7th BME International 24-hour Programming Contest

Qualifying Round Problem Set

3rd March, 2007

http://www.challenge24.org

The main sponsor of the contest is Fornax Co.

Diamond grade sponsors:

Morgan Stanley

MONTANA

mimox

Golden grade sponsors:

AdviseSoft
információtechnológiai kft.

Professional sponsors:

IEEE

HÍRKÖZLÉSI ÉS INFORMATIKAI TUDOMÁNYOS EGYESÜLET

Introduction

Welcome to the Challenge24 2007 Electronic Contest!

This year the Electronic Contest consists of 7 algorithmic problems. You will find 10 input files for each of the 7 problems enclosed in the zip file of the problem set. You can use any programming language or environment to generate the correct output files for these inputs. Once you are done, you can upload your output files through the Electronic Contest website after logging in with your team id and password.

Be quick about uploading the output files, because the scores awarded for every output file decrease with time. Uploading a correct output file at the start of the contest is worth 100 points. Uploading it just before the end of the contest is worth 80 points. During the contest its value decreases linearly with time. However you should also be careful with uploading solutions. Uploading an incorrect solution is worth -10 points.

Note that points are awarded per output file and not per problem. If your solution only works for 9 of the 10 input files, you will still be awarded points for those 9 output files. A single output file however is either correct or wrong – partially correct output files are not worth any points.

You have the opportunity to upload many output files together by packing them in an archive file. In this case the contained files are identified by their names (C5.out is the solution for the input file C5.in), and they are evaluated one by one the same way as if they had been uploaded separately. The only difference is that if the system encounters a wrong solution then the whole process stops to save you from getting much penalty.

Good luck and see you in the finals!

Problem A

String the Beads

You are given a set of beads and you want to make necklaces using up all of the beads. Each bead has a color and the diameter of the hole in each bead is an integer if measured in millimeters. You also have strings with different diameters. More precisely, you have exactly one string of k millimeter for each positive integer k (but of course you do not have a string of diameter 0). A bead can be strung onto a string if the size of its hole exactly matches the thickness of the string or if the string is exactly one millimeter thinner than the hole in the bead (otherwise the bead would be to loose). But of course you also have to pay attention to aesthetics. You are given a list of pairs of colors that look really awful together. You can not put two beads on the same string if their colors do not match.

Input:

The first line contains a single integer, the number of test cases. Then the first line of each test case consists of three integers: C, the number of possible colours, P the number of incompatible color pairs and B, the number of beads. The next B lines describe the beads. The size and the color of each bead is given. Then each of the next P lines consists of two integers in the range $\{1, 2, \ldots, C\}$, representing pairs of colors that do not match. From the next line the specification of the next case starts.

Output:

For each test case, you have to output a single line containing the word `POSSIBLE` or `IMPOSSIBLE`.

Example input:

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

Example output:

```
IMPOSSIBLE
POSSIBLE
```

Problem B

Strict Teachers

In a prestigious but modern school teachers pay very much attention to being perfectly fair with their students. So they invented a complicated examination system in which each student has to turn in a finite series of non-negative integer numbers in place of the traditional, but very subjective essays. Every teacher who evaluates the works of the students has a so called evaluation function (an integer function with two arguments represented by f), a non-negative integer A and a positive integer B.

When a student hands in an exam (a sequence of integer numbers $y_1, \ldots, y_N$) the teacher evaluates it by applying the `reduce` operation to the sequence with the starting value A and operator f (the evaluation function). The result of the operation is the student's score. The student has passed the exam if this score is divisible by B.

The `reduce` operation is widely used in functional languages for calculating sums and products over sequences. In the case of evaluating the exams its result can be found by calculating x_N where

$$\begin{aligned} x_0 &= A \\ x_i &= f(x_{i-1}, y_i) \end{aligned}$$

Note that students can also turn in an empty series ($N = 0$).

Because of a financial problem the director of the school decides to fire some of his teachers, but he wants to keep as much diversity in the school as possible. He holds that the essence of education is examination, so two teachers are different, if there is at least one such exam – that is a series of numbers – which they evaluate differently (i.e. one of them accepts it while the other rejects it). Your task is to find the equivalent teachers.

Input:

The first line of the input contains one integer, the number of teachers. Then two lines per teacher follows. The first line contains the numbers A and B. The second line contains a formula specifying the evaluation function. Each formula may contain parentheses, non-negative integer numbers, operators $+$, $-$ and $*$ and the symbols x and y representing the first and the second argument respectively. The operator $*$ has a higher precedence than operators $+$ and $-$, but operators $+$ and $-$ have the same precedence. Their meaning is as usual.

Output:

Output exactly one line for each equivalence class of teachers. One class is specified by a space delimited list of numbers ordered incrementally. Each number identifies a teacher with his or her index in the input list. The first teacher is represented as 1. The classes have to be in increasing order of their first elements.

