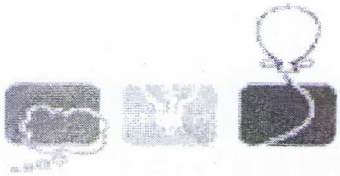


9th ACM Egyptian National Programming Contest

Faculty of Computers and Information,

Cairo University

29 October 2010

Dear contestants:

All programs will run in windows environment. Every program should read data from an input file and send results to the standard output. The data format is given and must be exactly followed. The only file to open is the input file with the given name for each problem.

We wish you a lot of fun and good luck with solving these problems.

[A] Sir, Yes Sir

Program: sir.(c|cpp|java)
Input: sir.in
Balloon Color: Grey

"Special Dedication to Hamza Darwish, Ahmed Saad, and Karim Nosseir"

Egyptian military is very well known with victories along the history. Egyptian soldiers are well known with bravery and sacrifice which they showed in many battles. The most famous one in the recent Egyptian history is the 6th of October which had been the greatest victory for the Egyptians and all the Arab nations in the 20th century, and had been leaded by the past Egyptian passed away president El-Sadat. Before that with about 20 years, there was the revolution of July 23rd leaded by the Free Officers Movement which addressed the most powerful resistance to the Egyptian kingdom and high-hand, who were known by venality and corruption, and ended by starting a new age to Egypt as a republic. These great victories and achievements of the Egyptian military contributed in the creation of a special personality for the military members which appears in obeying orders of higher military ranks and loyalty.

General Hamza is visiting a military unit in the depth of Western Sahara. He checks every little detail, soldiers food, warehouses, weapons, ammunition, etc. He passed by Major Karim and Lieutenant Saad with their group of soldiers on sport training time. Once they saw the general all of them stopped and stand in attention with no move. The general was so angry that soldiers have a number on their training suits but they are standing in a queue with no order. He asked Major Karim and Lieutenant Saad to help the soldiers to sort themselves, but to make it very silent, and to achieve that, General Hamza just allowed for a soldier to swap his place with his neighboring soldier, and the mission of Major Karim and Lieutenant Saad is to sort the soldiers with minimum number of swaps.

Input Specification

The first line of input contains an integer T that represents the number of test cases. T test data set follow first line of each data set contains an integer ($1 <= N <= 10000$) the number of soldiers. N integers follow each number Mi represents the label on the soldier training suit and the soldier is standing at position i in the queue ($1 <= i <= N$).

Output Specification

For each test case, output one line contains an integer that represents the minimum number of swaps to sort the soldiers by their labels on training suits.

Sample Input:

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

Sample Output:

```
2
```

[B] Least Common Multiple

Program: lcm.(c|cpp|java)
Input: lcm.in
Balloon Color: Green

Our spy satellite has intercepted a secret message transmitted by moon rabbits. Thanks to our intelligence, we already know that each secret message consists of N integers. Because of encryption, we know only the least common multiple of all numbers in the message.
Your task is to find out how many different messages could have been sent.

Input Specification

Input file contains on the first line the number of test cases T, then follow T lines each contains only two integer numbers N and L - number of words in the message and their least common multiple.
($1 <= N <= 1{,}000$, $1 <= L <= 2{,}000{,}000{,}000$).

Output Specification

Output T lines each contains only one integer number, which is the number of different messages.

Sample Input

```
1
2 6
```

Sample Output

```
9
```

[C] Moon Craters

Program: mooncrater.(c|cpp|java)
Input: mooncrater.in
Balloon Color: White

Mooncats live on the Moon. It is very cold there, so they live in the Moon's crater. Every crater has its own height that counts from the sea level being found not so long ago. Each mooncat chooses some crater and in each crater can live only one mooncat. To simplify, we represent the Moon surface as a square matrix of size *N* X *N*, each crater is a cell in this matrix, and the value in a cell denotes the height of the crater in this position.

Since life is difficult, mooncats want to live in joy, happiness and pleasure. They are rather stupid, so their pleasure is determined by a special condition:
for every 1 <= i <= k <= *N* and 1 <= j <= l <= *N*, the following inequality is satisfied: $h_{ij} + h_{kl} <= h_{kj} + h_{il}$. Check if mooncats live in pleasure.

Input Specification

Input file contains on the first line the number of test cases *T*, then follow *T* cases each contains number *N* - size of the surface of the Moon (1 <= *N* <= 1000). Next *N* lines contain *N* numbers each - the heights of every crater h_{ij}. Every height does not exceed 10^9 by absolute value.

