Problem A

Lucky Controller

(0.5 sec)

Input: Standard input
Output: Standard output

Egor works as a controller in the bus. Each day he is given a pack of tickets which he then sells. Recently he has become interested about how many tickets in the pack are lucky. He thinks that the more tickets are lucky the luckier day he will have. Now he wants to find out how lucky for him the next day is going to be. Each ticket number consist of n digits. The ticket is considered to be lucky if the sum of the first n/2 digits equals to the sum of the last n/2 digits. Egor knows that the numbers in the pack that he will be given can start with equal probability from any number in the interval from a to b inclusively. The pack holds k tickets. The numbers of the tickets are consecutive. Help Egor to find an expected quantity of lucky tickets in the pack.

Input consists of a single line with three integers a, b и k ($0 \le a \le b < 10^{12}$, $1 \le k \le 100\,000$). Integers a and b consist of same amount of digits, and this amount equals to the amount of digits in the number of each ticket. They may start with zeroes. The amount of digits in a and b is always even.

Output the expected quantity of lucky tickets in the pack in the form of irreducible fraction. In case the result is an integer – no slash should appear in the output.

Input	Output
0123 4567 150	6519/635
10 10 20	2
4000 4999 11	103/125

Problem B
Gluttonous robot
(1 sec)

Input: Standard input
Output: Standard output

Secret laboratory of Fatown has developed a new gluttonous robot which moves on stripe consisting of n+1 cells. Cells are numbered from 0 to n. The robot is located at cell with number 0; each other cell contains several bukazoids which gluttonous robot regales oneself with. The robot can do m single jumps (to adjacent cell) and k double jumps (over one cell). Additionally, m+2k=n. All jumps are jumps forward. To feed gluttonous robot you need to write a program which finds sequence of jumps with highest number of bukazoids on a way.

The first line at the input contains 3 integers: n(1<=n<=100), m(0<=m<=100), k(0<=k<=100). The second line contains n integers – number of bukazoids (up to 100) in corresponding cells of the stripe.

The first line at the output should contain highest number of bukazoids found. The second line should contain m+k+1 integers – numbers of cells visited by the robot, starting from cell with number 0.

Input	Output
5 1 2 5 2 7 3 1	13 0 1 3 5

Problem C
Collisions

(3 sec)

Input: Standard input
Output: Standard output

Identical small balls are located on a straight line and can move along this line only. Each ball moves with a constant velocity, but velocities of different balls may be different. When two balls meet, a perfectly elastic collision occurs. It’s a common-known physical fact, that when two equal-mass physical bodies A and B collide perfectly elastically, they swap their velocities, i. e. new A’s velocity is old B’s one, and new B’s is old A’s.

Your task is to write a program to find the total number of collisions.

The first line at input contains the number of balls N ($3 \le N \le 200\,000$). Each of the following N lines contains 2 space-separated integers — the starting coordinate and the velocity of corresponding ball. All start coordinates are in range $-10^{11} < x_i < 10^{11}$, all velocities are in range $-10^8 < v_i < 10^8$. All start coordinates are different. It’s guaranteed that each collision involves exactly two balls (none involves three or more balls together).

Your program should output exactly one integer number in a single line – the total number of collisions (or 987654321987654321 if the number is infinite).

Input	Output
3 –5 3 0 –1 7 –2	3

Problem D

Pairs of dancers

(1 sec)

Input: Standard input
Output: Standard output

Alex missed the ballroom dance competition that he wanted to attend. So now he wants to know the pairs of dancers whose dancing he missed. He had several photos from the competition, so he chose one where all dancers are clearly visible and wrote down the coordinates of all **N** dancers (**N** is even).

Then Alex determined the pairs of dancers by the following algorithm: from not yet paired dancers he chooses two closest (to each other) dancers and assumes that they dance together as a pair. Should he find several pairs of dancers with the same minimum distance between dancers, he chooses lexicographically smallest pair (Alex enumerated dancers by integer numbers from 1 to **N**, dancers are ordered inside a pair, one with lower number goes first). You are asked to help Alex to determine pairs of dancers.

