# Задача A. Делёж монет

| | |
|---|---|
| Имя входного файла: | `division.in` |
| Имя выходного файла: | `division.out` |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 64 мегабайта |

В этой задаче требуется разделить $K$ монет поровну на троих, или выяснить, что это невозможно.

## Формат входного файла

В первой строке входного файла задано число $K$ ($1 \leqslant K \leqslant 20$). Во второй строке даны $K$ целых чисел $A_1$, $A_2$, $\ldots$, $A_K$ через пробел — достоинства монет ($1 \leqslant A_i \leqslant 100$).

## Формат выходного файла

В первой строке выходного файла выведите '`YES`', если справедливый делёж монет возможен, и '`NO`', если разделить монеты поровну не удастся. Кроме того, если ответ положительный, в следующих трёх строках выведите через пробел наборы монет, на которые нужно разделить исходный набор, чтобы сумма монет в каждом наборе оказалась одинаковой. Если возможных решений несколько, разрешается выводить любое из них.

## Примеры

| division.in | division.out |
|---|---|
| 3<br>1 1 1 | YES<br>1<br>1<br>1 |
| 2<br>1 10 | NO |
| 5<br>1 2 3 4 5 | YES<br>1 4<br>2 3<br>5 |
| 6<br>1 2 4 8 16 32 | NO |

# Задача B. Наибольшее расстояние

| | |
|---|---|
| Имя входного файла: | `maxdist.in` |
| Имя выходного файла: | `maxdist.out` |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 64 мегабайта |

В некоторой стране $N$ городов и $M$ дорог. Каждая дорога представляет собой отрезок прямой, соединяющий два различных города, причём известно, что ни на одной дороге нет других городов, кроме тех, что задают её концы. Расстоянием между двумя городами считается минимальная суммарная длина дорог, по которым надо пройти, чтобы попасть из одного города в другой. Требуется вычислить наибольшее из таких расстояний.

## Формат входного файла

В первой строке входного файла задано число $N$ ($2 \leqslant N \leqslant 100$). В последующих $N$ строках содержатся пары целых чисел $X_i\,Y_i$ ($-1000 \leqslant X_i, Y_i \leqslant 1000$), задающие координаты $i$-го города на плоскости. Координаты всех городов различны. В следующей, $N+2$-ой строке, задано число $M$. Далее в $M$ строках записаны пары чисел $U_j\,V_j$, означающие, что существует дорога между городами $U_j$ и $V_j$. Дороги не имеют общих точек вне городов. Длина дороги между городами $A$ и $B$ задаётся обычным евклидовым расстоянием между точками плоскости: $\rho(A, B) = \sqrt{(X_B - X_A)^2 + (Y_B - Y_A)^2}$.

## Формат выходного файла

Выведите одно число — максимальное расстояние между двумя городами с точностью по крайней мере $10^{-6}$. Если существует пара городов, между которыми нет пути из дорог, выведите $-1$.

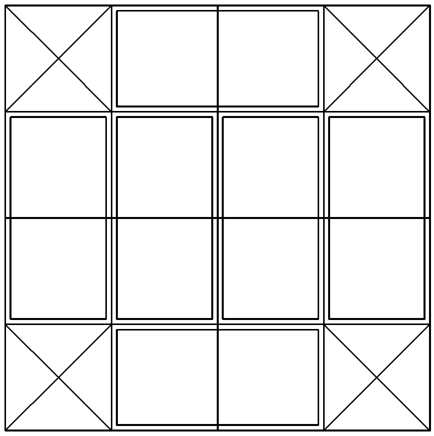
## Примеры

| maxdist.in | maxdist.out |
|---|---|
| 2<br>0 0<br>1 1<br>1<br>1 2 | 1.414213 |
| 2<br>0 0<br>1 1<br>0 | -1 |
| 5<br>0 0<br>0 2<br>2 0<br>2 2<br>1 1<br>6<br>1 2<br>2 4<br>4 3<br>3 1<br>2 5<br>4 5 | 4 |

# Задача C. Покрытие

| | |
|---|---|
| Имя входного файла: | cover.in |
| Имя выходного файла: | cover.out |
| Ограничение по времени: | 2 секунды |
| Ограничение по памяти: | 64 мегабайта |

Рассмотрим квадратный участок пола размера $4 \times 4$ дециметра, разделённый на 16 клеток размера $1 \times 1$ дециметр. Каждая клетка либо содержит дырку, либо является пустой. Требуется определить, можно ли покрыть все пустые клетки плитками размера $1 \times 2$ дециметра так, чтобы каждая пустая клетка была покрыта ровно одной плиткой и каждая плитка покрывала ровно две пустые клетки.



## Формат входного файла

Во входном файле заданы четыре строки, по четыре символа в каждой. Эти строки задают положение дырок в полу: каждый символ — это либо 'X', что означает дырку, либо '.', что означает пустую клетку пола.

## Формат выходного файла

Выведите 'YES', если можно замостить пол плитками, и 'NO' в противном случае.

## Примеры

| cover.in | cover.out |
|---|---|
| X..X<br>....<br>....<br>X..X | YES |
| .X.X<br>XXXX<br>.XXX<br>XXXX | NO |