Output Specification

Output *T* lines each contains either "HAPPY" or "UNHAPPY" – without quotes.

Sample Input

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

Sample Output

```
HAPPY
UNHAPPY
```

[D] ACM Ceremony

Program: acm.(c|cpp|java)
Input: acm.in
Balloon Color: Yellow

You are the CEO of the Association of Cream and Milk (ACM), and you are going to hold the annual ceremony of the association. In this ceremony you are going to talk about the association history and association plans to grow-up in market and convince people to give-up soda and similar unhealthy drinks to replace it with milk. At the end of the ceremony it is intended to appreciate the high performing employees who have excellent feedback from their managers, made excellent work records, and drink milk usually. There are three types of employees who are going to be appreciated in the ceremony which are lite, butter, and full cream employees. Each of them is going to have a different appreciation than the other and that is according to their experience, how many years they are working at the association, and for sure the amount of milk they are drinking daily.

Now you are so good at programming, but on the other side your mathematical records are very poor and you need to make a very complex mathematical computation, according to your mathematical background, which is going to be like a nightmare. Beside that you don't trust mathematicians; you decided to depend on yourself to calculate this complex computation. Now you know the cost of each type of employees' appreciation, but you have two limitations. The first limitation is the amount of attendees because your place is not that big to hold too much people, and the second limitation is that you already allocated a limited budget for this appreciation ceremony. Given the cost of each employee type (X,Y and Z), maximum number of people the place can hold P, and the allocated budget B, Can you calculate a combination of (X, Y and Z) to invite at least one employee of each type with exact total of P people and costs B budget?

Input Specification

The first line of input contains an integer T that represents the number of test cases. T lines follow each line consists of five integers, the first integer ($3 <= P <= 30000$) is the maximum number of people the place can hold, the second integer ($0 <= B <= 30000$) is the allocated budget for the ceremony, and the last three integers ($0 <= X, Y$ and $Z <= 1000$) represent the costs of different employees types.

Output Specification

For each test case, output one line contains three space separated integers represents the count of people who are going to be appreciated with same order of input (x, y, and then z). If there are multiple solutions print the one that minimizes x, if tie print the solution that minimizes y, and if tie print the solution that minimizes z. If there is no solution print "Crap!" quotes are for clarity.

Sample Input

```
2
5 10 1 2 5
100 250 1 5 10
```

Sample Output

```
3 1 1
65 33 2
```

[E] Pug on the Moon

Program: moonpug.(c|cpp|java)
Input: moonpug.in
Balloon Color: Purple

There is one unique moonpug that lives on the moon. Mooncats know about this.They decided to send a satellite to the space to observe the moonpug. Moonpug is informed about it.And it is not interested in disclosure of its location. Moonpug knows that a satellite can't observe moonpug if it is located inside a lunar crater. Moonpug can run with a constant speed of one meter per second. And it needs to get from point A to point B in a way that minimizes time when mooncat's satellite observes the moonpug.

Input Specification

Input file contains on the first line the number of test cases T, then follow T cases each contains one integer number N the number of lunar craters (0 <= N <= 100). Each crater is a non-degenerate convex polygon that is described as an array of vertices in clockwise or counterclockwise order. In the next lines there are descriptions of craters. Description of one crater starts with an integer M_i on a line by itself number of vertices in the polygon (3 <= M_i <= 1000). Next M_i lines contain two integers X_{ij} and Y_{ij} each is the coordinates of j_{th} point in the i_{th} crater. In the last line of input there are four integers coordinates of points A and B. All coordinates are integers not greater than 10^4 by absolute value.

Output Specification

Print only one number with at least six digits after decimal point the minimal time that satellite will observe the moonpug while it is travelling from point A to point B.

Sample Input

```
1
1
3
0 0
100 0
50 -50
0 1 100 1
```

Sample Output

```
2.000000
```

[F] Spaceship Connections

Program: connections.(c|cpp|java)
Input: connections.in
Balloon Color: Gold

There are N Mooncats' spaceships in open space near the Moon. Ships' commanders) of course, all of them are mooncats, but it is not important in our problem) want to discuss the battle with moon rabbits which will happen soon. For this purpose, they need to establish some wireless connections between ships. But there are some limitations on them. First of all, spaceships all belong to one plane in 3D space. More than that, if we could look at this plane with our sight coinciding with plane's normal vector, we could see that Mooncats' ships form a regular polygon with N vertices. So, there are restrictions on connections. First restriction is that no ship can be connected to more than one ship at a time. Second restriction is that if we draw line segments between pairs of ships that are connected; no two segments could intersect (in other case, both connections will be broken due to very high noise).

