International Collegiate Programming Contest
Quarter-Final

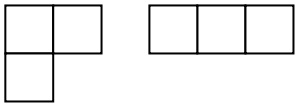
May 19, 2011

Problem A

Triomino

(1 sec)

Input: a.dat
Output: a.ans



You should tile rectangular grid M×N with triomino figures which are shown at the picture. The figures can be rotated through angles which are multiples of 90º, but it's prohibited to overlay figures at the same place. Of course, not all M×N rectangles can be tiled completely, but it's possible to minimize number of non-tiled cells.

Your task is to write a program which finds any allowed tile of M×N rectangle with triomino figures, which have minimal possible number of non-tiled cells.

Program should read from input two integers M and N — number of rows and columns of the rectangle (1≤M≤100, 1≤N≤100).

Program should output M rows with N numbers in each row. Each number indicates the number of the triomino figure, which covers this cell. If current cell is not covered by any figure, 0 number should be printed. Figures should be denoted with different numbers 1, 2, 3, ..., K (where K is total number of used figures).

Usually, there are many different correct answers for the same input data. Your program should write any one of them.

Input	Output
4 11	0 1 1 1 3 3 4 4 4 5 5 14 2 2 2 3 7 7 6 6 6 5 14 12 13 13 13 7 8 8 8 0 9 14 12 12 11 11 11 10 10 10 9 9

0	1	1	1	3	3	4	4	4	5	5
14	2	2	2	3	7	7	6	6	6	5
14	12	13	13	13	7	8	8	8	0	9
14	12	12	11	11	11	10	10	10	9	9

International Collegiate Programming Contest
Quarter-Final

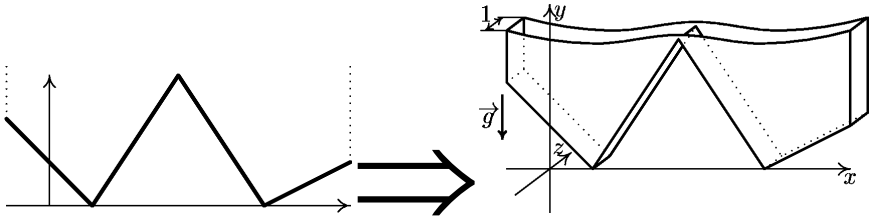
May 19, 2011

Problem B
Au reservoir

(1 sec)

Input: b.dat
Output: b.ans

Consider a polygonal chain. Its coordinates (x_1, y_1), (x_2, y_2), (x_3, y_3), …, (x_N, y_N) satisfy inequalities $x_1 < x_2 < x_3 < \ldots < x_N$ and $y_i \neq y_{i+1}$ for all i. Let's draw rays upward from the leftmost (x_1, y_1) and the rightmost (x_N, y_N) vertices. Then let's transform the plane figure into three-dimensional body with constant thickness of 1.



A reservoir was made according to these rules. Its front and back sides are plain, vertical, parallel each to other, and are at distance 1 each from other. The left and right sides (obtained from the verticals rays) are also plain, vertical and parallel each to other. The bottom of the reservoir is obtained from the initial polygonal chain. The reservoir is mounted so that irrespective of the bottom shape and of filling level it will never turn over.

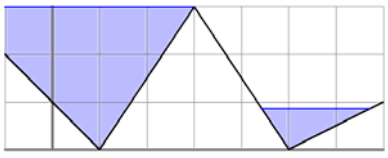
V cubic units of water are carefully filled into the reservoir, from its left side.

Your task is to write a program, calculating the square of resulting water surface.

Program should read from input the number of vertices in polygon chain N ($2 \le N \le 123\,456$), than N pairs of integers $x_1\ y_1\ x_2\ y_2\ \ldots\ x_N\ y_N$, meaning the coordinates of the vertices, and, at last, the water volume V. All coordinates are integers in range from -10^6 to 10^6; volume is an integer in range $0 \le V \le 10^{12}$.

Program should output exactly one floating-point number — the square of final water surface. Absolute accuracy of the answer should not be worse than 10^{-3}.