The first line of input contains even integer **N** (2 <= **N** <= 300). Each i-th line of the next **N** lines contains two integers — x and y coordinates of i-th dancer. All coordinates are less than 10^8 by absolute value.

You should output **N/2** lines. Each line must contain numbers of dancers in the corresponding pair. The first number in a line should be less than the second. Lines must be sorted in the lexicographically ascending order.

Input	Output
6 0 2 3 2 0 0 1 0 0 -2 2 -2	1 2 3 4 5 6
4 0 0 1 1 0 1 1 0	1 3 2 4

Problem E

St. Valentine Queue

(0.5 sec)

Input: Standard input
Output: Standard output

A queue of *limerents* (that is people in love) from all over the world to St. Valentine never gets empty. It is especially long on St. Valentine's Day. St. Valentine talks to each limerent during one time unit. Some limerents, due to long time in queue, are late to their trains and have to change their tickets. St. Valentine's Fund recovers ticket change fee and therefore aims at reduction of total recovery amount.

Let N be the quantity of limerents in the queue; i – a sequential number assigned to each limerent when entering the St. Valentine Administration; d_i – departure time of the i-th limerent's train (measured in time units), w_i – ticket change fee for the i-th limerent. It is guaranteed that a limerent is still not late visiting St Valentine at time d_i. Your task is to put limerents into the queue in the order that minimizes total recovery amount to be paid by St. Valentine's Fund.

Input contains the number of limerents N in the first line, followed by N lines containing d_i and w_i in the (i+1)-th line, separated with single space. All the numbers are positive integers and not greater than 30 000.

Output must contain the optimal queue as N lines, with i-th line containing number of limerent standing at i-th position. If several optimal orderings exist you need to output any of them.

Input	Output
7 3 40 2 60 6 10 1 30 4 70 4 50 4 20	2 1 5 6 3 4 7

Problem F

Obscene Words Filter

(1 sec)

Input: Standard input
Output: Standard output

Given a dictionary of obscene words S_1, S_2, ..., S_n and text T, find if text contains one of obscene words as subsequence. If it does, find smallest prefix of T that contains such subsequence.

First line of input contains one integer n - number of words in dictionary. Following n lines contain words from dictionary, one per line. Each word consists of ASCII characters with codes from 32 to 127, inclusive. Next line contains text T, consisting of the same set of characters. Total length of all words in dictionary doesn't exceed 100 KiB (100×2^{10} bytes). Total size of input file doesn't exceed 1MiB (2^{20} bytes).

Output **NO** if there is no obscene subsequence in the text. Otherwise output **YES** <X>, where X is the length of smallest prefix of T that contains some obscene subsequence.

Input	Output

`2`
`hello`
`world`
`abracadabra`
	`NO`

`2`
`hello`
`world`
`zzzheluuuulottt`
	`YES 12`

Problem G

Kuzma beaver game

(1 sec)

Input: Standard input
Output: Standard output

September 10^{th} is forester day. On this occasion Magic forest dwellers decided to make a party. Kuzma beaver is assigned to take care about intellectual games. Kuzma has invented at his opinion quite interesting game.

Rules of the game are simple. Master announces positive integer N. Player needs to calculate product of digits for each positive integer up to N and report the greatest product.

All answers should be known beforehand to allow interactive communication during a game. This causes problems since N could be relatively big (1<=N<=2 000 000 000). Kuzma is not good using computers; he asks you to write a program which having positive integer N will find correct answer.

Each line at input contains integer N

Corresponding line at output should contain greatest product for this N.

