

## A. Sort the Array

Time limit: 1s

Memory limit: 256 MB

Being a programmer, you like arrays a lot. For your birthday, your friends have given you an array  $a$  consisting of  $n$  **distinct** integers.

Unfortunately, the size of  $a$  is too small. You want a bigger array! Your friends agree to give you a bigger array, but only if you are able to answer the following question correctly: is it possible to sort the array  $a$  (in increasing order) by reversing **exactly one** segment of  $a$ ? See definitions of segment and reversing in the notes.

**Input**

The first line of the input contains an integer  $n$  ( $1 \leq n \leq 10^5$ ) — the size of array  $a$ .

The second line contains  $n$  distinct space-separated integers:  $a[1], a[2], \dots, a[n]$  ( $1 \leq a[i] \leq 10^9$ ).

**Output**

Print "yes" or "no" (without quotes), depending on the answer.

If your answer is "yes", then also print two space-separated integers denoting start and end (start must not be greater than end) indices of the segment to be reversed. If there are multiple ways of selecting these indices, print any of them.

**Examples****input**

3  
3 2 1

**output**

yes  
1 3

**input**

4  
2 1 3 4

**output**

yes  
1 2

**input**

4  
3 1 2 4

**output**

no

**input**

2  
1 2

**output**

yes
1 1

**Note**

Sample 1. You can reverse the entire array to get  $[1, 2, 3]$ , which is sorted.

Sample 3. No segment can be reversed such that the array will be sorted.

*Definitions*

A segment  $[l, r]$  of array  $a$  is the sequence  $a[l], a[l + 1], \dots, a[r]$ .

If you have an array  $a$  of size  $n$  and you reverse its segment  $[l, r]$ , the array will become:

$a[1], a[2], \dots, a[l - 2], a[l - 1], a[r], a[r - 1], \dots, a[l + 1], a[l], a[r + 1], a[r + 2], \dots, a[n - 1], a[n]$ .

## B. Two Substrings

Time limit: 2s

Memory limit: 256 MB

You are given string  $s$ . Your task is to determine if the given string  $s$  contains two non-overlapping substrings "AB" and "BA" (the substrings can go in any order).

**Input**

The only line of input contains a string  $s$  of length between 1 and  $10^5$  consisting of uppercase Latin letters.

**Output**

Print "YES" (without the quotes), if string  $s$  contains two non-overlapping substrings "AB" and "BA", and "NO" otherwise.

**Examples**

<b>input</b>
ABA
<b>output</b>
NO

<b>input</b>
BACFAB
<b>output</b>
YES

<b>input</b>
AXBYBXA
<b>output</b>
NO

**Note**

In the first sample test, despite the fact that there are substrings "AB" and "BA", their occurrences overlap, so the answer is "NO".

In the second sample test there are the following occurrences of the substrings: **BACFAB**.

In the third sample test there is no substring "AB" nor substring "BA".

## C. Combinador

Time limit: 1s

Implemente um programa denominado combinador, que recebe duas strings e deve combiná-las, alternando as letras de cada string, começando com a primeira letra da primeira string, seguido pela primeira letra da segunda string, em seguida pela segunda letra da primeira string, e assim sucessivamente. As letras restantes da cadeia mais longa devem ser adicionadas ao fim da string resultante e retornada.

**Entrada**

A entrada contém vários casos de teste. A primeira linha contém um inteiro **N** que indica a quantidade de casos de teste que vem a seguir. Cada caso de teste é composto por uma linha que contém duas cadeias de caracteres, cada cadeia de caracteres contém entre 1 e 50 caracteres inclusive.

**Saída**

Combine as duas cadeias de caracteres da entrada como mostrado no exemplo abaixo e exiba a cadeia resultante.

Exemplo de Entrada	Exemplo de Saída
2 Tpo oCder aa bb	TopCoder abab

\* Este problema é de autoria do TopCoder (www.topcoder.com/tc) e foi adaptado por Jeferson T. para utilização (autorizada) no URI OJ.

\* A reprodução não autorizada deste problema sem o consentimento por escrito de TopCoder, Inc. é estritamente proibida.

Por TopCoder\*  EUA

## D. Cormen --- The Best Friend Of a Man

Time limit: 1s

Memory limit: 256 MB

Recently a dog was bought for Polycarp. The dog's name is Cormen. Now Polycarp has a lot of troubles. For example, Cormen likes going for a walk.

Empirically Polycarp learned that the dog needs at least  $k$  walks for any two consecutive days in order to feel good. For example, if  $k = 5$  and yesterday Polycarp went for a walk with Cormen 2 times, today he has to go for a walk at least 3 times.