Input	Output
5 -1 2 1 0 3 3 5 0 7 1 8	6.309401



International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem C

The Flight Way

(1 sec)

Input: c.dat
Output: c.ans

Club of obsolete vehicles decided to celebrate the May 5th, 5555 with great flights of copies of ancient air planes. Club members have built in various places on the Earth surface N ($2 \le N \le 1000$) aerodromes and prepared planes. But suddenly an amendment to the ecological taxes law was adopted, so the club must hold the celebrations in short format. Two aerodromes were selected, and the only one plane will fly between them using the shortest possible way! The maximum distance it can fly without landings is R km, so if necessary it can use intermediate aerodromes.

Help the club to find the shortest route, considering the Earth surface accurate sphere with equator length exactly 40 000 km and ignoring the influence of flying height at the distance.

The first row of the input contains R – the maximum direct fly distance (in kilometers, $10 \le R \le 50\,000$). The second row contains the number of aerodromes N. The following N rows contain geographic coordinates of the aerodromes. The format of coordinates is: first the latitude (uppercase letter N or S — North or South, then degrees, minutes and seconds), then longitude (similarly, but using the letter E or W — East or West), all numbers and letters are separated by single space. Degrees and minutes are integers, seconds can have decimal fractions. At the last line of the input, two integers denotes the aerodromes selected as start and finish (1-based, i. e. numeration starts from 1).

Your program should output three lines. The 1st line should contain one floating-point value — the minimal possible length of route from start to finish, in kilometers, with accuracy up to meters. The 2nd line should contain exactly one integer K — the number of intermediate aerodromes in the route. The 3rd line should contain list of these intermediate aerodromes, in order from start to finish. If there are different correct answers, your program should find any one of them. If plane can reach finish from start without landings, the 2nd line should contain 0 and the 3rd line should be empty. If plane cannot reach finish from start, the 1st line must contain number 123456789.000, the 2nd — number 0 and the 3rd should be empty.

Input	Output
7127 5 N 90 0 0 E 0 0 0 N 0 0 0 W 15 0 0 S 90 0 0 E 0 0 0 N 30 0 0 E 175 0 0 S 30 0 0 W 175 0 0 1 3	20083.446 2 4 5

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem D

Probablisctic OR

(1 sec)

Input: d.dat
Output: d.ans

Everyone knows OR operation. Let us define new operation which we will call Probablistic OR. We will denote this operation as #. For given real number p ($0 \le p \le 1$) and two bits a and b:

- if a = 1 and b = 1, then #(a, b) = 1;
- if a = 0 and b = 0, then #(a, b) = 0;
- else #(a, b) = 0 with probability p, #(a, b) = 1 with probability 1-p.

Now for two given non-negative integers x and y we can define bitwise Probablistic OR operation. The result of this operation is a number received by performing # operation for each pair of bits of x and y in same positions. For example, for p= 0.5, x = 2, and y = 4, we will get 0, 2, 4 or 6 each with probability 0.25.

You will be given a list of non-negative integers. You have to implement a program which will calculate the expected value of the result of performing bitwise probablistic OR operation on all these numbers given some p. The numbers will be taken from left to right.

Input file starts with real number p ($0 \le p \le 1$) with exactly two digits after the decimal point. Integer n follows ($1 \le n \le 100$). Next line contains n numbers a_i in the order they are taking part in the operation ($0 \le a_i \le 10^9$).

Output the expected value of performing Probablistic OR operation on the given numbers for given p. Print the result with two digits after the decimal point.

Input	Output
0.25 4 1 2 3 4	5.11

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem E

Party

(1 sec)

Input: e.dat
Output: e.ans

Kate is preparing a party. She have bought a very strange garland for it. The garland is a closed chain of bulbs. Each bulb can be in one of the following states: N - don't glow, R - glow red, G - glow green, B - glow blue. Each second the state of each bulb changes according to the following table:

	N	R	G	B
N	N	R	G	B
R	R	N	B	G
G	G	B	N	R
B	B	G	R	N

where row is chosen by the current state of the bulb and column is chosen by the state of the bulb on the right. The value at the intersection of the chosen row and column gives the new state of the bulb. For example, if the bulb glows red (R) and the bulb on its right glows green (G) then in the next second the bulb will glow blue (B). And if the bulb and its right neighbour both glow blue then the bulb won't glow at all in the next second. Also all the bulbs change their states simultaneously. Such behaviour should (theoretically) lead to constant twinkling of the garland. Unfortunately it turns out that sometimes eventually the garland goes into such a state that all bulb don't glow. So the garland stops twinkling. Kate is rather worried that this can spoil the party. She is going to set the initial states of each bulb as she like. Help her determine for how long the garland will twinkle starting from this initial state.

