**Stack of coins**

**Send Feedback**

Ram and Laxman are two best friends. One day they get three stacks of coins where each coins has the same diameter, but they may vary in height. They both decide to make height of these stack equal but the problem seems to be difficult to both of them. So they need your help. You can change the height of a stack by removing and discarding its topmost coin any number of times.

Find the maximum possible height of the stacks such that all of the stacks are exactly the same height. This means you must remove zero or more coins from the top of zero or more of the three stacks until they're all the same height, then print the height.

Note: An empty stack is still a stack.

**Input Format:**

The first line contains three space-separated integers, n1, n2, and n3, describing the respective number of coins in stacks 1, 2, and 3. The subsequent lines describe the respective heights of each coin in a stack from top to bottom:

The second line contains space-separated integers describing the coin heights in stack 1. The first element is the top of the stack.

The third line contains space-separated integers describing the coin heights in stack 2. The first element is the top of the stack.

The fourth line contains space-separated integers describing the coin heights in stack 3. The first element is the top of the stack.

**Constraints:**

0 ≤ n1,n2,n3 ≤ 100000

1 ≤ height of any coin ≤ 100

Time limit = 1 sec

**Output Format:**

Print a single integer denoting the maximum height at which all stacks will be of equal height.

**Sample Input 1:**

5 3 4

3 2 1 1 1

4 3 2

1 1 4 1

**Sample Output 1:**

5

**Explanation:**

Observe that the three stacks are not all the same height. To make all stacks of equal height, we remove the first coin from stacks 1 and 2, and then remove the top two coin from stack 3.

As a result, the stacks undergo the following change in height:

1) 8-3 = 5

2) 9-4 = 5

3) 7-1-1 = 5

All three stacks now have height=5. Thus, we print as our answer.

**Swine Gene**

**Send Feedback**

A multinational company is asking you to help them genetically modify an apple. In order for the apples to grow faster, to get more of them, to make them bigger and make them look nicer and more symmetrical, the apple's DNA requires an insertion of a certain swine gene.

The apple's DNA is represented by a series of characters from the set {A, C, G, T}. The required swine gene is also comprised of characters from this set. The apple's DNA should be injected with some characters into some places, so that the resulting sequence contains a swine gene somewhere (in successive locations). To make things a bit more complicated, inserting each of the characters A, C, G, T has its own cost.

Help this multinational company in achieving their goal with the lowest possible total cost. As a reward, you get a ton of their apples.

**Input Format:**

The first line of input contains a sequence of N characters which represent the apple's DNA.

The second line of input contains a sequence of M characters which represent the swine gene that we want to insert into the apple's DNA.

Both the sequences are comprised only of characters from the set {A, C, G, T}.

The third line of input contains four integers from the interval [0, 1000]: the cost of inserting one character A, C, G, T, in that order.

**Constraints:**

1 ≤ N ≤ 10000

1 ≤ M ≤ 5 000

Time limit = 1 sec

**Output Format:**

The first and only line of output must contains the minimal total cost.

**Sample Input 1:**

GTA

CAT

5 7 1 3

**Sample Output 1:**

10

**Sample Input 2:**

TCGCGAG

TGCAG

10 10 15 15

**Sample Output 2:**

25

**Explanation:**

For testcase 1:

Some of the possible solutions are GCATA (inserted characters are C and first A) and GTCAT (the inserted characters are C and last T), the first solution costs 7 + 5, the second 7 + 3.

**Valid String**

**Send Feedback**

There is a town named Chefland where Chef lives. One can only enter the Chefland through visa which is granted by head Chef of the town named Chef Sanjeev Kapoor. Chef Sanjeev is very interested in the validation of string.

Hence he devised a test to get into the Chefland. Chef Sanjeev Kapoor considers a string to be valid if all characters of the string appear the same number of times. It is also valid if he can remove just 1 character at 1 index in the string, and the remaining characters will occur the same number of times. Given a string S, determine if it is valid. If so, return YES, otherwise return NO.

