Information

Memory limit

The limit is 512 MiB for each problem.

Source code limit

The size of each solution source code can't exceed 256 KiB.

Submissions limit

You can submit at most 50 solutions for each problem.

You can submit a solution to each task at most once per 30 seconds. This restriction does not apply in the last 15 minutes of the contest round.

Scoring

Each problem consists of several subtasks. The subtask score is awarded if all tests in the subtask are passed.

The number of points scored for the problem is the total number of points scored on each of its subtasks. The score for the subtask is the maximum number of points earned for this subtask among all the solutions submitted.

Feedback

To get feedback for your solution, go to "Runs" tab in PCMS2 Web Client and use "View Feedback" link. In each problem of the contest you will see the score for each subtask, or the verdict for the first failed test.

Scoreboard

The contestants' scoreboard is available during the contest. Use "Monitor" link in PCMS2 Web Client to access the scoreboard. The standings provided in PCMS2 Web Client are not final.

Interactive problems

Print new line character after each line.

After each action your program must flush standard output.

If you use "`writeln`" in Pascal, "`cout << ... << endl`" in C++, "`System.out.println`" in Java, "`print`" in Python, "`Console.WriteLine`" in C#, standard output is automatically flushed, no additional action is required. If you are using other way to output data, you should flush standard output. Note that even if you flush output, you absolutely must print new line character. To flush output you can use "`fflush(stdout)`" in C and C++, "`flush(output)`" in Pascal, "`System.out.flush()`" in Java, "`sys.stdout.flush()`" in Python, "`Console.Out.Flush()`" in C#. Note that you must explicitly "`flush`" output in Borland Delphi.

Common errors in interactive problems:

- "`Wrong Answer`" means that your final answer or intermediate actions are incorrect, or that your output is formatted incorrectly.
- "`Idleness Limit Exceeded`" means that your program is expecting some input while there is none. For example,
  - your program is waiting for input, but it must output more information, or it must terminate;
  - your program is waiting for input, but it didn't flush output or didn't print new line character, so the jury's program doesn't know it must act;
- "`Runtime Error`" usually is not specific to interactivity, it occurs because of some general bugs.

Problem A. A + B

Time limit: 2 seconds

You are given two real numbers a and b. Write a program to calculate $a + b$.

Input

The first line contains two real numbers — a and b ($-1\,000 \le a, b \le 1\,000$).

Output

Print the value of $a+b$ on the first line of the output. The answer is considered to be correct if its absolute or relative error does not exceed 10^{-4}.

Scoring

Subtask	Score	Constraints
1	50	a and b are integers
2	50	—

Examples

standard input	standard output
1.1 2.2	3.3
1 -1	0

Problem B. Zero-complexity Transposition

Time limit: 2 seconds

You are given a sequence of integer numbers. Zero-complexity transposition of the sequence is the reverse of this sequence. Your task is to write a program that prints zero-complexity transposition of the given sequence.

Input

The first line of the input contains one integer n — length of the sequence ($0 < n \leq 10^4$). The second line contains n integer numbers — $a_1, a_2, \ldots, a_n$ ($-10^15 \leq a_i \leq 10^15$).

Output

On the first line of the output print the sequence in the reverse order.

Scoring

Subtask	Points	Constraints
1	50	$-1000 \leq a_i \leq 1000$
2	50	—

Examples

standard input	standard output
3 1 2 3	3 2 1
5 -3 4 6 -8 9	9 -8 6 4 -3

Problem C. Big Array

Time limit: 2 seconds

You are required to calculate total sum of integer sequence consisting of n elements.

This problem has very big input data. If you are using C++ and your solution times out, try one of the following optimizations:

1. For `std::cin` from `iostream` call `ios::sync_with_stdio(false);` and `cin.tie(NULL);` in the very beginning of your program.
2. Use Linux version of g++ compiler.
3. In some compilers `scanf("%lld", &x);` can work faster than `std::cin`.

Input

First line contains an integer n — the length of the sequence a ($1 \le n \le 5 \cdot 10^6$).

The next line consists of integers $a_1, a_2, \ldots, a_n$ ($-10^{12} \le a_i \le 10^{12}$).

Output

Print single integer: the total sum of sequence elements.

Scoring

Subtask	Score	Constraints
1	24	$1 \le n \le 100$, $-5000 \le a_i \le 5000$
2	23	$1 \le n \le 10^5$
3	33	$1 \le n \le 10^6$
4	20	—

Feedback

In this problem the time taken by your program is shown for every test.

Example

standard input	standard output
4 2 -5 6 10	13

Problem D. Guess the Number

Time limit: 2 seconds

This is an interactive problem. Your program will interact with the jury program using standard input and output.

The jury's program chose a number from 1 to n. Your program's goal is to guess the number in at most 30 guesses. Your program submits a guess, then the jury's program replies if their chosen number is greater than, less than, or equal to your guess.

Interaction Protocol

First, read the number n from the standard input ($1 \le n \le 10^9$). Then the interaction goes as follows: for each guess print a line containing a single integer — your guess.

After that you should read one integer: the result of the guess. Three messages are possible:

- «1» — the chosen number is greater than your guess;
- «-1» — the chosen number is less than your guess;
- «0» — your guess is correct. After your program reads 0, it should stop.

Please note that you need to print a line break after each guess. You can read more details about interactive problems on the first page.

Scoring

Subtask	Points	Constraints
1	50	$1 \le n \le 30$
2	50	$1 \le n \le 10^9$

Example

standard input	standard output
5 -1 1 0	 3 1 2

Note

The example input/output have extra empty lines, so it's easier to see which guesses correspond to which jury's responses. You need to output line breaks in a real interaction, but you don't need to print empty lines.

Problem E. Error Correction

Time limit: 1 second

This is a "run-twice" problem, your program will be run twice for each test.

During the first run, your program will receive the string x, consisting of zeros and ones with length at most n. The program now has to print a string y, consisting of zeros and ones with length at most m. The number m is unknown to your program and depends on the current subtask. See the corresponding values and subtasks below. If your program outputs a string that's longer than m, you will receive a "`Wrong answer`" outcome.

Between the two runs, the judges' program will flip at most one character in string y, changing a zero into a one, or a one into a zero. Let's call the new string z.

During the second run, your program will be given z as the input. Your program has to restore the initial string x and print it.

Input

During the first run, the first line contains the integer 1. The second line contains the string x consisting of n zeros or ones ($10 \le n \le 100\,000$).

During the second run, the first line contains the integer 2. The second line contains the string z consisting of at most m zeros or ones ($10 \le m \le 300\,000$). This string is either equal to the string y printed by your program during the first run, or differs from it in exactly one position.

Output

During the first run, print the binary string y which will help you restore x after the change. The length of y must not exceed m.

During the second run, given the string z, restore the initial string x and print it.

Scoring

Subtask	Points	Constraints
1	40	$n = 10$, $m = 30$
2	15	$n = 100\,000$, $m = 300\,000$
3	15	$n = 100\,000$, $m = 200\,000$
4	15	$n = 100\,000$, $m = 101\,000$
5	15	$n = 100\,000$, $m = 100\,017$

Examples

standard input	standard output
1 0000000000	00000000000000000000000000000000

standard input	standard output
2 00000000000000000000000000000000	0000000000

standard input	standard output
1 0000000000	00000000000000000000000000000000

standard input	standard output
2 00000000000010000000000000000	0000000000