)$. When a choice is made, two touched squares are considered as sealed; that is, it is impossible for these two squares to touch any remaining squares. Continuously, there are two choices to fold the sealed square 0 and square 1 onto square 2 (see the bottom figure of Fig. 1). This process continues until you fold from one end all the way up to the other end.

In each folding, you get points by multiplying the scores on two touched sides. For example, in Fig. 1, folding $(0, A)$ to $(1, A)$ you get -5 points but folding $(0, B)$ to $(1, B)$ you get 30 points. The total score of the game is adding all the points of every folding. The example in Fig. 1 has maximum points of 50.

Given a paper strip with S squares, please output the maximum score a best folding can make.

Technical Specification

1. $2 \leq S \leq 30$

Input File Format

The first line of the input file contains an integer indicating the number of test cases to follow. Each test case begins with an integer S, which is the number of squares of a paper strip. Following S is S lines of scores that is on side A followed by the score that is on side B.

Output Format

For each test case, please output the maximum total score that a best folding can produce.

Sample Input

```
2
3
-1 10
5 3
4 2
3
-10 5
-10 3
-10 2
```

Output for the Sample Input

```
50
115
```

Problem D
Food Centers

Input File: *pd.in*
Time Limit: *10 sec.*

Problem Description

Recently Taiwan was hit by a typhoon so the government sets up a network to distribute food to people in the affected area. The network is a tree, every leaf is a shelter, and every edge is a road connecting two nodes in the tree. Every road is 1 kilometer in length.

Now we want to assign a subset of tree nodes to be *food centers* that distribute food. The food center can be set up in any tree node, including the leaves where the shelters are located. The rule is that if there is no food center at a shelter, people living in that shelter will have to go to the nearest food center by traveling along the unique path towards the root, and get food at the first food center they find. After they get food they will travel back to the shelter where they live. In addition, people in different shelters have different physical strength, so there is a limit in how far they can travel for food. For example, if people in a shelter have physical strength 2, then they can only travel to the next tree node towards the root, and then go back to their shelter. If people in a shelter have physical strength 0, then they must have a food center located right at their shelter.

Since the government only has a limited budget, it can only build k food centers. Also since it is always possible that people may reach the root for food, there must be a food center at the root, therefore we only need to choose locations for the remaining $k-1$ food centers. While choosing the locations for food centers, we want to balance the number of people that will be served by these k food centers. For ease of notation we use *workload* of a food center to denote the number of people severed by that food center. Now the problem is, given the number of people at each shelter and their physical strength, we want to choose the locations for the $k-1$ food centers, so that all people can reach a food center within their physical strength limit, and the *maximum* workload among all food centers is *minimized*.

Technical Specification

1. The number of test cases (c) is no more than 10.
2. The number of shelters (n) is no more than 10000.
3. The number of food centers (k) is no more than n.
4. The number of people living in a shelter is no more than 100.
5. The physical strength is no more than 100.

Input File Format

The first line of the input file contains an integer (c) indicating the number of test cases to follow. The first line for a case has two positive integers n and k for the number of tree nodes and the number of food centers to be built. The tree nodes are numbered from 0 to

$n-1$. Each of the next $n-1$ lines have three integers that describe a tree node (root has index 0 and does not need to be described). The first number of the i-th line is the index of the parent of i-th tree node, the second non-negative integer is the number of people living in the i-th tree node, and the third non-negative integer is the physical strength of the people living in that tree node. Note that if the tree node is an internal node, i.e., a non-leaf node, it does not have a shelter, so the number of people and the physical strength are both 0.

Output Format

The output has c lines. The i-th line has the minimum possible maximum workload among all food centers for the i-th test case. Note that for the inputs you are given it is always possible to find a solution such that everyone can reach a food center within his physical strength.

Sample Input 1

```
1
5 3
3 5 0
3 35 2
4 0 0
0 0 0
```

Sample Output for the Sample Input 1

```
35
```

Sample Input 2

```
1
10 3
3 0 0
3 0 0
0 0 0
1 0 0
4 0 0
2 26 2
5 24 6
1 84 5
1 66 10
```

Sample Output for the Sample Input 2

```
174
```

Problem E
Merger

Input File: *pe.in*
Time Limit: *10 sec.*

Problem Description

In business or economics, a merger is a kind of corporate strategy that can help a company grow rapidly without creating another business entity. In general, a merger comprises of two companies: one is the buyer, and the other is the seller. Moreover, after a merger, the formed company is named after the buyer company, and its stock value (V) is determined by the formula: $V = \alpha X + (1 - \alpha)Y$; where X and Y are the stock values of the buyer and the seller, and α is a scaling factor in the range of [0,1].

Suppose there are N companies located along a line, and let C_i denote the i-th company. There are two rules that must be followed in a merger:

1. The buyer must be adjacent geographically to the seller (i.e., there are no companies, among the N companies, located between the buyer and the seller).
2. The N companies have to collaborate in the process of the merger, so that they can become a single company with the highest stock value.

For instance, let $\alpha = 0.50$ and there are three companies, C_1, C_2, and C_3, along a line geographically. The initial stock values of the three companies are 10, 20 and 30 respectively. To maximize the stock value after merging them to one company, one of the best strategies is to let C_1 buy C_2 first, and C_1 buy C_3 next. Therefore, after the first merger, the formed company is named after C_1 and its stock value is 15.00. Then, after the second merger, the formed company is named after C_1 and its stock value is 22.50.