For example, if S="abc", it is a valid string because frequencies are {a:1, b:1,c:1}. So is S="abcc" because we can remove one c and have 1 of each character in the remaining string. If S="abccc" however, the string is not valid as we can only remove 1 occurrence of c. That would leave character frequencies of {a:1, b:1, c:2}.

**Input Format:**

A single string S.

**Constraints:**

1 ≤ |S| ≤ 100000

Each character s[i] belongs to [a-z]

Time limit = 1 sec

**Output Format:**

Print YES if string S is valid, otherwise, print NO.

**Sample Input 1:**

aabbcd

**Sample Output 1:**

NO

**Sample Input 2:**

abcdefghhgfedecba

**Sample Output 2:**

YES

Explanation:

For testcase 2:

All characters occur twice except for e which occurs 3 times. We can delete one instance of e to have a valid string.

**Find the correct password**

**Send Feedback**

Mike is an evil plotting genius and has gotten hold of a list of all possible passwords for a certain user account. The first thing he noticed was all the passwords are of odd length. Mike assumes that the correct password is the one which can be found in both the original and reverse order in the list. For example, if the word “tulipan” would be the correct password, the word “napilut” has to also appear in the list. Given that both words are correct passwords, Mike will try to use both, one at a time.

Help Mike discover what the correct password is and output its length and central character.

**Input Format:**

The first line of input contains the integer N, the number of possible passwords.

Each of the following N lines contains a single word S, its length being an odd number. All characters are lowercase letters of the English alphabet.

**Constraints:**

1 ≤ N ≤ 100

2 < S < 14

Time limit = 1 sec

**Output Format:**

The first and only line of output must contain the length of the correct password and its central letter.

The solution will be unique.

**Sample Input 1:**

4

las

god

psala

sal

**Sample Output 1:**

3 a

**Sample Input 2:**

4

kisik

ptq

tttrp

tulipan

**Sample Output 2:**

5 s

**Explanation:**

For testcase 1:

The required pair of words is “las” and “sal”. Their length is 3 letters and the central character is 'a'.

For testcase 2:

The word “kisik” can be found in both the original and reverse order on the list (the word is a palindrome), so it is a valid correct password.

**Robert and his kingdom**

**Send Feedback**

It has been a prosperous year for King Robert and he is rapidly expanding his empire. This kingdom has many cities connected by two-way roads(bidirectional). To ensure higher connectivity, two cities are sometimes directly linked by more than one road.

King Robert wants a better connectivity between cities of his kingdom. To do so, he wants to determine the sum of the minimum distances between each pair of cities and represent it in binary representation.

His kingdom has N cities and M bidirectional roads. Each of the roads has a distinct length, and each length is a power of two (i.e., 2 raised to some exponent). It's possible to reach any city from any other city.

**Input Format:**

->The first line contains two space-separated integers denoting N (the number of cities) and M (the number of roads), respectively.

->Each line i of the M subsequent lines contains the respective values of Ai, Bi, and Ci as three space-separated integers. These values define a bidirectional road between cities Ai and Bi having length 2^Ci.

**Constraints**

1<=N<=100000

1<=M<=200000

1<=Ai,Bi<=N, (Ai != Bi)

0<=Ci<M

All Ci are distinct

**Output Format:**

Find the sum of minimum distances of each pair of cities and print the answer in binary representation.

**Sample Input:**

5 6

1 3 5

4 5 0

2 1 3

3 2 1

4 3 4

4 2 2

**Sample Output:**

1000100

**Explanation:**

let d(x,y) be the minimum distance between city x and city y.

d(1,2)=8

d(1,3)=10

d(1,4)=12

d(1,5)=13

d(2,3)=2

d(2,4)=4

d(2,5)=5

d(3,4)=6

d(3,5)=7

d(4,5)=1

Sum = 8+10+12+13+2+4+5+6+7+1 = 65 = 1000100(in binary)

**Abhishek's homework**

**Send Feedback**

Abhishek couldn't even finish half of the homework exercises in "Advanced topics in maths" class. The teacher is really upset and gives him one final problem to solve - if he can't solve it, he is gonna fail the course.

Abhishek is a given a number N, he need to find no. of distinct perfect squares or perfect cubes from 1 to N ( both inclusive ).

