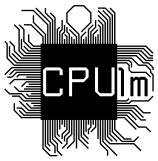
Test Session for the Northwestern Europe Regional Contest 2026

NWERC 2026 Test Session

November 28, 2026

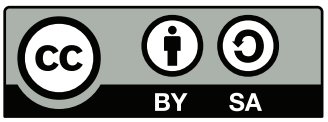






Problems

Assembling Primes
Bicryption
[kœri]wurstwerfen



Copyright © 2026 by the NWERC 2026 jury. This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.
https://creativecommons.org/licenses/by-sa/4.0/

This page is intentionally left blank.

NWERC 2026 Test Session

Assembling Primes
Time limit: 3 seconds

You are given a set S of digits. Find out whether there exists a prime number p such that the set of digits occurring in the decimal representation of p (with no leading zeros) is exactly S. Note that you may use the digits of S multiple times, as long as you use each digit at least once.



Assembling primes using plastic tiles.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10$), the number of digits in S.
- One line with n integers $s_1, \ldots, s_n$ ($0 \leq s_i \leq 9$), so that $S = \{s_1, \ldots, s_n\}$. The s_i are in strictly increasing order.

Output

If a prime number p ($2 \leq p \leq 10^{18}$) using the given set of digits exists, output such a prime number. Otherwise, output "`impossible`".

It can be proven that if any prime number using the given set of digits exists, then such a prime number exists within these bounds.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1	Sample Output 1
6 2 3 4 5 8 9	998244353

Sample Input 2	Sample Output 2
1 1	11

Sample Input 3	Sample Output 3
1 5	5

Sample Input 4	Sample Output 4
2 3 6	impossible

Sample Input 5	Sample Output 5
10 0 1 2 3 4 5 6 7 8 9	509118392752643

NWERC 2026 Test Session

Bicryption
Time limit: 2 seconds

After AKing[1] the practice session, the team Alternative Better Encryption are discussing the solutions to the problems. In particular, they were underwhelmed to see yet another multi-pass problem themed around encryption.



The enigma, an encryption machine that solves this problem. CC BY 2.0 by William Warby on Wikimedia Commons

Instead of the boring encryption protocol of that problem, they invent an alternative and better encryption for secure communication to use in the contest tomorrow. They are calling this their *Newer Well-Encrypted Robust Communication*.

Anna and Ben each encrypt a word to send to Charlotte to test their new invention. These may be two different words. To encrypt a word w, they use two words x and y of the same length as w, where the ith letters of all three words are pairwise distinct for all $1 \leq i \leq |w|$. Charlotte then receives two words x' and y', which are either the two encrypted words from Anna or those from Ben. These words may be out of order, that is, she only knows that either $x' = x$ and $y' = y$, or $x' = y$ and $y' = x$.

Find a strategy for Anna and Ben to encrypt their words such that Charlotte can decrypt the words again. Charlotte should also determine who of Anna or Ben sent the word.

Input

The input consists of:

- One line with "`Anna`", "`Ben`", or "`Charlotte`", representing pass 1, 2, and 3 respectively.
- One line, depending on the pass:
  - **Pass 1:** A string w_a ($1 \leq |w_a| \leq 10^5$), the word Anna encrypts.
  - **Pass 2:** A string w_b ($1 \leq |w_b| \leq 10^5$), the word Ben encrypts.
  - **Pass 3:** Two strings x' and y' ($1 \leq |x'| = |y'| \leq 10^5$), the encrypted words of either Anna or Ben, in arbitrary order.

All input strings consist of only English lowercase letters.

The input strings w_a and w_b are fixed before the interaction, the input of Pass 3 might be chosen adaptively and adversarially depending on your outputs of the first two passes.

This is a multi-pass problem. Your program will be invoked three separate times per test case.

A testing tool is provided to help you develop your solution.

Output

- **Pass 1:** Output two strings x_a and y_a ($|x_a| = |y_a| = |w_a|$) that encode w_a.
- **Pass 2:** Output two strings x_b and y_b ($|x_b| = |y_b| = |w_b|$) that encode w_b.
- **Pass 3:** Output the name of the sender, either "`Anna`" or "`Ben`", and that person's original decrypted word.

[1] An alternative term for fully solving a problemset.

All output strings, except for the name, must consist exclusively of **English lowercase letters** and contain no spaces. The name may additionally contain uppercase letters.

In passes 1 and 2, the ith letters of w, x and y of that pass must all be distinct for $1 \leq i \leq |w|$, that is: $w_i \neq x_i, w_i \neq y_i, x_i \neq y_i$.

Sample Case 1

Sample Input	Pass 1	Sample Output
Anna koeri		curyr wrust

Sample Input	Pass 2	Sample Output
Ben plaintext		notreally encyrpted

Sample Input	Pass 3	Sample Output
Charlotte notreally encyrpted		Ben plaintext

Sample Case 2

Sample Input	Pass 1	Sample Output
Anna koeri		curyr wrust

Sample Input	Pass 2	Sample Output
Ben plaintext		notreally encyrpted

Sample Input	Pass 3	Sample Output
Charlotte wrust curyr		Anna koeri

[kœri]wurstwerfen
Time limit: 1 second



CC BY-SA 4.0 by Bubutine on Wikimedia Commons

Cafeteria food is often lacking in both quality and taste, but the [kœri]werk in Karlsruhe never lets anyone down. They offer vegan [kœri]wurst and fries, and one can choose between six spice mixes for the sauce.

For Conni and her friends, every day is [kœri]wurst day. Today, Conni has already got her food, while her friends are still queueing. Waiting for her friends, Conni is increasingly bored, and she starts throwing pieces of her [kœri]wurst into the air to catch with her mouth. Her throws are getting higher and higher until one piece hits the ceiling and gets stuck there. Oopsie.

Now, she wants to work out for how long the [kœri]wurst can be airborne without getting stuck to the ceiling. For simplicity, Conni assumes that she throws the [kœri]wurst upwards from a height of 1 m (the height of her hand) and that it is airborne until she catches it with her mouth, also at a height of 1 m. If a [kœri]wurst takes t seconds to reach the apex of its trajectory, then the height difference in metres between its release point and the apex is given by

$$s(t) = \frac{1}{2}gt^2.$$

Assume that the gravitational constant g is $9.81\,\mathrm{m/s^2}$ and ignore atmospheric drag.

Also note that by symmetry the [kœri]wurst spends the same amount of time moving upwards as it does moving downwards. Given the height of the ceiling, how long can the [kœri]wurst stay in the air?

The [kœri]wurst will only get stuck to the ceiling if it still has positive vertical velocity when reaching the height of the ceiling. If the apex of the [kœri]wurst's trajectory is exactly at the height of the ceiling, then it will just graze the ceiling without getting stuck.

Input

The input consists of:

- One line with an integer h ($2 \leq h \leq 10^{15}$), the height of the ceiling in metres.

Output

Output the maximum time in seconds that a [kœri]wurst can be airborne without getting stuck to the ceiling.

Your answer should have an absolute or relative error of at most 10^{-6}.

Sample Input 1	Sample Output 1
6	2.0192751

Sample Input 2	Sample Output 2
5033921	2026.1129