Example input:

```
4
0 10
(x+5)*(y+3)
1 10
x*y+5*y+3*x+15
0 10
x*x-y*y
0 10
x*y+5*y+3*x+15
```

Example output:

```
1 4
2
3
```

Problem C

Replacing the Talking Pets

Since a certain crustacean physician has consumed the last of the fish which were usually stuck in our ears and eased the communication of different intergalactic species by translating their words, a tender was published to create a machine translation system. A transtemporal outsourcing company turned out to be the lowest bidder (they were able to make such a low quote because they knew centuries such as the 21^{st} where enthusiastic programmers worked on difficult problems for fun).

Your task is to create a program for generating dictionaries. The input for the program consists of a dictionary mapping the words of language A to language C, a dictionary mapping the words of language B to language C and a thesaurus that lists groups of synonymous word in language C. The program has to produce a dictionary for translating from language A to language B.

To be able to solve the problem, some things should be known about the languages spoken in the future. After the invention of time travel evolution reversed somehow and the human brain deteriorated severely. Human languages also became quite simple. There is a universal set of concepts. Each word in each language corresponds to a set of concepts that the word can refer to. We call this set the meaning set of the word (M_w). Two words are synonymous if their meaning sets are intersecting. The entries in a thesaurus simply corresponds to the concepts in the given language. The entry of the concept c_i contains all the words that have the i^{th} concept in their meaning sets (that is $\{w : c_i \in M_w\}$). You can assume that there are no words with empty concept sets. In a good dictionary from language X to language Y the entry for the word w (of language X) contains all the words in language Y that have an intersecting meaning set with w.

Input:

The first line contains three integers: K, L, and M. K is the number of entries in the $A \to C$ dictionary, L is the number of entries in the $B \to C$ dictionary, and M is the number of entries in the thesaurus for language C. The next K lines describe the $A \to C$ dictionary: the first word of every line is the A language word, and the remaining words are the possible translations of this word in language C. The following L lines describe the $B \to C$ dictionary in the same format. The last M lines contain the thesaurus for language C: every line contains a set of synonymous words that denote the same concept.

Output:

The output should contain one line for every word in language A, sorted in alphabetic order. The first word should be the word in language A. If the set of possible translations in language B can be determined unambiguously, then the remaining words should be the list of these possible translations in alphabetic order. Otherwise you have to output `AMBIGUOUS` as the second word in the line.

Example input:

```
2 1 2
apple alma
pear alma korte
apfel alma
alma
alma korte
```

Example output:

```
apple apfel
pear AMBIGUOUS
```

Problem D

Race Track

Fuel is of central importance in all motor sports, as it is a serious embarrassment for a team if they run out of fuel during a race. Precise planning is crucial and your help is needed for it. There is a circular race track of k kilometers, where k is a positive integer. There is a fuel station after each kilometer on the track. So for example in a 5 km long track there are 5 stations. Each station has a certain amount of fuel, and when the car reaches a station, all the station's fuel is loaded into the cars tank. (The tank can be considered to have an infinite capacity.) The car has an empty tank at the beginning, so it has to start from a fuel station.

The starting fuel station can be chosen arbitrarily. Some choices can lead to the fuel depleting between stations while others can enable the car to complete a full lap. The team has an expert for picking the best starting location, but he requests that a program be written for determining if it is even possible to pick a station that would enable the completion of a full lap. He does not enjoy getting up in vain.

The amount of fuel in each station is given by the following recursion (we denote the amount of fuel in station k with F_k):

$$\begin{aligned} F_1 &= A \\ F_i &= (F_{i-1}B + C) \bmod M \end{aligned}$$

Input:

Each test case is specified by one line, containing 6 integers N, F, A, B, C, M. N is the length of the track in kilometers (that is the number of the stations), F is the amount of fuel needed to drive one kilometer, A, B, C and M are the parameters in the recursion above.

Output:

For each test case, you have to output a single line containing the word `POSSIBLE` (if there is a station that enables completing a full lap) or `IMPOSSIBLE` (if there is no such station).

Example input:

```
3 2 2 0 2 10
1 3 2 0 2 10
```

Example output:

```
POSSIBLE
IMPOSSIBLE
```

Problem E

Meteor Defense

You are the mayor of a small planet (Procyon II). Being the economic genius that you are, your first action as mayor was selling the planet's atmosphere to an alien race of traveling merchants. For the price of the atmosphere you have bought a luxurious villa with a large pool, two sports cars and an ivory statue of yourself. There was nothing luxurious for sale for the sum that was left over after fulfilling your dreams so you went ahead and bought a tunnel digging machine.

As it turned out, your last purchase was the most useful in the end. Without atmosphere the planet is now vulnerable to meteors. You have had to move the cities underground and connected them with a system of tunnels. The tunnel digging machine has a simple and intuitive user interface – you just had to click any two cities and it would dig a tunnel between them in a straight line. Its operation however is very noisy and you are worried that this might disappoint some voters. So you have kept the total length of the tunnel network to a minimum while at the same time connecting all the cities (your sports cars would not have much of a point if you could not drive to any city).