Abhishek wanted to ask the teacher about the definition of perfert square and perfect cube but he remembered that he had heard this term in last week's lesson just before falling asleep. Now his life is in your hands again!

Note : If a number, like 729, which is both square ( 27^2 ) and cube (9^3) should be counted once only.

**Input Format:**

First line contains T no. of test cases

The following line contains T space separated integers. Each integer represents the value of N.

**Constraints**

1≤T≤1000

1≤N≤1000000000000000

**Output Format:**

For every test case print count in new line

**Sample Input1:**

2

67 10

**Sample Output1:**

10

4

**Sample Input2:**

2

100 425

**Sample Output2:**

12

25

**Explanation:**

For sample input1:

For Test #1, The numbers are 1,4,8,9,16,25,27,36,49,64

For Test #2, The numbers are 1,4,8,9

**Hungry Hackerman**

**Send Feedback**

Hackerman is hungry programmer and has stumbled upon a local restaurant. The restaurant offers N meals and has an interesting pricing policy: each meal i has two assigned prices, Ai and Bi. hackerman pays A only for the first ordered meal, while B prices apply for all other meals.

Hackerman can't decide how many meals to order. In order to make his decision easier, he has asked you to compute, for each k between 1 i N (inclusive), the minimum total price for k ordered meals. Hackerman doesn't care which particular meals he orders or in which order he orders them, however he won't order the same meal twice. Order, order, order.

**Input Format:**

The first line of input contains the positive integer N, the number of different meals offered by the restaurant.

Each of the following N lines contains two positive integers, Ai and Bi, the prices for meal i as described above.

**Constraints:**

2 ≤ N ≤ 500000

1 ≤ Ai,Bi ≤ 1000000000

**Output Format:**

Output must consist of N lines, where line k contains the minimum price for ordering exactly k different meals.

**Sample Input:**

3

10 5

9 3

10 5

**Sample Output:**

9

13

18

**Explanation:**

k = 1: Hackerman pays A2 = 9 for the starting meal 2.

k = 2: Hackerman pays A1 = 10 for the starting meal 1, then B2 = 3 for meal 2.

**Hackerman and his chessboard**

**Send Feedback**

Hackerman has become a hardcore patriot, so he has asked you to draw him a Croatian chessboard (checkerboard).

The chessboard consists of red and white cells. The upper left cell is red, with the remaining cells alternating between white and red in rows as well as columns. We will represent red areas with ‘X’ (uppercase letter x) characters, and white areas with ‘.’ (period) characters.

Hackerman's chessboard should consist of R × C cells, that is, R rows and C columns. Each row should be A characters high, and each column B characters wide. Consider the sample tests below for further clarification

**Input Format:**

The first line of input contains two positive integers R and C from the problem statement.

The second line of input contains two positive integers A and B from the problem statement.

**Constraints:**

1 ≤ R, C ≤ 10

1 ≤ A, B ≤ 10

**Output Format:**

The output must consist of a total of R \* A rows and C \* B columns, forming the chessboard described above.

**Sample Input 1:**

2 4

2 2

**Sample Output 1:**

XX..XX..

XX..XX..

..XX..XX

..XX..XX

**Sample input 2:**

5 5

2 3

**Sample Output 2:**

XXX...XXX...XXX

XXX...XXX...XXX

...XXX...XXX...

...XXX...XXX...

XXX...XXX...XXX

XXX...XXX...XXX

...XXX...XXX...

...XXX...XXX...

XXX...XXX...XXX

XXX...XXX...XXX

**Maximize Rs**

**Send Feedback**

The CS & IT Department students have been facing tough competitions from each other since ages, in being the best in research & innovation.

This time CS Department is taken the edge over IT Department by designing a special weather predictor device which inputs previous years data and perform analysis over it to give a prediction of current weather condition. This device needs a unique string code to be unlocked and activated, which is always kept safe with the Head Of Department's Computer.

