4132 Bad barcodes

You work for a company that develops bar code scanners. A client has recently approached your company to assist in the development of a scanner for a particular type of barcode that he has designed himself. It soon became clear that this particular barcode is not incredibly robust to errors, but since you are not the one playing golf with the client, there is not much you can do about it.

Like most linear bar codes, this code consists of a sequence of alternating black and white bars. A sample of this code, encoding the value 602206, looks like this:

Figure 1: Sample bar code, with a value of 602206.

The sequence of bars can be partitioned into three parts: a *start code*, the *body*, and the *stop code*. Observe that there are actually four kinds of bars in the sample shown in Figure 1: *narrow black* bars, *wide black* bars, *narrow white* bars and *wide white* bars. The wide bars are exactly twice as wide as the narrow bars. The white bars and the black bars alternate strictly, so you can never have two consecutive narrow black bars, which might have been mistaken for a wide black bar.

The function of the alternation of black and white is thus merely to allow us to identify the individual bars. In fact, you can reduce the four types of bar to just two types: *narrow* (N) and *wide* (W). A sequence of 5 consecutive bars can be mapped to the integers 0–9 using the following table:

Digit	Sequence
0	NNWWN
1	WNNNW
2	NWNNW
3	WWNNN
4	NNWNW
5	WNWNN
6	NWWNN
7	NNNWW
8	WNNWN
9	NWNWN

Similarly, the *start code* is encoded as NNNN, and the *stop code* is encoded as WNN. You can easily spot the start and stop codes in the sample barcode above, but before you can decode the *body* of the code, you first have to de-interleave the bars belonging to the body.

This particular bar code system always contains an even number of digits (and thus also bars) in the body. This allows us to interleave the bars of each pair of digits to effectively use both the white and black bars to encode information. For example, the digits 6 and 0 have encodings NWWNN and NNWWN, respectively. If these two encodings are interleaved, we end up with NNWNWWNWNN. Formally, if $b_{n,m}$ represents the m-th bar of the n-th digit of the number being encoded, the interleaving produces

$$b_{n,0}b_{n+1,0}b_{n,1}b_{n+1,1}b_{n,2}b_{n+1,2}\cdots b_{n,4}b_{n+1,4} \qquad b_{n+2,0}b_{n+3,0}b_{n+2,1}b_{n+3,1}\cdots$$

Continuing the example, if we prefix our interleaved sequence (corresponding to '60') with the start code, we get NNNN NNWNWWNWNN — this sequence matches the first 14 bars of Figure 1. Subsequent pairs of digits, such as 22 (or 06) in the sample above, are interleaved in the same way.

That takes care of the specification of the bar code. The bar code reader that your company has developed scans the code using a laser. The laser scans left-to-right across the bar code at a fixed rate, sampling the the output of a photodetector at fixed intervals. The photodetector produces a '1' bit if no reflected light is detected (over a black bar), and a '0' otherwise (over a white bar). Given the sample shown in Figure 1, the bar code scanner produces the output sequence

`10101011011001001010110010101100101001100110101101`.

Note that a narrow bar is represented by a single bit, and a wide bar by a pair of bits. In a perfect universe, it would be a simple matter of reversing the steps above to recover the value of the body of the code represented by this bit sequence. Unfortunately, the bar code scanner occasionally inverts (i.e., $0 \rightarrow 1$, or $1 \rightarrow 0$) the value of at most one bit in the sequence, which sometimes results in a sequence that is decodable, but which produces the wrong value when decoded.

To compensate for this, the quick fix was to require that all the values encoded in the body must be palindromes, i.e., the second half of the sequence of decimal digits must be the reverse of those in the first half, just like the value 602206.

Your task is to take the sequence of bits produced by the scanner, and to decode them to retrieve the value represented by the bar code.

Input

Your input will consist of an arbitrary number of records, one record per line, with each record representing a sequence of bits produced by the scanner.

The length of each input record will be limited to a maximum of 200 bits.

Output

Your output will be a single integer value for each input record. This value must be the value of the body of the bar code sequence.

Some input sequences may contain one single bit inversion, which your program must be able to correct.

Sample Input

```
10101101001010110011001100101010100110101011001101
00101101001010110011001100101010100110101011001101
10101101101010110011001100101010100110101011001101
```

Sample Output

```
123321
123321
123321
```