Mooncats' admiral wants you to implement a system which will check if a connection between ships is possible. A connection is possible if after establishing it, no restrictions will be violated. And more than that, admiral wants you to develop a real-time system which will be able to track connections and disconnections between ships.

Input Specification

Input file contains on the first line the number of test cases T, then follow T cases each contains two integer numbers N and M number of Mooncats' spaceships and number of operations to process (1 <= N, M <= 100,000) In the following M lines there are operations. Each operation has form "CONNECT i j" which means that commanders of ships i and j want to establish a connection between their ships, or "DISCONNECT i j" which means that commanders of ships i and j have finished negotiations and want to break the connection. Numbers i and j are different.

Output Specification

For each operation, output one character '+' if the operation was successful or '-' if the operation failed. Operation of CONNECT type is considered successful if there is no connection between the specified ships and it is possible to establish this connection without violating the restrictions, otherwise it is unsuccessful (in this case, the new connection is not established). Operation of type DISCONNECT is successful if at the moment of its execution there was a connection between specified ships, otherwise it is unsuccessful. All characters should be on a single line.

Sample Input

```
1
8 4
CONNECT 1 3
CONNECT 2 4
DISCONNECT 1 3
CONNECT 2 4
DISCONNECT 2 4
CONNECT 1 2
CONNECT 3 4
DISCONNECT 1 3
```

Sample Output

```
+-++++-
```

[G] The Hangover

Program: hangover.(c|cpp|java)
Input: hangover.in
Balloon Color: Pink

Four friends Doug, Phil, Stu and Alan are traveling to Las Vegas, in a silver vintage Mercedes for the trip, for a bachelor party to their best friend Doug. They get a villa at Caesars Palace hotel and casino, then sneak onto the roof and make toasts to a night they will never forget. The next morning, the three groomsmen wake up in the suite with no memory of the previous night, and that is because Alan put Ruphylin in toasts instead of Ecstasy, as the drug dealer cheated on him and gave him Ruphylin instead of Ecstacy. Poor Alan thought that way he is helping to memorize that night and make too much fun, but he didn't actually know what is waiting for them next. Soon they are realizing that Doug is missing. The suite is in severe disorder, a tiger is in the bathroom, a baby is in the closet, Stu is missing a tooth, one of the suite's mattresses is impaled on a statue outside, Phil is wearing a hospital bracelet, and a valet brings them a police cruiser instead of the Mercedes.

Away from all this misery Stu has a very big problem, which he found out that he expended all the money in his bank account in playing cards. Stu girlfriend Melissa is too sneaky and checking all his bank accounts. He wants to put the money back to his bank account before she finds out what he had done. Lucky Stu played on very simple card game tables, when he starts playing, he has to pay a fixed amount M to the card dealer, then if he wins he got the same amount of money he already has, and finally he pays again another M amount to the card dealer (so if he has got initially $15 and M = $10, Stu is going to pay M=$10 before he starts, so he has got now $5. After winning he will got the same amount he already has, so he will have $5+$5=$10. Finally he has to pay another M=$10 to the card dealer summing his total money to ZERO). Stu played that game N times, He was so lucky again, it seems that drugged people are so lucky to win these days, as he won all the card games he got into, but he wasn't so lucky also because he lost all his money playing this game. Now Stu has ZERO money in his bank account. Given the amount M and the number of times he played N, Help poor Stu to know how much did he expended that night, before Melissa check his bank account records.

Input Specification

The first line of input contains an integer T that represents the number of test cases. T lines follow each line consists of two integers, the first integer (0 <= M <= 100) is the amount paid to the dealer when start playing and leaving table, and the second integer (0 <= N <= 100) is the number of times Stu played this card game.

Output Specification

For each test case print one line contains the amount of money Stu got initially, in money format $X.xx, before he got into all this misery. Stu doesn't care about values less than 1 cent.