This unique string code consists only character 'R' and 'K'. This time task is given to you to extract the code using the given hint. CS Department gives you a string S consisting of characters 'R' and 'K' only. Now you are allowed to do exactly one move that is you have to choose two indices i and j (1<=i<=j<=|S| where |S| is string length ) and flip all the characters at position X where i<=X<=j. Flipping the character means :

if S[X]='R' : //If character at position X is 'R' then change it to 'K'

S[X]='K'

else : //else if character at position X is 'K' then change it to 'R'

S[X]='R'

Now your goal is that after exactly one move you have to obtain the maximum number of R's. So help department with maximum R's.

**Input Format:**

The first line contains the number of test cases T. Each test case contains a string S consisting of characters 'R' and 'K' only.

**Constraints:**

1<=T<=10

1<=|S| <=10^6 (length of string)

**Output Format:**

For each test case print the maximal number of R's that can be obtained after exactly one move.

**Sample Input:**

2

KKRRRK

RKRK

**Sample Output:**

5

3

**Explanation:**

In first sample, flip the characters from 1 to 2(i=1,j=2).After flipping string becomes: RRRRRK. So, it contains five R's. There is no way to make the complete string equal to RRRRRR.

**Catch Brackets**

**Send Feedback**

At Coding Ninjas, we love to play games. We all are fond of a game called Catch Brackets.

Game Description: Let us say that screen is divided into 'row' \* 'col' number of pixels. The main character is kept at a pixel on the bottom most row. The position of main character is denoted by character 'M'. The main character can move one pixel left or right or stay still. This means it is not necessary to move the main character.

It is given that all the other pixels of the game (except the last row) are filled by either one of these elements: '(' (opening bracket), ')' (closing brackets), '.' (empty pixel), '\*' (landmines). Except the last row, all the other rows simultaneously move one pixel down and eventually move out of the screen.

While each row is moving out, the main character want to move and catch brackets. The caught bracket gets added to its array. The objective of the main character is to make longest array of balanced brackets.

The game becomes over when either main character catches a landmine or all the rows move out of the screen.

An array or sequence of balanced brackets is defined as follows:

1. () is a sequence of balanced brackets.

2. If p is a sequence of balanced brackets, then (p) is also a sequence of balanced brackets.

3. If p and q are sequences of balanced brackets, then pq is also a sequence of balanced bracket.

Your task is to find the longest possible array of balanced brackets.

**Input format:**

The first line of input contains two space separated integers, that denote the values of 'row' and 'col'. The value of 'row' and 'col' lies in the range: [1, 300]. Each of following 'row' number of lines contains 'col' number of space separated characters. Characters can be '.', '(', ')', '\*' or 'M'.

The test cases would be such that at least one array of balanced brackets is possible to catch.

**Constraints:**

Time Limit: 0.5 seconds

**Output Format:**

The first line of output contains the length of longest array or sequence of balanced brackets.

The following line contains the array itself. If there are multiple answers, output the lexicographically smallest one.

**Sample Input 1:**

5 4

.)..

\*.((

.).\*

(.((

.M..

**Sample Output 1:**

4

()()

**Rahul and Strings**

**Send Feedback**

Rahul likes equality. One day his professor gave him two Strings, A and B, he want to make them equal. But Rahul is little busy in studying, so he want your help in determining if it's possible to make A equal to B using the following operations. If so, print 1 on a new line. Otherwise, print 0.

You can perform the following operations on the string, :

1) You can convert zero or more of A's lowercase letters to uppercase.

2) You can remove all of the remaining lowercase letters in string A.

For example, given A=AbcDE and B=ABDE, in A we can convert b and delete c to match B. If A=AbcDE and b=AD, matching is not possible because capital letters can not be deleted.

**Input Format:**

The first line contains a single integer q, the number of queries.

Each of the next q pairs of lines is as follows:

- The first line of each query contains a single string, A.

- The second line of each query contains a single string, B.

**Constraints**

1<=q<=10

1<|a|,|b|<=1000

String A consists only of uppercase and lowercase English letters, ascii[A-Z,a-z].

String B consists only of uppercase English letters, ascii[A-Z].

**Output Format:**

For each query, print 1 on a new line if it's possible to make string A equal to string B. Otherwise, print 0.

