Lëtzebuerger Informatiksolympiad 2021

Task Description Qualification Round

All the programs must be realized in the form of console applications.

Under the input is meant either the direct entry of data from the keyboard or the redirection from a text file in console mode. Under output is meant either the direct display of data to the screen or the redirection to a text file in console mode (see remarks on the site *www.infosolympiad.lu* under the heading "Les questionnaires").

The formats of the data as well as of the results shown in the execution examples must absolutely be respected.

For testing respectively for submitting a program, the source file (*.cpp, *.pyc or *.java) must be uploaded to the automated online judge CMS (Contest Management System), accessible via the homepage. Please refer to the CMS and check the execution commands there.

The filename of the source file should be the same than the task name (lower case mandatory in Java).

See also the remarks for every programming language given on the homepage (→ *Les questionnaires*).

TASK 1 PETALS 25 POINTS

Description

It is the birthday of your dear friend and you collected his favourite flowers – lioflowers. The lioflower has petals of three different colours (red, white, and blue). Your friend loves those lioflowers that have an equal number of petals of each colour. Therefore, given a lioflower, how many petals do you have to cut off at least to make it the perfect gift for your friend?

Task

Write a program that computes the minimal number of petals you must cut off so that the flower has an equal number of red, white, and blue petals.

Input and output of the program

Input data

The first line contains a single number ***N***, the number of petals. The second line contains ***N*** characters. Each character is either ‘r’ (red petal), ‘w’ (white petal), or ‘b’ (blue petal).

Output data

The minimal number of petals that must be cut off.

Execution examples

First example

Input	Output

`7`
`rrwwbbb`
	`1`

Second example

Input	Output

`5`
`rwbrw`
	`2`

Constraints

$1 \leq \boldsymbol{N} \leq 10^5$.

Distribution of points

Subtask	Points	Description
1	5	$\boldsymbol{N} \leq 6$.
2	5	The colours are in the following order: first the red petals, then the white petals, and finally the blue petals, e.g., rrrrwwwbbbbb.
3	15	No additional constraints.

TASK 2 SUMMERCAMP 25 POINTS

Description

Local authorities organise a summer camp for children. The person who oversees organisational questions decides to organise activities exclusively for groups of at least 5 children whose first names begin with the same letter.

Task

You are invited to write a program which indicates the beginning letters with at least five first names given.

Restrictions

- The number $\boldsymbol{N}$ of first names given is between 1 and 150: $1 \leq \boldsymbol{N} \leq 150$.
- Any first name has between 1 and 30 exclusively alphabetic non accentuated characters (‘a’ to ‘z’ and ‘A’ to ‘Z’). The first character is always given in uppercase.

Input and output of the program

Input data

The first line contains the number of first names ***N*** and is followed by ***N*** lines giving each a first name.

Output data

The text “Activities organised for:” followed by a space. This text is then followed by either ...

- the beginning characters for which there are at least 5 first names-beginning with the character. The characters are given in alphabetical order.
- “/” if there are none.

Execution examples

First example

Input	Output
12 Magalie Lucas Lisa Marie Melanie Lazar Mike Magalie Lana Lejla Maurice Paul	Activities organised for: LM

Second example

Input	Output
13 Yannick Nicolas Jenny Fabio Fabien Vincent Rachel Anastasia Lucas Axelle Melanie Pascal Steve	Activities organised for: /

TASK 3 PASCAL 25 POINTS

Description

Suppose you are given a collection of ***n*** objects which are all different, such as playing cards or balls of different colours. Let ***p*** be an integer such that $0 \le p \le n$. What is the number of different ways in which one can select ***p*** objects among our collection? The answer to this question is written $\binom{n}{p}$ (sometimes C_n^p in French-speaking countries) and is read as "***n*** choose ***p***". They are given by the formula:

$$\binom{n}{p} = \frac{n!}{p! \cdot (n-p)!}$$

where $n! = n \cdot (n-1) \cdot \ldots \cdot 3 \cdot 2 \cdot 1$ is the factorial of ***n***. These numbers have the interesting property that for $1 \le p \le n$:

$$\binom{n+1}{p} = \binom{n}{p} + \binom{n}{p-1}$$

We can interpret this more geometrically if we arrange these numbers into a triangle as follows:

$$\begin{array}{llll} \binom{0}{0} & & & \\ \binom{1}{0} & \binom{1}{1} & & \\ \binom{2}{0} & \binom{2}{1} & \binom{2}{2} & \\ \binom{3}{0} & \binom{3}{1} & \binom{3}{2} & \binom{3}{3} \\ & \ldots & & \end{array} \qquad\qquad \begin{array}{llll} 1 & & & \\ 1 & 1 & & \\ 1 & 2 & 1 & \\ 1 & 3 & 3 & 1 \\ & \ldots & & \end{array}$$

They form what is called **Pascal's triangle**. The interesting property from above now implies that every entry in this triangle is given by the sum of the entry directly above it and the entry above and to the left of it! For instance we can compute $\binom{3}{2} = 3 = 2 + 1 = \binom{2}{2} + \binom{2}{1}$ as follows:

$$\begin{array}{llll} 1 & & & \\ 1 & 1 & & \\ 1 & \boxed{2 \quad 1} & & \\ 1 & 3 & \boxed{3} & 1 \\ & \ldots & & \end{array}$$

Task

Write a program that computes the entries in Pascal's triangle $\binom{n}{p}$.

