A. Crystal Ball Sequence

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

Crystal ball sequence on hexagonal lattice is defined as follows: n-th element is the number of lattice points inside a hexagon with $(n+1)$ points on each side. The formula is $H_n = 3 \cdot n \cdot (n+1) + 1$. You are given n; calculate n-th element of the sequence.

Input

The only line of input contains an integer n $(0 \le n \le 9)$.

Output

Output the n-th element of crystal ball sequence.

Sample test(s)

input

```
1
```

output

```
7
```

input

```
3
```

output

```
37
```

B. Hexakosioihexekontahexaphobia

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

This (unpronounceable) word means simply fear of number 666.

You are given a string of digits. Check whether it is scary for a person suffering from this phobia, i.e., whether it contains number 666 as a substring.

Input

The input will consist of a single string p. The string contains between 1 and 100 digits ('0'-'9'), inclusive. The string doesn't contain any other characters except digits.

Output

Output "`YES`" if given string contains number 666, and "`NO`" otherwise (quotes for clarity only).

Sample test(s)

input

```
123098
```

output

```
NO
```

input

```
16660
```

output

```
YES
```

input

```
1606061
```

output

```
NO
```

Note

Note that 666 must be a contiguous substring of p, not a subsequence (see sample 3).

C. Eval

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

You are given a simple arithmetic expression of the form $a?b$, where a and b are integer constants, and $?$ can be one of the following operations: '+' (addition), '-' (subtraction), '*' (multiplication), '/' (integer division) or '%' (modulo operation).

Output the result of evaluation of this expression.

Input

The input is a single line containing an expression $a?b$. Here a and b are integers between 1 and 999, inclusive; $?$ is an operation character: '+', '-' (ASCII code 45), '*', '/' or '%'.

Output

Output a single integer — the result of evaluation of this expression.

Sample test(s)

input

```
123+456
```

output

```
579
```

input

```
192/5
```

output

```
38
```

input

```
945%19
```

output

```
14
```

D. Caesar Cipher

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

Caesar cipher is one of the simplest encryption techniques. To transform the original message into encrypted one using key k, one has to replace each letter with a letter which is k positions later in the alphabet (if this takes the position beyond Z, the rest of it is counted from the start of the alphabet). In a more formal way, if letters of the alphabet are enumerated starting with 0, the result of encryption for character x will be $(x+k) \mod 26$ (26 is the number of letters in the Latin alphabet).

You are given the original message and the encryption key k. Output the result of encryption.

Input

The first line of input contains an integer k ($0 \le k \le 25$) — the encryption key.

The second line contains the original message — a sequence of uppercase Latin letters ('A'-'Z'). The length of the message is from 1 to 10, inclusive.

Output

Output the result of encryption.

Sample test(s)

input

```
5
CODEFORCES
```

output

```
HTIJKTWHJX
```

input

```
13
FALSERULEZ
```

output

```
SNYFREHYRM
```

E. Chessboard

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Chessboard is a board of $n \times n$ squares arranged in two alternating colors (black and white). Top left square is white. You are given board size n. Output an image of a chessboard, with black and white squares marked with '#' and '.' characters, respectively.

Input

The only line of input contains an integer n $(1 \le n \le 9)$.

Output

Output an image of $n \times n$ chessboard.

Sample test(s)

input

```
4
```

output

```
.#.#
#.#.
.#.#
#.#.
```

F. Pairwise Sums

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

You are given an array of n integers. For each element output the sum of itself and the previous element. For the first element, output the sum of the first and the last elements of the array.

Input

The input consists of a single line of space-separated integers. The first number is n $(2 \leq n \leq 50)$ — the size of the array. The following n numbers are the elements of the array $(1 \leq a_i \leq 1000)$.

Output

Output the sums $a_1 + a_n$, $a_2 + a_1$, ..., $a_n + a_{n - 1}$, separated with spaces.

Sample test(s)

input

```
4 1 2 3 4
```

output

```
5 3 5 7
```

input

```
5 5 46 372 81 9
```

output

```
14 51 418 453 90
```

G. Hamming Distance

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

Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. You are given two strings; calculate the distance between them.

Input

The input consists of two lines. Each line contains a string of characters 'A'-'Z' between 1 and 100 characters, inclusive. The strings have equal length.

Output

Output Hamming distance between the strings.

Sample test(s)

input

```
CODECHEF
TOPCODER
```

output

```
6
```

input

```
HAMMING
DISTANC
```

output

```
6
```

H. Array Sorting

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Sorting arrays is traditionally associated with high-level languages. How hard can it be in FALSE? Sort the given array in non-descending order.

Input

The input consists of a single line of space-separated integers. The first number is n $(1 \le n \le 10)$ — the size of the array. The following n numbers are the elements of the array $(1 \le a_i \le 100)$.

Output

Output space-separated elements of the sorted array.

Sample test(s)

input

```
3 3 1 2
```

output

```
1 2 3
```

input

```
7 12 2 3 44 5 60 2
```

output

```
2 2 3 5 12 44 60
```