ADA - Análisis y Diseño de Algoritmos, 2020-2
Tarea 1: Semanas 1 y 2

Para entregar el viernes 21/domingo 23 de agosto de 2020

Problemas conceptuales a las 23:59 (21 de agosto) por correo electrónico al monitor

Problemas prácticos a las 23:59 (23 de agosto) en la arena de programación

---

Tanto los ejercicios como los problemas deben ser resueltos, pero únicamente las soluciones de los problemas deben ser entregadas. La intención de los ejercicios es entrenarlo para que domine el material del curso; a pesar de que no debe entregar soluciones a los ejercicios, usted es responsable del material cubierto en ellos.

Instrucciones para la entrega

Para esta tarea y todas las tareas futuras, la entrega de soluciones es *individual*. Por favor escriba claramente su nombre, código de estudiante y sección en cada hoja impresa entregada o en cada archivo de código (a modo de comentario). Adicionalmente, agregue la información de fecha y nombres de compañeros con los que colaboró; igualmente cite cualquier fuente de información que utilizó.

¿Cómo describir un algoritmo?

En algunos ejercicios y problemas se pide "dar un algoritmo" para resolver un problema. Una solución debe tomar la forma de un pequeño ensayo (es decir, un par de párrafos). En particular, una solución debe resumir en un párrafo el problema y cuáles son los resultados de la solución. Además, se deben incluir párrafos con la siguiente información:

- una descripción del algoritmo en castellano y, si es útil, pseudo-código;
- por lo menos un diagrama o ejemplo que muestre cómo funciona el algoritmo;
- una demostración de la corrección del algoritmo; y
- un análisis de la complejidad temporal del algoritmo.

Recuerde que su objetivo es comunicar claramente un algoritmo. Las soluciones algorítmicas correctas y descritas *claramente* recibirán alta calificación; soluciones complejas, obtusas o mal presentadas recibirán baja calificación.

Ejercicios

1.1-2 (página 11), 1.2-2, 1.2-3, 1-1 (página 14), 2.1-1, 2.1-2, 2.1-3 (página 22), 2.2-1, 2.2-2 (página 29), 2.3-1 (página 37), 2.3-4, 3.1-1 (página 52), 3.1-4, 3.1-8 (página 53), 3.2-1, 3.2-7, 3-2 columnas O, Ω y Θ (página 61), 3-3 parte *a* (página 62).

Problemas conceptuales

1. Escribir el código de honor del curso.
2. Ejercicios 1, 3 y 5: Orden asintótico y eficiencia algorítmica (Kleinberg & Tardos páginas 67 y 68).
3. Ejercicio 11: *StoogeSort* (Erickson, página 50).
4. Ejercicio 27: *Political convention* (Erickson, página 59).

Problemas prácticos

Hay cinco problemas prácticos cuyos enunciados aparecen a partir de la siguiente página.

A - Cantor

Source file name: `cantor.py`
Time limit: 1 second

The ternary expansion of a number is that number written in base 3. A number can have more than one ternary expansion. A ternary expansion is indicated with a subscript 3. For example, $1 = 1_3 = 0.222\ldots_3$, and $0.875 = 0.212121\ldots_3$. The Cantor set is defined as the real numbers between 0 and 1 inclusive that have a ternary expansion that does not contain a 1. If a number has more than one ternary expansion, it is enough for a single one to not contain a 1. For example, $0 = 0.000\ldots_3$ and $1 = 0.222\ldots_3$, so they are in the Cantor set. But $0.875 = 0.212121\ldots_3$ and this is its only ternary expansion, so it is not in the Cantor set. Your task is to determine whether a given number is in the Cantor set.

Input

The input consists of several test cases. Each test case consists of a single line containing a number x written in decimal notation, with $0 \leq x \leq 1$, and having at most 6 digits after the decimal point. The last line of input is '`END`'. This is not a test case.

The input must be read from standard input.

Output

For each test case, output '`MEMBER`' if x is in the Cantor set, and '`NON-MEMBER`' if x is not in the Cantor set.

The output must be written to standard output.

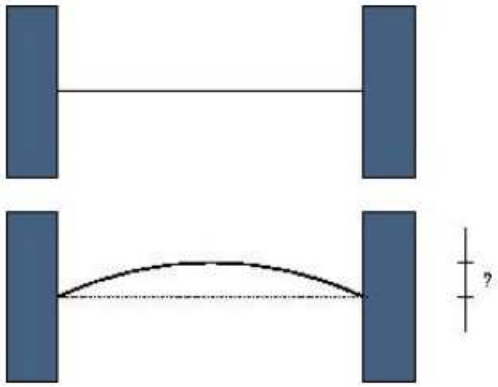
Sample Input	**Sample Output**

`0`
`1`
`0.875`
`END`
	`MEMBER`
	`MEMBER`
	`NON-MEMBER`

B - Expanding Rods

Source file name: `expand.py`
Time limit: 1 second

When a thin rod of length L is heated n degrees, it expands to a new length $L' = (1 + n * C) * L$, where C is the coefficient of heat expansion. When a thin rod is mounted on two solid walls and then heated, it expands and takes the shape of a circular segment, the original rod being the chord of the segment. Your task is to compute the distance by which the center of the rod is displaced.