**Sample Input:**

1

sdEGgGb

SEGGB

**Sample Output:**

1

**Explanation:**

We have A=sdEGgGb and B=SEGGB. We perform the following operation:

1)Capitalize the letters s and b in A so that A=SdEGgGB.

2)Delete all the remaining lowercase letters in A so that A=SEGGB.

Because we were able to successfully convert A to B, we print 1 on a new line.

**Jessa and her array**

**Send Feedback**

Walter is out of town, and Jessa is alone at the office. She was bored, so she took a piece of paper and wrote down a sequence A of length N, which contains each positive integer between 1 and N, inclusive, exactly once. After that, she took another piece of paper and wrote down M descriptions of the sequence A.

Each description has one of the following formats:

If the maximum number in positions between x and y (inclusive) equals v, then it would be written in following format:1 x y v

If the smallest or minimum number in positions between x and y (inclusive) equals v, then it would be written in following format: 2 x y v.

Then Walter came, saw, and stole the first paper. Jessa is desperate and has asked you to find some sequence matching the descriptions, not necessarily equal to the original sequence.

Note:If multiple solution exist, you can print any.

**Input Format:**

The first line of input contains two positive integers N, the length of the sequence, and M, the number of descriptions.

Each of the following M lines contains a description as described above.

**Constraints:**

1 ≤ N ≤ 200

0 ≤ M ≤ 40000

**Output Format:**

The first and only line of output must contain a sequence of N space-separated positive integers (matching the descriptions and containing all positive integers from 1 to N), or -1 if no such sequence exists.

**Sample Input1:**

3 2

1 1 1 1

2 2 2 2

**Sample Output1:**

1 2 3

**Sample Input2:**

4 2

1 1 1 1

2 3 4 1

**Sample Output2:**

-1

**Can you calculate the strength of a number?**

**Send Feedback**

Let us begin with a positive integer N and find the smallest positive integer which doesn't divide N. If we repeat the procedure with the resulting number, then again with the new result and so on, we will eventually obtain the number 2 (two). Let us define strength(N) as the length of the resulting sequence.

For example, for N = 6 we obtain the sequence 6, 4, 3, 2 which consists of 4 numbers, thus strength(6) = 4.

Given two positive integers A < B, calculate the sum of strengths of all integers between A and B (inclusive), that is,

strength(A) + strength(A + 1) + ... + strength(B).

**Input Format:**

The first and only line of input contains two positive integers, A and B.

**Constraints:**

3 ≤ A < B < 10^17

Time limit: 1 sec

**Output Format:**

The first and only line of output should contain the requested sum of strengths.

**Sample Input 1:**

3 6

**Sample Output 1:**

11

**Sample Input 2:**

100 200

**Sample Output 2:**

262

**Guess the sequence**

**Send Feedback**

Ankit and Ayush have devised a new game, creatively named Rotate. First, Ankit imagines a number sequence of length N and divides it into sections, with each section containing K numbers (K evenly divides N). The first section contains numbers in the first K positions in the sequence, the second section the following K positions, and so on.

Then, Ayush asks Ankit to apply a number of operations on the sequence, with each operation beingone of the following two types:

1. Rotate the numbers in each section to the left/right by X positions

2. Rotate the whole sequence to the left/right by X positions

Notice that an operation of type 2 can change the numbers belonging to each section. After applying all the operations, Ankit reveals the final sequence to Ayush. Ayush's task is finding Ankit's starting sequence. He has asked you for help.

**Input Format:**

The first line of input contains three positive integers: N, the length of the sequence, K is the size of each section, and Q is the number of operations.

Each of the following Q lines contains two integers: A, the operation type, and X, the number of positions to rotate by. A negative number represents rotation to the left, while a positive one represents rotation to the right.

The last line of input contains N space-separated integers Zi representing the final sequence (after applying all operations).