Constraints

n and ***p*** are natural numbers such as $0 \le p \le n \le 40$.

Input and output of the program

Input data

The two numbers ***n*** and ***p*** separated by a space.

Output data

The value of $\binom{n}{p}$.

Execution examples

First example

Input	Output
`6 0`	`1`

Second example

Input	Output
`6 4`	`15`

Third example

Input	Output
`15 10`	`3003`

Fourth example

Input	Output
`40 20`	`137846528820`

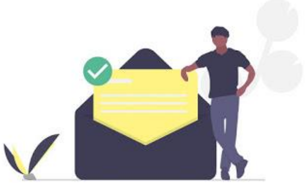
Distribution of points

Subtask	Points	Description
1	15	$0 \le \boldsymbol{p} \le \boldsymbol{n} \le 20$.
2	10	No additional constraints.

TASK 4 CHECKSUM 25 POINTS

Description

The following checksum algorithm can be used to check the validity of certain codes, for instance IBAN (bank account numbers):



1) The code may consist of decimal digits, letters (in upper case and lower case), and spaces. After the spaces have been removed, the code must contain between 5 and 34 characters (digits and letters). The checksum is located in the third and fourth position of the code, and it must be a number between 02 and 98.
2) The four characters at the far left of the code are removed and appended on the right.
3) All digits in the code remain unchanged, but each letter is replaced by a two-digit number in the same place: A => 10, B => 11, …, Z => 35; a => 10, b => 11, … z => 35 (the transformation is case insensitive). The code therefore may become longer.
4) The code now represents an integer number with up to 66 decimal digits. If the remainder of the Euclidean division of this number by 97 is equal to 1, the code is valid.

Example

Let's consider the code "`Lu05 1111 702460 83    0000`" and check its validity:

Step 1 (remove spaces) => "`Lu0511117024608300000`"; the length (20 characters) is acceptable.

Step 2 (rearrange four characters) => "`1111702460830000Lu05`".

Step 3 (replace letters) => "`1111702460830000213005`" (L becomes 21, u becomes 30, the code now has 22 digits).

Step 4 (division): the remainder of 1111702460830000213005 divided by 97 is equal to 1, so the checksum "`05`" is correct.

Task

Write a program that checks the validity of entered codes and suggests correct codes if the one entered is wrong or incomplete.

Input and output of the program

Input data

The first line contains a single number ***N***, the number of codes to analyse. Each of the ***N*** following lines contains a code. This code may have a valid or wrong checksum. It may (but need not) contain one joker “`?`” which represents an illegible character that can be replaced by any permitted character (digit or letter).

Output data

The program produces one line of output for each of the ***N*** codes entered. If the code is complete (without “`?`”) and valid, the output line is “`OK`”. If the code is complete, but has a wrong checksum, the improved code is written in the output, without spaces, with upper case letters, and with the correct checksum in the third and fourth position. If the code contains a joker “`?`”, the valid codes, where “`?`” has been replaced by a digit or an upper case letter, are written in one line, separated by a single space if there is more than one valid code. If there is no valid substitution for a joker “`?`”, the output line is “`impossible`”.

Constraints

$N \in \mathbb{N}^*$, $N \leq 100$, the codes in the input may contain digits, letters, and spaces, but no other type of characters. The length of the entered codes is always acceptable. These input constraints need not be verified by the program.

Remarks

For official IBAN numbers, there are additional constraints (for instance, the first two characters must form a valid two-letter code of an existing country), but such constraints **must not** be checked by the program, because the input data will contain examples that conform to the general description on the preceding page, but **not** necessarily to official IBAN numbers.

Note that the integer number obtained in step 4 of the algorithm has up to 66 decimal digits (about 220 bits). Such a number causes overflow errors when stored in an integer variable (in most programming languages). The problem can be circumvented by an appropriate modulo computation.

Execution example

Input	Output

`5`
`Lu05 1111 702460 83   0000`
`xy12 3456 Hello     7890`
`LU0511117024608 ? 0000`
`Lu   05111170246083   ?000`
`LU0711117024608?0000`
	`OK`
	`XY813456HELLO7890`
	`LU051111702460830000`
	`LU051111702460830000 LU05111170246083F000`
	`impossible`

Distribution of points

Subtask	Points	Description
1	5	Checking the validity of the codes (without joker), each code length is 5.
2	10	Checking the validity of the codes (without joker), with code lengths up to 34.
3	10	General case.