Input	Output
1 101090000 28994 4876 2789	1 43046721 10368 2268 1008

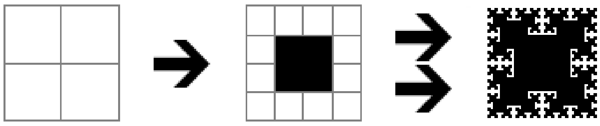
Problem H

Piece of cake

(1 sec)

Input: Standard input
Output: Standard output

Fyodor celebrates his birthday today. Before the guests come he decorates a cake with chocolate cream in a special way. At the beginning cake looks like a square divided into 4 equal white square parts. Fyodor calls fractalization the following sequence of actions. All the little squares of cake get united into groups of 2x2 so that there are no ungrouped fragments. After that each small square is divided into 4 equal squares so that group of 2x2 becomes a group of 4x4. As the last action 4 squares in the middle of each group are filled with chocolate. Fyodor does not stop at one fractalization and repeats it N times, even when he has to use a microscope. Illustration below shows the initial cake, first fractalization result, and the cake after the fifth fractalization:



Now Fyodor wants to cut pieces of cake with beautiful patterns for guests, but it is difficult to assess beauty of a piece looking at the whole cake. Fyodor wants a program that will quickly show the pattern of rectangular part of the cake.

Single line at the input contains five non-negative integers: N, R1, R2, C1, C2. N – the number of fractalization iterations (N < 20), R1 and R2 – first and last rows, C1 and C2 – first and last columns of the part. Following restrictions are also met: R1 <= R2, C1 <= C2; 0 <= R2 - R1, C2 - C1 < 100; 0 <= R1, R2, C1, C2 < 2^{N+1}

Output should contain R2 - R1 + 1 lines each containing C2 - C1 + 1 characters. Each symbol corresponds to a square and should be 1 in case it's filled with chocolate and 0 otherwise.

Input	Output

`1 0 3 0 3`	`0000`
	`0110`
	`0110`
	`0000`

`2 0 3 0 3`	`0000`
	`0110`
	`0111`
	`0011`

`13 50 55 95 100`	`101111`
	`100111`
	`100111`
	`101111`
	`101101`
	`100001`

Problem I
Rome 16
(1 sec)

Input: Standart input
Output: Standart output

Roman numerals are numeral system of ancient Rome based on letters of the alphabet, which are combined to signify the sum (or in some cases, the difference) of their values. This system is decimal but not directly positional, since same digits standing at different positions in usual decimal system are represented by different Roman digits, and one decimal digit could be represented by few Roman.

There are seven Roman numerals associated to the decimal I = 1, V = 5, X = 10_{10}, L = 50_{10}, C = 100_{10}, D = 500_{10}, M = 1000_{10}. Generally, Roman numerals are written in descending order from left to right, and are added sequentially, for example MMX (2010) is interpreted as 1000 + 1000 + 10. Certain combinations employ a subtractive principle, which specifies that where a symbol of smaller value precedes a symbol of larger value, the smaller value is subtracted from the larger value, and the result is added to the total. For example, MCMXLIV equals 1944. I may precede V and X, X may precede L or C. The numerals V, L, and D may not be followed by a numeral of greater or equal value.

Subtractive principle has been introduced at medevail adges. Originally it allowes only one symbol of smaller value to preced a symbol of larger value. Nowadays this limitation could be skipped to allow shorter notation for some numbers. Also modern computers are much better in hexadecimal numbers and probably it makes sence to introduce “hexadedimal” Roman notation.

Let hexadecimal Roman notation be a notation in which Roman numerals are equivalent to the following numbers: I = 1, V = 8, X = 10_{16}, L = 80_{16}, C = 100_{16}, D = 800_{16}, M = 1000_{16}. Is some hexadecima digit can be represented using addition and subtraction, notation with less symbols is used. If both notations produce same number of symbols addition is used. For example number F_{16} is written as IX_{16}, $\mathrm{5C8}_{16}$ becomes $\mathrm{CCCDLXXXXV}_{16}$.

You need to write a program which can perform computation in hexadecimal Roman notation, namely the following operations: addition, subtraction, and multiplication. All source numbers and results will be integers within [1, $\mathrm{4FFF}_{16}$].

The first line at input contains one integer **N** (**0**<**N**$\leq$**100**), the number of test lines. Each line contains test data in format “<A><O><B>” whitout any spaces. <A> and <B> are numbers in hexadecimal Roman notation, <O> designates operation: `+`, `-`, `*`.

Every line at output should contain hexadecimal Roman notation of calculation result for corresponding test line at the input.