**Constraints:**

1 ≤ N ≤ 100000

1 ≤ K ≤ 100 000

1 ≤ Q ≤ 100000

1 ≤ A ≤ 2

-100000 ≤ X ≤ 100000

0 ≤ Zi ≤ 100000

Time limit: 1 sec

**Output Format:**

The first and only line of output must contain the required starting sequence

**Sample Input 1:**

4 2 2

2 2

1 1

3 2 1 0

**Sample Output 1:**

0 1 2 3

**Explanation:**

The starting sequence is 0 1 2 3. After the first operations, the sequence is 2 3 0 1, and after the second operation, it becomes 3 2 1 0. Ths corresponds to the final sequence.

**Sample Input 2:**

8 4 4

1 3

1 15

1 -5

2 -1

6 10 14 19 2 16 17 1

**Sample Output 2:**

6 10 14 1 2 16 17 19

**Sebi, the great mathematician**

**Send Feedback**

Sebi likes solving mathematical problems a lot. He spends a lot of time in getting expertise in mathematics. One day his friend gave a problem. In order to prove it expertise, he need to solve this problem.

Sebi is given two integers A and B. He has to calculate the sum of distances between each pair of numbers belonging in the interval [A, B].

**Input Format:**

The first and only line of input contains integers A, B.

**Constraints:**

1 ≤ A ≤ B ≤ 10^50000

Time limit = 1 sec

**Output Format:**

The first and only line of output must contain the required number from the text. Given that the number could be extremely large, output answer modulo 1000000007.

**Sample Input 1:**

288 291

**Sample Output 1:**

76

**Sample Input 2:**

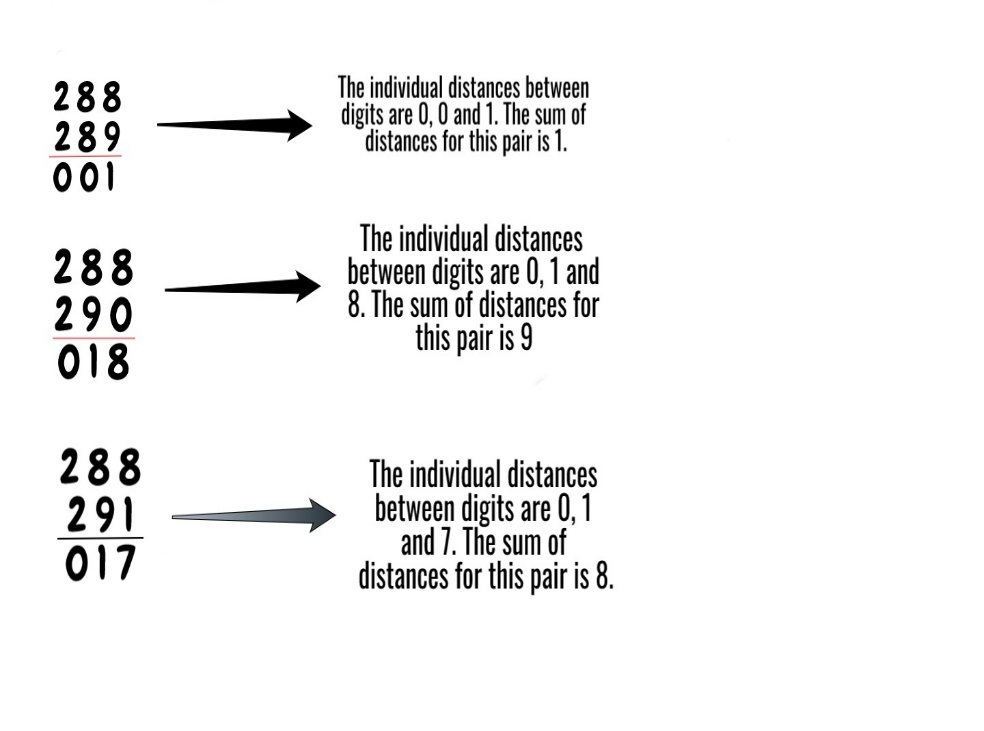
1 5

**Sample Output 2:**

40

**Explanation:**

For testcase 1:



The other distances are, respectively, (289, 290) = 10, (289, 291) = 9, (290, 291) = 1. Each of them counts twice, which is in total 2 \* (1 + 9 + 8 + 10 + 9 +1) = 76.