Input

The input contains multiple lines. Each line of input contains three non-negative numbers: the initial length of the rod in millimeters, the temperature change in degrees and the coefficient of heat expansion of the material. Input data guarantee that no rod expands by more than one half of its original length. The last line of input contains three negative numbers and it should not be processed.

The input must be read from standard input.

Output

For each line of input, output one line with the displacement of the center of the rod in millimeters with 3 digits of precision.

The output must be written to standard output.

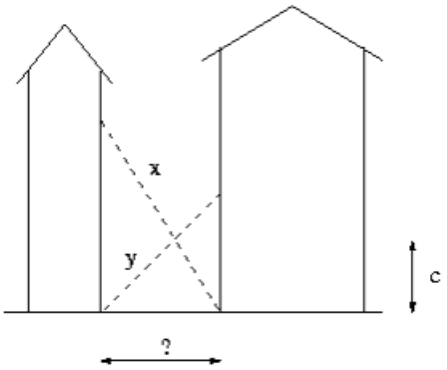
Sample Input	**Sample Output**

`1000 100 0.0001`
`15000 10 0.00006`
`10 0 0.001`
`-1 -1 -1`
	`61.329`
	`225.020`
	`0.000`

C - Crossed Ladders

Source file name: `ladders.py`
Time limit: 1 second

A narrow street is lined with tall buildings. An x foot long ladder is rested at the base of the building on the right side of the street and leans on the building on the left side. A y foot long ladder is rested at the base of the building on the left side of the street and leans on the building on the right side. The pointvwhere the two ladders cross is exactly c feet from the ground. How wide is the street?



Input

Each line of input contains three positive floating point numbers giving the values of x, y, and c.

The input must be read from standard input.

Output

For each line of input, output one line with a floating point number giving the width of the street in feet, with three decimal digits in the fraction.

The output must be written to standard output.

Sample Input	**Sample Output**

`30 40 10`
`12.619429 8.163332 3`
`10 10 3`
`10 10 1`
	`26.033`
	`7.000`
	`8.000`
	`9.798`

D - The Closest Pair Problem

Source file name: `pair.py`
Time limit: 1 second

Given a set of points in a two dimensional space, you will have to find the distance between the closest two points.

Input

The input contains several sets of input. Each set of input starts with an integer N ($0 \leq N \leq 10\,000$), which denotes the number of points in this set. The next N line contains the coordinates of N two-dimensional points. The first of the two numbers denotes the x-coordinate and the latter denotes the y-coordinate. The input is terminated by a set whose $N = 0$ and should not be processed. The value of the coordinates will be less than 40 000 and non-negative.

The input must be read from standard input.

Output

For each set of input produce a single line of output containing a floating point number (rounded up to four decimals after the decimal point), denoting the distance between the closest two points. If there is no such two points in the input whose distance is less than 10 000, print the line '`INFINITY`'.

The output must be written to standard output.

Sample Input	**Sample Output**
3 0 0 10000 10000 20000 20000 5 0 2 6 67 43 71 39 107 189 140 0	INFINITY 36.2215

E - WiFi

Source file name: `wifi.py`
Time limit: 1 second

One day, the residents of Main Street got together and decided that they would install wireless internet on their street, with coverage for every house. Now they need your help to decide where they should place the wireless access points. They would like to have as strong a signal as possible in every house, but they have only a limited budget for purchasing access points. They would like to place the available access points so that the maximum distance between any house and the access point closest to it is as small as possible. Main Street is a perfectly straight road. The street number of each house is the number of metres from the end of the street to the house. For example, the house at address 123 Main Street is exactly 123 metres from the end of the street.

Input

The first line of input contains an integer specifying the number of test cases to follow. The first line of each test case contains two positive integers n, the number of access points that the residents can buy, and m, the number of houses on Main Street. The following m lines contain the house numbers of the houses on Main Street, one house number on each line. There will be no more than 100 000 houses on Main Street, and the house numbers will be no larger than one million.

The input must be read from standard input.

Output

For each test case, output a line containing one number, the maximum distance between any house and the access point nearest to it. Round the number to the nearest tenth of a metre, and output it with exactly one digit after the decimal point.

The output must be written to standard output.

Sample Input	**Sample Output**
1 2 3 1 3 10	1.0