Polycarp analysed all his affairs over the next  $n$  days and made a sequence of  $n$  integers  $a_1, a_2, \dots, a_n$ , where  $a_i$  is the number of times Polycarp will walk with the dog on the  $i$ -th day while doing all his affairs (for example, he has to go to a shop, throw out the trash, etc.).

Help Polycarp determine the minimum number of walks he needs to do additionally in the next  $n$  days so that Cormen will feel good during all the  $n$  days. You can assume that on the day before the first day and on the day after the  $n$ -th day Polycarp will go for a walk with Cormen exactly  $k$  times.

Write a program that will find the minimum number of additional walks and the appropriate schedule — the sequence of integers  $b_1, b_2, \dots, b_n$  ( $b_i \geq a_i$ ), where  $b_i$  means the total number of walks with the dog on the  $i$ -th day.

**Input**

The first line contains two integers  $n$  and  $k$  ( $1 \leq n, k \leq 500$ ) — the number of days and the minimum number of walks with Cormen for any two consecutive days.

The second line contains integers  $a_1, a_2, \dots, a_n$  ( $0 \leq a_i \leq 500$ ) — the number of walks with Cormen on the  $i$ -th day which Polycarp has already planned.

**Output**

In the first line print the smallest number of additional walks that Polycarp should do during the next  $n$  days so that Cormen will feel good during all days.

In the second line print  $n$  integers  $b_1, b_2, \dots, b_n$ , where  $b_i$  — the total number of walks on the  $i$ -th day according to the found solutions ( $a_i \leq b_i$  for all  $i$  from 1 to  $n$ ). If there are multiple solutions, print any of them.

**Examples****input**

```
3 5
2 0 1
```

**output**

```
4
2 3 2
```

**input**

```
3 1
0 0 0
```

**output**

1 0 1 0
------------

<b>input</b>
--------------

4 6 2 4 3 5
----------------

<b>output</b>
---------------

0 2 4 3 5
--------------

## E. Given Length and Sum of Digits...

Time limit: 1s

Memory limit: 256 MB

You have a positive integer  $m$  and a non-negative integer  $s$ . Your task is to find the smallest and the largest of the numbers that have length  $m$  and sum of digits  $s$ . The required numbers should be non-negative integers written in the decimal base without leading zeroes.

**Input**

The single line of the input contains a pair of integers  $m, s$  ( $1 \leq m \leq 100, 0 \leq s \leq 900$ ) — the length and the sum of the digits of the required numbers.

**Output**

In the output print the pair of the required non-negative integer numbers — first the minimum possible number, then — the maximum possible number. If no numbers satisfying conditions required exist, print the pair of numbers "-1 -1" (without the quotes).

**Examples**

<b>input</b>
2 15
<b>output</b>
69 96

<b>input</b>
3 0
<b>output</b>
-1 -1

## F. Bicho da Goiaba

Time limit: 1s

Recentemente na Nlogônia, diversas goiabeiras estão sendo infestadas por uma espécie de larva denominada Marangonis Ovidius, conhecida por ser extremamente nociva às plantações. Porém, o Dr. Icaronieris, prodígio da medicina atual e amante de boas goiabas, inventou um remédio que elimina essa infestação por completo.

Ainda não contente com o efeito do remédio em somente uma goiabeira, ele foi além: criou um mecanismo que, dia após dia, espalha o remédio para todas as árvores adjacentes às árvores cujo remédio está ativo. Por exemplo, o remédio é aplicado em uma árvore específica. No dia seguinte, ele se espalha para as árvores adjacentes a esta, e no outro, para as adjacentes às adjacentes da árvore inicial, e assim por diante, até que toda a infestação seja eliminada.

O doutor, no entanto, não possui tempo para testar a eficiência e a viabilidade de sua criação no papel. Ele precisa de um programa que, dadas as goiabeiras infectadas e as coordenadas da árvore onde será aplicado o remédio, verifique a quantidade de dias para que todas as goiabeiras estejam curadas. Como pagamento, o doutor lhe prometeu 100 caixas de goiaba (com bicho ou sem bicho, você escolhe) todo mês, além de um vale-ticket no IEF. Vai perder essa?

## Entrada

A primeira linha de entrada é composta por um número inteiro representando a quantidade de casos de teste. Cada caso de teste é composto por **A** ( $2 \leq A \leq 100$ ) e **B** ( $2 \leq B \leq 100$ ) representando a quantidade de linhas e a quantidade de colunas da matriz, respectivamente. Em seguida, será dada uma matriz binária **A** x **B**, com 0 indicando que não há goiabeira e 1 indicando que há uma goiabeira infectada. Posteriormente, serão dadas as coordenadas iniciais **X** ( $1 \leq X \leq A$ ) e **Y** ( $1 \leq Y \leq B$ ) onde será aplicado o remédio. Não haverá goiabeiras sem goiabeiras adjacentes, isto é, o remédio sempre conseguirá alcançar todas as goiabeiras.

