Problem C
Leaky Cryptography
Input: C.txt

The ACM ICPC judges are very careful about not leaking their problems, and all communications are encrypted. However, one does sometimes make mistakes, like using too weak an encryption scheme. Here is an example of that.

The encryption chosen was very simple: encrypt each chunk of the input by flipping some bits according to a shared key. To provide reasonable security, the size of both chunk and key is 32 bits.

That is, suppose the input was a sequence of m 32-bit integers.

$$N_1 \quad N_2 \quad N_3 \ \ldots \ N_m$$

After encoding with the key K it becomes the following sequence of m 32-bit integers.

$$(N_1 \wedge K)\ (N_2 \wedge K)\ (N_3 \wedge K) \ldots (N_m \wedge K)$$

where $(a \wedge b)$ is the *bitwise exclusive or* of a and b.

Exclusive or is the logical operator which is 1 when only one of its operands is 1, and 0 otherwise. Here is its definition for 1-bit integers.

$$\begin{array}{ll} 0 \oplus 0 = 0 & 0 \oplus 1 = 1 \\ 1 \oplus 0 = 1 & 1 \oplus 1 = 0 \end{array}$$

As you can see, it is identical to addition modulo 2. For two 32-bit integers a and b, their bitwise exclusive or $a \wedge b$ is defined as follows, using their binary representations, composed of 0's and 1's.

$$a \wedge b = a_{31} \ldots a_1 a_0 \wedge b_{31} \ldots b_1 b_0 = c_{31} \ldots c_1 c_0$$

where

$$c_i = a_i \oplus b_i \ (i = 0, 1, \cdots, 31).$$

For instance, using binary notation, 11010110 ^ 01010101 = 10100011, or using hexadecimal, `d6` ^ `55` = `a3`.

Since this kind of encryption is notoriously weak to statistical attacks, the message has to be compressed in advance, so that it has no statistical regularity. We suppose that $N_1\ N_2 \ldots N_m$ is already in compressed form.

However, the trouble is that the compression algorithm itself introduces some form of regularity: after every 8 integers of compressed data, it inserts a checksum, the sum of these integers. That is, in the above input, $N_9 = \sum_{i=1}^{8} N_i = N_1 + \cdots + N_8$, where additions are modulo 2^{32}.

Luckily, you could intercept a communication between the judges. Maybe it contains a problem for the finals!

As you are very clever, you have certainly seen that you can easily find the lowest bit of the key, denoted by K_0. On the one hand, if $K_0 = 1$, then after encoding, the lowest bit of $\sum_{i=1}^{8} N_i \wedge K$ is unchanged, as K_0 is added an even number of times, but the lowest bit of $N_9 \wedge K$ is changed, so they shall differ. On the other hand, if $K_0 = 0$, then after encoding, the lowest bit of $\sum_{i=1}^{8} N_i \wedge K$ shall still be identical to the lowest bit of $N_9 \wedge K$, as they do not change. For instance, if the lowest bits after encoding are `1 1 1 1 1 1 1 1 1` then K_0 must be 1, but if they are `1 1 1 1 1 1 1 0 1` then K_0 must be 0.

So far, so good. Can you do better?

You should find the key used for encoding.

Input

The input starts with a line containing only a positive integer S, indicating the number of datasets in the input. S is no more than 1000.

It is followed by S datasets. Each dataset is composed of nine 32-bit integers corresponding to the first nine chunks of a communication. They are written in hexadecimal notation, using digits '`0`' to '`9`' and lowercase letters '`a`' to '`f`', and with no leading zeros. They are separated by a space or a newline. Each dataset is ended by a newline.

Output

For each dataset you should output the key used for encoding. Each key shall appear alone on its line, and be written in hexadecimal notation, using digits '`0`' to '`9`' and lowercase letters '`a`' to '`f`', and with no leading zeros.

Sample Input

```
8
1 1 1 1 1 1 1 1 8
3 2 3 2 3 2 3 2 6
3 4 4 7 7 b a 2 2e
e1 13 ce 28 ca 6 ab 46 a6d
b08 49e2 6128 f27 8cf2 bc50 7380 7fe1 723b
4eba eb4 a352 fd14 6ac1 eed1 dd06 bb83 392bc
ef593c08 847e522f 74c02b9c 26f3a4e1 e2720a01 6fe66007
7a4e96ad 6ee5cef6 3853cd88
60202fb8 757d6d66 9c3a9525 fbcd7983 82b9571c ddc54bab 853e52da
22047c88 e5524401
```

Output for the Sample Input

```
0
2
6
1c6
4924afc7
ffff95c5
546991d
901c4a16
```