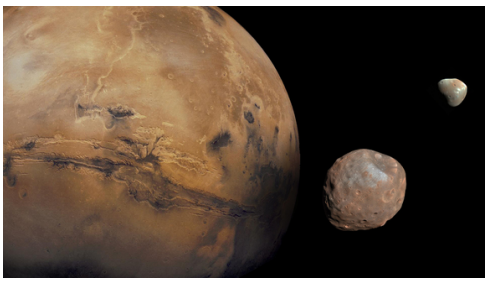
Problem J: Jumbled Packets

Time limit per pass: 1 second



Mars and its two moons Phobos and Deimos, Image by NASA's Goddard Space Flight Center / JPL-Caltech / GSFC / Univ. of Arizona on NASA

Due to further and further advancements in technology and space exploration, humanity is sending probes to more planets and moons than ever before. One of these probes is currently residing on Phobos, one of the two Martian moons. This probe transmits binary data back to Earth but it can only do so in short time frames every few hours. Its fast orbit combined with its small size and close proximity to Mars means that Mars blocks radio waves most of the time by physically being in between the probe and Earth. During one time frame of data transmission, one large packet of n binary bits is transmitted, exactly the amount possible within the time frame.

The probe has already been deployed for a few years, so mechanical problems have started to pile up. A recent and very problematic change has been the unreliability of the onboard clock, likely caused by the large and frequent temperature fluctuations ranging from about $-112°$ to $-4°$ Celsius. As a result, the probe has been transmitting data packets at incorrect times, when radio waves cannot reach Earth. To mitigate this, the probe is now sending its data packets multiple times in succession.

Even though this has led to Earth receiving n bits of data, researchers are not able to determine the start and end of the data packet, making the data useless. The received data consists of a possibly empty suffix, followed by a prefix of the sent data, totalling to exactly n bits. For example, if the original packet was "`00101001`", the received data might be "`00100101`" (the colours are only for visual reasons).

You are to write software to encode the data before transmission into a message of the same length, such that you can decode it after receiving the mangled packet. Luckily, the engineering department has managed to improve the design of the receiving antenna, increasing the signal-to-noise ratio so you are now able to use an alphabet of three distinguishable symbols for radio transmission instead of only two.

Your program will be run twice for each test case. In the first pass, your program will be given a binary string to encode. In the second pass, your program will be given your encoded string from the first pass, which may have been modified as explained above. Decode this string to restore the input from the first pass.

A testing tool is provided to help develop your solution.

Input

The input consists of:

- One line with either "`Encode`" or "`Decode`", depending on the action you are to perform.
- One line with an integer n ($1 \leq n \leq 10^5$), the size of the data packet.
- One line with a string s of n characters, the data packet.

In the encoding pass, this string consists of the symbols '`0`' and '`1`'.

In the decoding pass, this string will be a cyclic rotation of your ternary string. In other words, this string is the string you printed in the encoding pass, potentially modified to reflect how this data packet might have been received on Earth, as described in the statement.

Output

In the encoding pass, output the encoded string of length n using the symbols '0', '1', and '2'.

In the decoding pass, output the original binary string, the data packet from the encoding pass.

Sample Input 1 (Pass 1)

```
Encode
3
001
```

Sample Output 1 (Pass 1)

```
002
```

Sample Input 1 (Pass 2)

```
Decode
3
200
```

Sample Output 1 (Pass 2)

```
001
```

In the first sample, the encoded ternary string "002" was received as "200".

Sample Input 2 (Pass 1)

```
Encode
4
0100
```

Sample Output 2 (Pass 1)

```
2100
```

Sample Input 2 (Pass 2)

```
Decode
4
2100
```

Sample Output 2 (Pass 2)

```
0100
```

In the second sample, the encoded ternary string was received exactly the way it was encoded.