## Saída

Para cada caso de teste, imprima a quantidade de dias para que a infestação seja completamente eliminada.


Exemplo de Entrada	Exemplo de Saída
2 3 4 1 0 1 1 1 0 0 1 0 1 1 0 1 1 5 3 1 0 0 0 1 0 0 1 1 0 1 1 1 0 0 2 2	5 3



19/12/2017

(F) Bicho da Goiaba - URI Online Judge - Codepit

The Last Contest 2016 - IFSULDEMINAS

Por João Marcos Salvanini Bellini de Moraes, IFSULDEMINAS  Brazil

## G. Crescimento Populacional

Time limit: 1s

Mariazinha quer resolver um problema interessante. Dadas as informações de população e a taxa de crescimento de duas cidades quaisquer (A e B), ela gostaria de saber quantos anos levará para que a cidade menor (sempre é a cidade A) ultrapasse a cidade B em população. Claro que ela quer saber apenas para as cidades cuja taxa de crescimento da cidade A é maior do que a taxa de crescimento da cidade B, portanto, previamente já separou para você apenas os casos de teste que tem a taxa de crescimento maior para a cidade A. Sua tarefa é construir um programa que apresente o tempo em anos para cada caso de teste.

Porém, em alguns casos o tempo pode ser muito grande, e Mariazinha não se interessa em saber exatamente o tempo para estes casos. Basta que você informe, nesta situação, a mensagem "Mais de 1 seculo."

**Entrada**

A primeira linha da entrada contém um único inteiro **T**, indicando o número de casos de teste ( $1 \leq T \leq 3000$ ). Cada caso de teste contém 4 números: dois inteiros **PA** e **PB** ( $100 \leq PA \leq 1000000$ ,  $PA < PB \leq 1000000$ ) indicando respectivamente a população de A e B, e dois valores **G1** e **G2** ( $0.1 \leq G1 \leq 10.0$ ,  $0.0 \leq G2 \leq 10.0$ ,  $G2 < G1$ ) com um dígito após o ponto decimal cada, indicando respectivamente o crescimento populacional de A e B (em percentual).

**Atenção:** A população é sempre um valor inteiro, portanto, um crescimento de 2.5 % sobre uma população de 100 pessoas resultará em 102 pessoas, e não 102.5 pessoas, enquanto um crescimento de 2.5% sobre uma população de 1000 pessoas resultará em 1025 pessoas. Além disso, não utilize variáveis de precisão simples para as taxas de crescimento.

**Saída**

Imprima, para cada caso de teste, quantos anos levará para que a cidade A ultrapasse a cidade B em número de habitantes. Obs.: se o tempo for mais do que 100 anos o programa deve apresentar a mensagem: Mais de 1 seculo. Neste caso, acredito que seja melhor interromper o programa imediatamente após passar de 100 anos, caso contrário você poderá receber como resposta da submissão deste problema "Time Limit Exceeded".

Exemplo de Entrada	Exemplo de Saída
6	51 anos.
100 150 1.0 0	16 anos.
90000 120000 5.5 3.5	12 anos.
56700 72000 5.2 3.0	Mais de 1 seculo.
123 2000 3.0 2.0	10 anos.
100000 110000 1.5 0.5	100 anos.
62422 484317 3.1 1.0	

Adaptado por Neilor Tonin, URI  Brasil

## H. Tipo de Combustível

Time limit: 1s

Um Posto de combustíveis deseja determinar qual de seus produtos tem a preferência de seus clientes. Escreva um algoritmo para ler o tipo de combustível abastecido (codificado da seguinte forma: 1.Álcool 2.Gasolina 3.Diesel 4.Fim). Caso o usuário informe um código inválido (fora da faixa de 1 a 4) deve ser solicitado um novo código (até que seja válido). O programa será encerrado quando o código informado for o número 4.

**Entrada**

A entrada contém apenas valores inteiros e positivos.

**Saída**

Deve ser escrito a mensagem: "MUITO OBRIGADO" e a quantidade de clientes que abasteceram cada tipo de combustível, conforme exemplo.

Exemplo de Entrada	Exemplo de Saída
8	MUITO OBRIGADO
1	Alcool: 1
7	Gasolina: 2
2	Diesel: 0
2	
4	

Agradecimentos a Cássio F.

Adaptado por Neilor Tonin, URI  Brasil