Sample Input

```
2
8 2
72 3
```

Sample Output

```
$18.00
$189.00
```

[H] Star Ship

Program: starship.(c|cpp|java)
Input: starship.in
Balloon Color: Orange

Moon rabbits are planning massive colonization of Omega Prime. For that purpose, a great starship was built. But this starship needs fuel to operate. Thus moon rabbits need ***F*** barrels of fuel. There are ***N*** fuel providing companies on the Moon. Each company has f_i barrels of fuel in their stocks. The i_{th} company provides fuel at cost c_i money units per barrel ($1 <= i <= N$). Now, colonization government needs to minimize the total cost of all fuel. You are to help them with their problem and get rid of the evil moon rabbits once and forever: from Omega Prime, they will not disturb the Earth.

Input Specification

The first line contains ***T*** - the number of test cases. Then ***T*** test cases follows. To minimize the risk of enemy intrusion in government plans, the moon rabbits sent us encrypted data. First line of each test case contains two integer numbers ***N*** and ***F*** --- number of fuel providing companies and total amount of fuel needed for colonization ($0 <= N <= 10^7$, $0 <= F <= 10^18$). Second line contains four integer numbers f_1, c_1, *a* and *b*. To decrypt the message, you have to use formulas

$F_i = (F_{i-1} * a + b) \bmod 2^7$ and
$C_i = (c_{i-1} * a + b) \bmod 2^{31}$

for all $1 < i <= N$. *a* and *b* fit in signed 32-bit integer number.

Output Specification

Output only one integer for each test case --- the total amount of money units needed to buy the necessary amount of fuel. If the total amount of fuel available on the Moon is smaller than ***F***, output the total cost of all fuel on the Moon.

Sample Input

```
1
5 15
2 1 1 2
```

Sample Output

```
65
```

[I] Function

Program: function.(c|cpp|java)
Input: function.in
Balloon Color: Red

Moon rabbits reproduce really fast. Not as fast as Fibonacci rabbits, but still very quickly. Mooncat scientists discovered that the quantity of moon rabbits changes according to the following law: $x_{t+1} = f(x_t)$, where x_t is the quantity of moon rabbits at moment t. Surely, if we could know function f exactly, we could compute the quantity of moon rabbits at any moment with any initial conditions. Unfortunately, mooncat scientists don't know the function itself, but they were able to state some important properties of f. First of all, we will consider f only on positive integer values of argument (there could not be zero of negative quantity of moon rabbits). Function f is strictly increasing, thus $f(x+1) > f(x)$ for every x. Of course, $f(x)$ is integer for every x (have you ever seen 2.5 moon rabbits or pi moon rabbits?). And more than that --- after countless experiments, mooncat scientists were able to discover that $f(f(x)) = 3x$. Scientists are sure that it is possible to recover function f using these properties. Help them to do it.

Input Specification

The input file contains ***T*** test cases. The first line of the input file contains the integer ***T*** (1 <= ***T*** <= 1000). Each of the following ***T*** lines describes one test case and contains one integer ***N*** (1 <= ***N*** <= 10^18).

Output Specification

The output file should consist of ***T*** lines. Each line should contain the value $f(N)$ for the respective test.

Sample Input

```
1
7
```

Sample Output

```
12
```

[J] Master Spark

Program: masterspark.(c|cpp|java)
Input: masterspark.in
Balloon Color: Blue

This was a decisive battle for future of the Earth. Invasion of the moon rabbits was inevitable and United Command decided to use the ultimate weapon called "Master Spark". It is a powerful all-annihilating beam. The destructive beam is very big, so one can assume it has a shape of a cylinder of radius ***r*** (infinite in both front and back directions). Master Spark went through the moon annihilating all matter on its way. As you know, rabbits (especially moon ones) reproduce very fast. They live on the moon surface and speed of their reproduction depends on residential area (all moon surfaces). So United Command needs to know the area of annihilated moon surface to estimate the number of enemies. Now this is your turn to save mankind. Calculate the destroyed area of the moon. Remember, "Master Spark" is a bidirectional infinite all-annihilating beam.

Input Specification

Input file contains on the first line the number of test cases T, then follow T cases each contains four numbers --- radius of the moon and coordinates of its center at the time of the ``Master Spark" shot. Next line contains four numbers --- radius of the destructive beam and coordinates of some point on axis of the cylinder. ``Master Spark" was fired from point (0, 0, 0). All numbers are integers and do not exceed 10^3 by absolute value.

Output Specification

For each test case output the destroyed area of the moon surface. The answer should be accurate to at least four decimal digits.

Sample Input

```
2
10 100 0 0
5 1 0 0
10 100 0 0
50 0 1 0
```

Sample Output

```
168.357443
0.0000
```