Codeforces Round #230 (Div. 2)

A. Nineteen

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Alice likes word "nineteen" very much. She has a string *S* and wants the string to contain as many such words as possible. For that reason she can rearrange the letters of the string.

For example, if she has string "xiineteenppnnnewtnee", she can get string "x**nineteen**pp**nineteen**w", containing (the occurrences marked) two such words. More formally, word "nineteen" occurs in the string the number of times you can read it starting from some letter of the string. Of course, you shouldn't skip letters.

Help her to find the maximum number of "nineteen"s that she can get in her string.

Input
The first line contains a non-empty string *S*, consisting only of lowercase English letters. The length of string *S* doesn't exceed 100.

Output
Print a single integer — the maximum number of "nineteen"s that she can get in her string.

Examples

input
nniinneetteeeenn
output
2

input
nneteenabcnneteenabcnneteenabcnneteenabcnneteenabcii
output
2

input
nineteenineteen
output
2

B. Three matrices

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Chubby Yang is studying linear equations right now. He came up with a nice problem. In the problem you are given an $n \times n$ matrix W, consisting of integers, and you should find two $n \times n$ matrices A and B, all the following conditions must hold:

- $A_{ij} = A_{ji}$, for all i, j $(1 \le i, j \le n)$;
- $B_{ij} = -B_{ji}$, for all i, j $(1 \le i, j \le n)$;
- $W_{ij} = A_{ij} + B_{ij}$, for all i, j $(1 \le i, j \le n)$.

Can you solve the problem?

Input

The first line contains an integer n $(1 \le n \le 170)$. Each of the following n lines contains n integers. The j-th integer in the i-th line is W_{ij} $(0 \le |W_{ij}| < 1717)$.

Output

The first n lines must contain matrix A. The next n lines must contain matrix B. Print the matrices in the format equal to format of matrix W in input. It is guaranteed that the answer exists. If there are multiple answers, you are allowed to print any of them.

The answer will be considered correct if the absolute or relative error doesn't exceed 10^{-4}.

Examples

input
2 1 4 3 2
output
1.00000000 3.50000000 3.50000000 2.00000000 0.00000000 0.50000000 -0.50000000 0.00000000

input
3 1 2 3 4 5 6 7 8 9
output
1.00000000 3.00000000 5.00000000 3.00000000 5.00000000 7.00000000 5.00000000 7.00000000 9.00000000 0.00000000 -1.00000000 -2.00000000 1.00000000 0.00000000 -1.00000000 2.00000000 1.00000000 0.00000000

C. Blocked Points

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Imagine you have an infinite 2D plane with Cartesian coordinate system. Some of the integral points are blocked, and others are not. Two integral points A and B on the plane are 4-connected if and only if:

- the Euclidean distance between A and B is one unit and neither A nor B is blocked;
- or there is some integral point C, such that A is 4-connected with C, and C is 4-connected with B.

Let's assume that the plane doesn't contain blocked points. Consider all the integral points of the plane whose Euclidean distance from the origin is no more than n, we'll name these points special. Chubby Yang wants to get the following property: no special point is 4-connected to some non-special point. To get the property she can pick some integral points of the plane and make them blocked. What is the minimum number of points she needs to pick?

Input

The first line contains an integer n ($0 \le n \le 4 \cdot 10^7$).

Output

Print a single integer — the minimum number of points that should be blocked.

Examples

input
1
output
4

input
2
output
8

input
3
output
16

D. Tower of Hanoi

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Tower of Hanoi is a well-known mathematical puzzle. It consists of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

With three disks, the puzzle can be solved in seven moves. The minimum number of moves required to solve a Tower of Hanoi puzzle is $2^n - 1$, where n is the number of disks. (c) Wikipedia.

SmallY's puzzle is very similar to the famous Tower of Hanoi. In the Tower of Hanoi puzzle you need to solve a puzzle in minimum number of moves, in SmallY's puzzle each move costs some money and you need to solve the same puzzle but for minimal cost. At the beginning of SmallY's puzzle all n disks are on the first rod. Moving a disk from rod i to rod j $(1 \le i, j \le 3)$ costs t_{ij} units of money. The goal of the puzzle is to move all the disks to the third rod.

In the problem you are given matrix t and an integer n. You need to count the minimal cost of solving SmallY's puzzle, consisting of n disks.

Input

Each of the first three lines contains three integers — matrix t. The j-th integer in the i-th line is t_{ij} $(1 \le t_{ij} \le 10000;\ i \ne j)$. The following line contains a single integer n $(1 \le n \le 40)$ — the number of disks.

It is guaranteed that for all i $(1 \le i \le 3)$, $t_{ii} = 0$.

Output

Print a single integer — the minimum cost of solving SmallY's puzzle.

Examples

input
0 1 1 1 0 1 1 1 0 3
output
7

input
0 2 2 1 0 100 1 2 0 3
output
19

input
0 2 1 1 0 100 1 2 0 5
output
87

E. Yet Another Number Sequence

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Everyone knows what the Fibonacci sequence is. This sequence can be defined by the recurrence relation:

$$F_1 = 1, F_2 = 2, F_i = F_{i-1} + F_{i-2} \ (i > 2).$$

We'll define a new number sequence $A_i(k)$ by the formula:

$$A_i(k) = F_i \times i^k \ (i \geq 1).$$

In this problem, your task is to calculate the following sum: $A_1(k) + A_2(k) + \ldots + A_n(k)$. The answer can be very large, so print it modulo 1000000007 $(10^9 + 7)$.

Input

The first line contains two space-separated integers n, k $(1 \leq n \leq 10^{17}; 1 \leq k \leq 40)$.

Output

Print a single integer — the sum of the first n elements of the sequence $A_i(k)$ modulo 1000000007 $(10^9 + 7)$.

Examples

input
1 1
output
1

input
4 1
output
34

input
5 2
output
316

input
7 4
output
73825