BitBitJump

Time limit:	3 seconds
Memory limit:	1024 megabytes

BitBitJump is a one instruction set computer. Thus, it has only one instruction: `bbj a, b, c`, which copies an a-th bit of memory to the b-th bit of memory and then jumps to address c.

Let's consider a 16-bit BitBitJump computer. It has 2^{16} bits of memory organized in 2^{12} 16-bit words. Words are counted from 0, and bits in a word are counted from the least significant (0-th) bit to the most significant (15-th) bit.

This computer has a single instruction pointer register (IP), and execution starts with IP = 0. If the current IP $\geq 2^{12} - 2$, the computer stops. Otherwise, it uses the IP-th word as a, the (IP + 1)-th word as b, the (IP + 2)-th word as c, and performs the `bbj a, b, c` instruction: copies the $(a \,\&\, 15)$-th bit of the $(a \gg 4)$-th word to the $(b \,\&\, 15)$-th bit of the $(b \gg 4)$-th word, and sets IP = c. Here, '&' represents bitwise AND, and '$\gg$' represents bitwise shift right operation. Notice that the value of c is read from memory after the bit copy, so if the instruction modified its own c, the new value will be used for IP.

For example, the `bbj 32, 35, 5` instruction placed at the memory start will be executed as follows:

1. $a = 32$ and $b = 35$ are read from the memory.
2. The 0-th bit of the 2-nd word (its value is $5 \,\&\, 1 = 1$) will be copied to the 3-rd bit of the same word, so the 2-nd word will have the value of $5 + 2^3 = 13$.
3. Then $c = 13$ is read from memory, and IP is set to 13.

Let's call the $(2^{12} - 1)$-th word ($2^{16} - 16 \ldots 2^{16} - 1$-th bits of memory) an *IO-word*. An *x-comparator* is a program which checks whether the value of the IO-word is equal to x. It should stop after execution of no more than 2^{12} instructions, leaving the lowest bit of the IO-word equal to 1 if the original value of the IO-word was equal to x, and 0 otherwise.

Write a program that generates an x-comparator for the given value of x.

Input

The input contains a single decimal integer x ($0 \leq x < 2^{16}$) — the value for which to build the x-comparator.

Output

The output should contain the x-comparator program dump. Dump consists of values for the first n words of the memory ($1 \leq n \leq 2^{12} - 1$). All other words, except the IO-word, are filled with zeroes.

For each of the n words, output its value as a four-character hexadecimal number. Values should be delimited by space or new line characters.

Example

standard input
0

standard output
fff0 0026 0003 fff1 0056 0006 fff2 0086 0009 fff3 00b6 000c fff4 00e6 000f fff5 0116 0012 fff6 0146 0015 fff7 0176 0018 fff8 01a6 001b fff9 01d6 001e fffa 0206 0021 fffb 0236 0024 fffc 0266 0027 fffd 0296 002a fffe 02c6 002d ffff 02f6 0030 0004 fff0 0fff 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff 0000 fff0 0fff

Note

A dump in the sample output contains a 0-comparator. It consists of the following blocks:

- 16 instructions: the i-th of them, counting from 0, copies the i-th bit of the input word to the 6-th bit of its own c. If the copied bit is zero, it will proceed to the next instruction; otherwise, the next instruction number will be increased by 64.
- The following instruction copies the 4-th bit of the 0-th word (value 1) to the 0-th bit of the IO-word and jumps to the stop address.
- 16 unused words filled with 0.
- 16 equal instructions (starting from word 67). Each of them copies the 0-th bit of the 0-th word (value 0) to the 0-th bit of the IO-word and jumps to the stop address.