Recently however scientists have reported that a larger meteor is now on its way towards the planet. It could very well destroy a city if it were to fall on one. It would even collapse all the tunnels leading to this city. You only have a limited number of meteor defense systems to deploy and not enough to protect all of the cities. Your task is to decide which cities it would be most economic to protect.

All cities have equal chances of being hit by the incoming meteor. Your assistant has prepared you a list of cities with the number of voters supporting you in each city. Once the giant meteor hits a city and its tunnels collapse the tunnel system might be disconnected. It will have to be restored and all this digging will lose you some more supporters. The coordinates of every city on the list were transformed to a scale in which the digging of the tunnel between two cities at unit distance would lose you 1 supporter. The list of cities to be protected would have to be the list of the cities that would lose you the highest number of supporters if they happened to be struck by the meteor.

Input:

The first line of the input file contains two numbers separated by a space character: N (the number of cities) and M (the number of meteor defense systems)

The next N lines each contain three number separated by space characters: X_i and Y_i (the coordinates of the i^{th} city in supporter-space) and S_i (the number of your supporters in the i^{th} city)

Output:

The output file consists of M lines each containing two numbers: X_i and Y_i (a city to be protected as identified by its coordinates). The order of the lines is arbitrary.

Example input:

```
5 2
0 1000 1000
1000 100 0
1000 1000 0
1000 2100 0
2200 1000 0
```

Example output:

```
1000 1000
0 1000
```

Problem F

Some More Meteor Defense

Ten years have passed and some of the mistakes of the late mayor from Problem E have been slowly repaired. It turned out that with the constant noises from the tunnel digging machine the entertainment sector plummeted and people actually started doing their jobs instead of watching movies and playing games. You are one such born-again engineer and your task today is estimating the effectiveness of meteor defense systems installed above the tunnels.

The meteor defense system consists of a shield that covers a number of segments of the tunnel and can be moved along the tunnel. Unfortunately the movement of this heavy instrument is really slow, as it can only be moved the distance of one segment per day.

Astronomers have predicted the trajectories of incoming meteors for the next century. Your task is to calculate the maximum number of meteors that can be caught by the shield during this century.

Input:

The first line of the input consists of three integers, L, P and N, the number of segments of the tunnel, the number of tunnel segments the shield can cover and the number of the meteors in the century. The shield is always positioned in the first P segments of the tunnel $(0, 1, \ldots, P-1)$ on the day 0. In the next N lines, the parameters of the meteors are given in chronological order. Each meteor is described by two integers T and S, where T is the day of the arrival (measured in the number of days elapsed since the installation of the shield) and S is the index of the segment where the meteor would hit the tunnel. (S is an integer from 0 to $L-1$.) Note that more than one meteor can arrive on the same day!

Output:

The output is a single integer, the total number of meteors that can be caught if the shield is moved optimally.

Example input:

```
10 3 4
0 3
5 5
6 0
7 0
```

Example output:

```
2
```

Problem G

Interplanetary Etymology

Settlers of different planets have distinct words for the same notions because of the millennia spent in isolation. For example they call a "Big Mac" "Bug Mac" on Betelgeuse (owing to the gourmet insects indigenous to the planet). Recently however the appearance of affordable FTL communication devices enabled etymologists to collect and analyse large corpora of planetary variations.

The need arose for a tool to help map the spread and development of word variations. Usually etymologists have an idea about how the settlers moved from planet to planet, and this spread of human civilization can be represented by a binary tree. The leaves represent the current colonies, while other nodes are past versions of these colonies or abandoned colonies or colonization hiveships. The input for the tool consists of this binary tree as assembled by the researchers and the different variations of a word for every current colony.

The distance in etymological terms between two words is the number of letters that are *not* the same in the two words (e.g. $D(\text{"Big Mac"},\text{"Bug Mac"}) = 1$). It is only defined for words of the same length, and the input is preprocessed so that all variations are the same length. Each node (whatever it represents) has a version of the word in analysis associated with it, but it is only known for the leaves of the tree. This means that every edge in the tree can be associated with the distance between the words of the two nodes connected by it. Depending on the word variations in the non-leaf nodes the sum of the distances in the tree can be different. The output of the tool has to be the minimal total distance. This is useful for the etymologists for evaluating migration models and the actual word variations will only have to be enumerated by a later version.

Input:

A leaf node is represented with the word variation used on the colony between quotation marks: `"Bug Mac"`

A non-leaf node is represented with the two child nodes between parenthesis and separated by a comma: `("Bug Mac","Big Mac")`

Output:

The output is simply a number (in decimal representation).

Example input:

```
(("alma","alba"),"alap")
```

Example output:

```
3
```