Technical Specification

1. N is an integer, and $0 < N \leq 1,000$.
2. The initial stock value of a company is a positive integer smaller than or equal to 10,000.
3. α is a real number, and $0 \leq \alpha \leq 1$.

Input File Format

The first line of the input file contains an integer indicating the number of test cases to follow. For each test case, the first line contains a positive integer N, representing the number of companies to merge; and the second line contains a positive real number α, which has exact two digits after decimal. In the following N lines, the i-th line contains the initial stock value (which is a positive integer) of the i-th company, which is in the same order of their geographical location (from the west to the east).

Output Format

Please output one number in one line for each test case. The number represents the highest stock value of the company after the merger. The value of the output number should be round off to two digits after decimal.

Sample Input 1

```
1
3
0.50
10
20
30
```

Sample Output for the Sample Input 1

```
22.50
```

Sample Input 2

```
1
3
0.50
10
10
10
```

Sample Output for the Sample Input 2

```
10.00
```

Problem F
A Renting Problem

Input File: *pf.in*
Time Limit: *10 sec.*

Problem Description

Assume that there are n different available processors. For each processor, we can use it alone or sharing with others. For convenience, we call them *alone* and *sharing* processors, respectively. We want to rent k alone processors from them and execute simultaneously to finish our task. Clearly, each alone processor is more expensive than its sharing processor. An alternative way is that each alone processor can be replaced by exactly 3 different sharing processors if their total cost is cheaper than the alone processor. The constraint on the number of processors can be described by the equation: $3 * |S_\alpha| + |S_\beta| = 3 * k$ where S_α and S_β are the sets of the selected alone and sharing processors, respectively. We can see that if $S_\beta = \emptyset$ (respectively, $S_\alpha = \emptyset$), then $|S_\alpha|$ (respectively, $|S_\beta|$) must be equal to k (respectively, $3 * k$) to fulfill the requirement. If $|S_\beta| = 3$, then $|S_\alpha|$ can only be equal to $k - 1$, etc. What we want is to find the cheapest way to rent processors with sets S_α and S_β in order to finish our parallel task.

For instance, assume that the set of available processors

$$S = \left\{ \begin{array}{cccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ 8 & 10 & 14 & 20 & 42 & 44 \\ 3 & 6 & 12 & 18 & 2 & 1 \end{array} \right\}$$

where $s_1 = (8, 3), s_2 = (10, 6), \ldots$, and $s_6 = (44, 1)$. That is, the costs to rent processor s_1 as an alone and a sharing processor are 8 and 3 dollars, respectively, and the corresponding renting costs for processor s_2 are 10 and 6 dollars, respectively, etc. Assume that our parallel task only needs two alone processors, i.e. $k = 2$. Thus, the constraint on the number of processors can be formulated as $3 * |S_\alpha| + |S_\beta| = 6$. We can see that $S_\alpha = \emptyset$ and $S_\beta = \{s_1, s_2, s_3, s_4, s_5, s_6\}$ satisfy the constraint. Its corresponding value is $\sum_{s_i \in S_\alpha} \alpha_i + \sum_{s_i \in S_\beta} \beta_i = 3 + 6 + 12 + 18 + 2 + 1 = 42$. However, in this example, $S_\alpha = \{s_2\}$ and $S_\beta = \{s_1, s_5, s_6\}$ achieve the cheapest way to rent processors. The corresponding value is $\sum_{s_i \in S_\alpha} \alpha_i + \sum_{s_i \in S_\beta} \beta_i = 10 + 3 + 2 + 1 = 16$.

Technical Specification

1. N is an integer, and $0 < N \leq 1,000$.
2. The initial stock value of a company is a positive integer smaller than or equal to 10,000.
3. α is a real number, and $0 \leq \alpha \leq 1$.

Input File Format

The first line of the input file contains an integer m, $m \leq 4$, which represents the number of test cases. Each test case contains three lines of input data. The first line of a test

case contains two positive integers n (≤ 100) and k (≤ 50) which are the aforementioned variables. The second line of a test case contains n integers which are the renting prices of n alone processors. Note that no price is greater than 100. The third line also contains n integers which are the renting prices of n sharing processors corresponding to their alone processors listed in the previous line. For example, in the following sample input, the first test case contains three processors. The prices for renting alone processors are $9, 10$, and 16, respectively, and the prices for renting their respective sharing processors are $6, 9$, and 2, respectively.

Output Format

For each test case, output the cheapest price for renting renting processors in one line.

Sample Input

```
2
3 2
9 10 16
6 9 2
6 3
8 10 14 20 42 44
3 6 12 18 2 1
```

Sample Output

```
19
30
```

Problem G
Survivor

Input File: *pg.in*
Time Limit: *5 sec.*

Problem Description

Below is a simple game which is difficult to predict the outcome. Let n people sit in a circle, numbered from 1 to n, in clockwise ordering, i.e. person number 2 sits on the left hand side of number 1. Given a number k , starting from number 1, every k person will leave the game. Determine the number of the final survivor. For example, if n = 10 and k = 3, the order list of the person number to leave should be 3, 6, 9, 2, 7, 1, 8, 5, 10, so the final survivor is person number 4.

Technical Specification

1. $2 \leq n \leq 10000$
2. $1 \leq k \leq 1000$

Input Format

There are five lines in the input file, and each line contains two integers n, k separated by a space. In other words, there are five cases to be solved.

Output Format

For each test case output the number of the final survivor in one line.

Sample Input

```
5  3
10 3
15 7
10000 499
10000 100
```

Sample Output

```
4
4
5
1429
9332
```