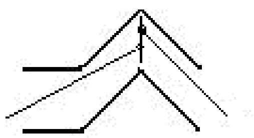
Input	Output
3	XXXLV
XIIV+XXXXII	CDXXVIIII
XXIIII*XXXIII	CXI
XXIV*IV	

All-Ukrainian Collegiate Programming Contest
Final
Vinnytsia National Technical University
September 10, 2010

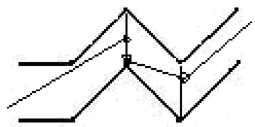
Problem J
Tunnel
(1 sec)

Input: Standard input
Output: Standard output

A tunnel with square profile consists of **(n-1)** sections. Floor in each section is flat and can go up or down along the section. Coordinates $[x_1,y_1]$, $[x_2,y_2]$,..., $[x_n,y_n]$ where $x_1 < x_2 < \ldots < x_n$.describe points where tunnel's floor starts, ends, or one section joins other. Ceiling of the tunnel is 1 meter above the floor and corresponding sections' join points are at coordinates $[x_i,y_i + 1]$.



1 light translator
Example 1



2 light translators
Example 2

Laser beam is directed to the tunnel from its start. To facilitate beam transition through the tunnel light translators could be installed at section boundaries. These translators can retransmit laser emission in necessary direction, moreover retransmitted beam not necessarily starts at the point where coming beam hit the translator.

As soon as light translators could be installed at sections boundaries only, their coordinates can only be $x_1, x_2, \ldots, x_n$.

Your task is to determine the minimal number of light translators necessary to allow bean transition through the tunnel.

The first line at input contains integer **N (2<=N<=1000)**. Following **N** lines contain 2 floats each – coordinates x_i and y_i.

The only line at output should contain one integer – the minimal number of light translators necessary.

Input	Output
4 1 1 2 1 3 2 4 1	1
5 1 1 2 1 3 2 4 1 5 2	2

Problem K
Young Hacker

(3 sec)

Input: Standard input
Output: Standard output

Hackers often have to crack passwords for different data encryption systems. A novice hacker Bill faced such a problem one day. After performing several experiments he noticed regularity in the formation of an encryption key. He knew that a key is an odd integer **K**, such that $\mathbf{K^2}$ does not divide (**K-1)**! and its value is in the range **[A; B] (A ≤ K ≤ B).** Note, that (**K-1)! = (K-1)*(K-2)*...*2*1**). He was not able to advance further due to his poor mathematics.

To help young hacker you have to find all possible values of the key.

Input consists of two integers **A** and **B (3 ≤ A < B ≤ 10^{18}, B - A ≤ 100)**.

Your program has to print to a single line of output all possible values of key **K** in ascending order divided by a single space. It is guaranteed that there is at least one key in the range.

Input	Output
3 8	3 5 7

Problem L

Leapers

(1 sec)

Input: Standard input
Output: Standard output

There is a chessboard of the size M×N. K fairy chess pieces called (p,q)-leapers (p<q) are placed in some squares on this board. Leaper's move is similar to a regular chess knight's move, with some constraints though. When (p,q)-leaper moves, it can move p squares horizontally and q squares vertically (only upward), or q squares horizontally (only leftward) and p squares vertically. In other words, the move to q squares must be in a direction where corresponding coordinate decreases. Moving outside of the board is prohibited. However several leapers are allowed to occupy the same square.

Two players are playing the game, alternating moves. In his turn a player chooses some leaper and moves it according to the rules. The player who is not able to move any leaper loses the game. Given a board configuration determine the winner, assuming both players play optimally.

The first line of input contains 5 integers: M, N, K, p, q ($1 \le M,N \le 10^9$, $1 \le K \le 10^5$, $1 \le p < q \le 20$). Each of following K lines contains coordinates r_i and c_i of corresponding leaper ($1 \le r_i \le M$, $1 \le c_i \le N$).

The single line of output should contain string **First**, if the first player wins the game under optimal strategy, and **Second** otherwise.

Input	Output

`10 10 2 1 2`
`3 7`
`7 3`
	`Second`

`7 4 3 1 3`
`2 3`
`1 5`
`4 3`
	`First`