The input file consists of a single string containing characters 'N', 'R', 'G' and 'B', which describes the initial state of the garland. Each character defines the initial state of some bulb. The bulbs are listed from left to right. The first bulb is located on the right of the last one. The length of the string will be no more than 1234567 characters.

Output the number of seconds during which the garland will twinkle. If the garland won't stop twinkling (at least until the power is turned off) print "Party!" (quotes for clarity).

Input	Output
RGBG	4

The garland will change the state in such a way:

RGBG
BRRB
GNGN
GGGG
NNNN

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem F

Strong-Willed Victory

(1 sec)

Input: f.dat
Output: f.ans

Yuri loves football. He is very happy when his favorite team wins. He is even happier when this is a strong-willed victory. A strong-willed victory is an outcome when the team wins, but was losing at least at one moment during the match. Recently Yuri learned the result of the last match of his favorite team: it won. He doesn't know how the score developed and now he is interested what is the probability that this was a strong-willed victory. Help Yuri calculate this. We will assume that any scenario of the match is equally probable. The scenario of the match is determined by the sequence in which the teams score goals.

The input file consists of two integers: a - the number of goals Yuri's favorite team scored and b - the number of goals scored by the rival team. ($0 \le b < a \le 1\,000\,000$)

Print the probability that the last match of Yuri's favorite team was a strong-willed victory with six digits after the dot.

Input	Output
2 1	0.333333

One of the three possible match scenarios is a strong-willed victory: when the rival team scores first. For the other two scenarios Yuri's favorite team isn't losing for a single moment.

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem G

Segment Sum

(6 sec)

Input: g.dat
Output: g.ans

A segment of a sequence $a_1, a_2, \ldots, a_N$ is a sequence $a_i, a_{i+1}, \ldots, a_j$ for some $1 \le i \le j \le N$. Given an integer sequence consider a set of all its segments, ordered by the sum of elements. Your task is to find the sum of elements of K-th segment in this order (0-based). In this problem two segments are considered different, if one of their ends does not coincide. So actually it can be a multiset, for example for sequence 2, 2 all its different segments are {2}, {2}, {2,2}. Therefore any sequence of length N will have exactly $N(N+1)/2$ segments. For given example, their sums will be 2, 2, 4 in order.

First line of input contains two integer numbers N and K – length of the sequence and 0-based number of interesting segment ($1 \le N \le 10^5$, $0 \le K < N(N+1)/2$). Second line contains N integers $a_1, a_2, \ldots, a_N$ – the elements of the sequence ($-10^9 \le a_i \le 10^9$).

Input	Output
3 0 1 4 -3	-3
3 1 1 4 -3	1
3 2 1 4 -3	1
3 3 1 4 -3	2

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem H

Sum

(1 sec)

Input: h.dat
Output: h.ans

Roman's parents gave him an undirected connected weighted graph with N vertexes and N-1 edges. Roman wants to know the sum of all the paths' lengths in this graph. Let's consider path's length as sum of all the edges that lay on this path. Roman said that the path from U to V is the same as from V to U, so he doesn't distinguish them.

The first line of the input file contains the single integer number N($2 \le N \le 10^5$) – the number of vertexes in graph. It is followed by N-1 lines containing the description of the edges. Every line of description consists of three integer numbers: vertices connected by the edge (labeled [1…N]) and edge's weight.

Output a single integer number – the sum of all the paths' lengths in the given graph modulo 10^9.

Input	Output
3 1 2 1 1 3 3	8

An explanation to the example: All the paths are 1->2, 1->3, 2->1->3 and their lengths' sum is 1+3+4=8.

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem I

Distance Between Lletters

(1 sec)

Input: i.dat
Output: i.ans

You are given a string s containing lower-case letters of Latin alphabet. We define a distance between two symbols s_i, s_j of this string as difference between their positions, i.e. $|j-i|$. Let us consider all pairs of equal symbols located in different positions, and calculate distances between symbols of each pair.

Write the program for determining minimum of all such distances.

The single line of input file contains a string s. The length of this string does not exceed 5×10^6.

Output one number, minimal distance between equal symbols. If there are no pairs of equal symbols in the string, output the number 0.

