Mathematics Department

"A computer is the mathematician's best friend"

μ-Games

April 2025

Rules:

The idea of this event is to gap the bridge between mathematics and programming. When working on these exercises, we hope that the participant will have a better understanding of the underlying mathematical concepts. You will not be required to do many difficult programming tasks. With array manipulation and basic functionality, you should be able to solve all the exercises.

When working on these exercises, you must follow the following rules.

- You are allowed to work in groups of a maximum of 4 people.
- You will have 3 hours to solve the problems.
- For problems, you can use the default maths library of your programming language (for example *import math* in Python). Other libraries like *numpy* cannot be used.
- You cannot look up any computer code that may help you solve the problem.

After 3 hours, the solutions to the exercises will be discussed. To check your own solution, go to the website http://clover.science.uu.nl/dj.

A standard setup of your code could look like this, where *function()* is the method used to solve the problem.

```
# First, we read the size of the testset (if necessary).
n = input()
arr = []

for _ in range(n):
    row = input().split()
    arr.append(row)

def function():
    pass

answer = function()

# Finally, we print the answer of the testset.
print(answer)
```

Problem 1: Pythagorean Triples

Difficulty: ★ ☆ ☆ ☆ ☆
Key words: Number theory

A Pythagorean triplet is a set of 3 numbers $a, b, c \in \mathbb{N}$, with the property that $a^2 + b^2 = c^2$. A pythagorean triplet is called primitive if a, b, c are co-prime, i.e. if they have no common factors. Your task is to find k unique primitive pythagorean triplets, where $n < a, b, c < 2n^2$ for some given n, k.

Input

- A positive integer $N \leq 10^{14}$
- A positive integer $k \leq 10000$

Output

- k lines containing a, b, c separated by a space.

Examples

Input	Output
10	143 24 145
2	119 120 169

Input	Output
1000	1008015 2008 1008017
4	1008007 6024 1008025
	1007991 10040 1008041
	1007967 14056 1008065

Problem 2: Multiplicative Orders

Difficulty: ★ ★ ★ ☆ ☆
Key words: Group theory, Number theory

Given is $N \in \mathbb{N}$. The multiplicative order $o \in \mathbb{Z}_{>0}$ of $x \in \mathbb{Z}/n\mathbb{Z}$ is the least number such that $x^o = 1$.
In this exercise you are given $N \in \mathbb{N}$, $k \in \mathbb{N}$ and k numbers $x_i \in \mathbb{Z}/N\mathbb{Z}$. You must give the multiplicative orders

Input

- A positive integer $N \leq 10^{17}$
- A positive integer $k \leq 1000$
- k natural numbers $0 \leq x_i < N$.

Output

- The multiplicative orders of x_i on separate lines

Examples

Input	Output
108	18
4	18
5	6
7	18
91	
11	

Input	Output
7	6
2	6
2	
4	

Problem 3: Summing Dice

Difficulty: ★ ★ ★ ★ ☆
Key words: modular arithmetic, algebra, probability theory

Given are 2^k dices with $n \leq N$ faces with numbers $0 \leq x_i < N$. We are interested in the sum of the dices modulo N, we call this number S. Given k, N, n, x_i give the number of ways S can be the values $0..N-1$ denoted by S_i.

Input

- A positive integer $k < 15$
- A positive integer $N < 1000$
- A positive integer $n \leq N$
- n natural numbers $0 \leq x_i < N$ representing the numbers on the faces of the dice.

Output

- The numbers S_i each on their own line.

Examples

Input	Output
3	85
6	0
2	85
0	0
4	86
	0

Problem 4: Polynomial Problem

Difficulty: ★ ★ ★ ★ ★
Keywords: Number Theory, Algebra

Part A

In his programming class, Peter has an assignment to calculate the coefficients of a polynomial given the roots of this polynomial. As Peter knows his Vieta formulas, this is not a problem at all for him.

After half an hour, Peter is done and has several polynomials written in his notebook. However, he forgot the roots that were given. Can you help him? In other words: given the coefficients of a polynomial of degree n that has n distinct integer roots, can you find these roots?

Input

- One line containing an integer $0 \leq n \leq 10$, denoting the degree of the polynomial.
- One line containing $n+1$ integers, the i'th of which is the integer $a_{i-1} < 10^{10}$ denoting the coefficient of the term $a_{i-1}x^{i-1}$.

Output

- Output the n roots of the polynomial, space separated, in increasing order.

Examples

Input	Output
1 -1 1	1

Input	Output
3 0 -1 0 1	-1 0 1

Part B

This time, the n distinct roots are not necessarily integers. Can you still find all the n roots up to 6 decimals?

Input

- One line containing an integer $0 \leq n \leq 10$, denoting the degree of the polynomial.
- $n+1$ lines, the i'th line of which contains a float $a_{i-1} < 10^{10}$ denoting the coefficient of the term $a_{i-1}x^{i-1}$.

Output

- Output the n roots of the polynomial, space separated, in increasing order. Each root should have an absolute error of at most 10^{-6}.

Examples

Input	Output
1 -0.5 1	0.4999993

Problem 5: Organic Optimization

Difficulty: ★ ★ ★ ★ ★
Key words: Linear Programming, Optimization, Geometry

Many real-world problems can be stated as an optimization problem under inequality constraints. We are interested in solving the problem:

$$\begin{aligned} \text{maximize} \quad & c^T x, \\ \text{subject to} \quad & Ax \leq b, \quad x_i \geq 0. \end{aligned} \tag{1}$$

In the whole setting, we only consider $x \in \mathbb{R}^2$. Moreover, we can assume that a solution to the problem exists. Your task is to determine which point in the feasible domain ($A\mathbf{x} \leq \mathbf{b}$) gives the maximum value of $\mathbf{c}^T\mathbf{x}$.

Input

- The first line contains two space separated number c_1 c_2 indicating the vector **c**.
- The next line contains an integer n ($3 \leq n \leq 1e5$), the number of rows of A
- The next line contains space separated number b_1 b_2 $\ldots$ b_n indicating the vector **b**.
- The next n lines contain each row of the matrix A.
- Two space-separated indices i j, indicating the number of inequalities, that is, 1 for the first inequality $A_{11}x_1 + A_{12}x_2 \leq b_1$.

Output

- **a** Output 1 if the intersection of the two inequalities represents a solution to our optimization problem, and 0 otherwise.
- **b** Output a possible normalized direction vector in which we can move along the feasible domain such that the objective increases most, starting at the intersection from **a**, to increase the value $\mathbf{c}^T x$ (while remaining feasible). Assume that we are not in the optimum state and that the intersection is feasible. Output as two space-separated floats, both rounded to the 5th decimal.
- **c** Output a value $\mathbf{c}^T\mathbf{x}^*$ that maximizes system (1). Output as two space-separated floats, rounded to the 5th decimal.