COCI '09 Contest 6 #3 Dosadan

Time limit: 0.6s **Memory limit:** 32M

Mirko received a message from his friend Slavko. Slavko, being a world-class cryptologist, likes to encrypt messages he sends to Mirko. This time, he decided to use One Time Pad encryption. OTP is impenetrable if used correctly, and Slavko knows this. He however, doesn't want Mirko to bang his head on an impossible task, so he sent a few hints along with his message.

Mirko knows that Slavko's original plaintext contained **only** small letters of the English alphabet (`a` - `z`), full stop `.` and space ` ` (ASCII 32_{10}). Also, he knows that Slavko used only digits `0` to `9` as his key. After much thought, he realized he can determine locations of all spaces and full stops in the plaintext. He now asked you to write a program that will do so automatically.

From his previous dealings with Slavko, Mirko knows how OTP encryption works. Let's look at a simple example. Suppose you want to encode the string `abc efg` using the key `0120123`.

Start	ASCII Hexadecimal	Encrypted Message
`abc efg`	`61 62 63 20 65 66 67`	`51 53 51 10 54 54 54`
`0120123`	`30 31 32 30 31 32 33`	

First, you transform both the key and plaintext into hexadecimal numbers using ASCII encoding. Then you align them and perform the XOR operation on each pair. The resulting sequence is the encrypted message.

Input Specification

The first line of input contains one integer N $(1 \le N \le 1\,000)$, the number of characters in the encrypted message.

The next line contains N integers, written in hexadecimal, larger than or equal to 0_{10} and smaller than or equal to 127_{10}, the encrypted message.

Output Specification

The first and only line of output should contain N characters, each representing one character in the plaintext. If the i-th character of plaintext is a letter, the i-th character of output should be a dash `-`, if not, you should output a full stop `.`.

Sample Input 1

```
7
51 53 51 10 54 54 54
```

Sample Output 1

```
---.---
```

Sample Input 2

```
7
53 53 51 54 54 51 10
```

Sample Output 2

```
------.
```