Input	Output
abcacba	2

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem J

String potential

(6 sec)

Input: j.dat
Output: j.ans

You are given a string s containing k first lower-case letters of Latin alphabet. As in the problem I, we define a distance between two symbols s_i, s_j of this string as difference between their positions, i.e. $|j-i|$. Let us define the potential of string in the following way. Let the function $f(c_1,c_2)$ which for each pair of letters gives some weight is known. This function is symmetric with respect to arguments, i.e. $f(c_1,c_2) = f(c_2,c_1)$. The potential between the pair of symbols s_i, s_j will be defined as a product of a pair weight and a distance between of its symbols. The string potential will be calculated by summing up of potential of all pairs of symbols.

Write the program that calculates the potential of given string s.

The first line of the input file contains integer k ($1 \le k \le 26$), the number of first Latin letters allowing in string. The second line contains a string s. Its length does not exceed 10^6. In following k lines the weight function is given. i-th line consists of i numbers, j-th number in i-th line defines the weight for i-th and j-th Latin letters. All weights do not exceed 10^6 by absolute value.

Output one number – the potential of a string s.

Input	Output
3 abcacba 1 0 1 1 0 1	32

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem K

Fruits

(1 sec)

Input: k.dat
Output: k.ans

Petro sells very exotic fruits at the marked. Petro owns *N* such fruits, each of some positive weight. Petro doesn't know the weights. Besides that, Petro owns a balance scale. He can place in left and right pans some fruits. The scale shows whether total weight of fruits in each pan is equal, or one pan outweighed other pan. After several such weightings, the Petro wants to predict the result of the next weighting before performing it.

Write a program that will help Petro to know what the scales will show, when performing the next weighing.

The first line of the input file contains two integers *N* and *M* ($2 \le N \le 20$, $0 \le M \le 50$). Each of following *M* lines defines one of implemented weighting and has following format: firstly, numbers of fruits in left pan are given, then one of symbols "<", "=", ">", determining the weighting result, and lastly, numbers of fruits in right pan are given. In the last line in similar way forthcoming weighting is given, but its result is unknown and indicated by the symbol "?". Within one weighting, any fruit can be located either left pan, or right pan, or is not involved in weighing.

In the output file you should write all possible results of forthcoming weighing. In the case of several possible results you should keep the order "<", "=", ">". If input data is contradictory, you should write "Impossible".

Input	Output
4 2 1 2 > 3 4 1 = 3 4 ? 2	<
3 3 1 2 > 3 1 = 2 3 2 > 1 1 ? 3	<=>
3 3 1 = 2 2 = 3 3 > 1 1 2 ? 3	Impossible

International Collegiate Programming Contest
Quarter-Final

May 19, 2011

Problem L

Football club

(1 sec)

Input: l.dat
Output: l.ans

Modern football clubs have a lot of players in their staff, but only 11 players can participate in a starting lineup in a particular match. One of them (goalkeeper) take the place near own team goal and is permitted to touch the ball with hands or arms within own penalty area. Other players have also some jobs and positions in the field. Defenders are mainly located on own side of the field, their job is to stop opposing attacking players from scoring. Midfielders are acted in the middle of field, helping to defenders or forwards, depending on game situation. And finally, forwards are located primarily on opposite side of the field and are responsible for scoring goals.

Indeed each player can play at any position, but his efficiency at one position will be greater than at other position. This particularly depends on the experience of playing at corresponding position, individual skills and other factors. The number of players at each position, except goalkeeper, can vary, but must be conformed to one of allowed formations.

You are a manager of the football club called ACM ILAN. There are N players in the club. For each player, his efficiency at each position is known. There are also K formations. You can choose and distribute players with respect to one of these formations.

Write the program determining most suitable formation and starting lineup providing maximal total efficiency.

The first line of the input file contains two integers N and K ($11 \le N \le 30000$, $1 \le K \le 10$). Each of following N lines consists of 4 integers – player efficiencies at position of goalkeeper, defender, midfielder and forward respectively. Efficiency is an integer between 0 and 100 inclusively. Each of last K lines contains 3 integers, determining allowed formation – the number of defenders, midfielders and forwards in starting lineup when choosing this formation (the sum of these numbers is equal 10).

Output one number, the maximal total efficiency of starting lineup.

Input	Output
12 3 90 10 10 10 20 50 40 30 20 90 20 70 30 60 20 60 30 70 20 20 20 80 50 70 80 20 20 20 10 20 40 80 20 30 80 30 10 20 90 60 10 40 40 90 10 50 20 80 4 4 2 4 3 3 3 4 3	850