Basic Matrix Recurrence Practice

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	4 megabytes

Define the column vector:

$$v_n = (a_n, b_n, c_n, d_n)^T$$

Initial vectors:

$$v_0 = (1, 2, 3, 4)^T, \quad v_1 = (5, 6, 7, 8)^T$$

For all $n \geq 1$:

$$v_{n+1} \equiv P v_n + Q\big(v_{n-1} \circ v_{n-1}\big) \pmod{61}$$

P, Q are 4×4 constant matrices. The operator $\circ$ denotes Hadamard (element-wise) product.

The following conventions are used throughout this statement:

$$x \circ y = (x_1 y_1, x_2 y_2, x_3 y_3, x_4 y_4)^T, \qquad (Mx)_r = \left(\sum_{c=1}^{4} M_{r,c} x_c\right) \bmod 61.$$

Hence,

$$v_{n-1} \circ v_{n-1} = (a_{n-1}^2, b_{n-1}^2, c_{n-1}^2, d_{n-1}^2)^T.$$

All additions and multiplications in the recurrence are performed modulo 61.

To avoid massive I/O overhead, queries are processed online and only one final checksum is printed.

Maintain a 64-bit unsigned integer `rnd_seed` (initialized by input `seed`) and generate random numbers by:

```
uint64_t next_rand() {
    rnd_seed ^= rnd_seed << 13;
    rnd_seed ^= rnd_seed >> 7;
    rnd_seed ^= rnd_seed << 17;
    return rnd_seed;
}
```

Before query 1, set `last_ans` $= 0$ and `final_hash` $= 0$. For the i-th query ($1 \leq i \leq q$), do the following:

1. Generate:
$$n_i = (\texttt{next\_rand()} \oplus \texttt{last\_ans}) \bmod 10^{18}$$
2. Compute $v_{n_i} = (a, b, c, d)^T$, then compress:
$$\texttt{current\_ans} = a \mid (b \ll 6) \mid (c \ll 12) \mid (d \ll 18)$$
3. Update:
$$\texttt{last\_ans} \leftarrow \texttt{current\_ans}$$
$$\texttt{final\_hash} \leftarrow \texttt{final\_hash} \oplus (\texttt{current\_ans} \cdot i)$$

Bit operations are standard unsigned-integer operations: $\oplus$ is bitwise XOR, $\ll$ is left shift, and $\mid$ is bitwise OR.

Input

The first 4 lines contain matrix P, each line has 4 integers in $[0, 60]$.

The next 4 lines contain matrix Q, each line has 4 integers in $[0, 60]$.

The last line contains two integers `q` and `seed`: $1 \leq q \leq 10^7$ and $0 \leq \texttt{seed} < 2^{64}$.

In official tests, elements of P and Q are generated independently and uniformly at random in $[0, 60]$.The number of official test files does not exceed 30.

Output

Print a single 64-bit unsigned integer, which is the final value of `final_hash` after all q queries are processed.

Examples

standard input	standard output
1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 1 1 0 0 1 5 7	42019442
6 5 4 3 2 6 5 4 3 2 6 5 4 3 2 6 6 6 5 5 4 4 3 3 2 2 1 1 5 1 4 2 5 1844674407370955161	67023929

Note

Strict Memory Limit Warning: The memory limit for this problem is extremely tight.