The 41st Annual ACM

International Collegiate Programming Contest

Asia Regional – Daejeon

Nationwide Internet Competition

Practice Problem Set

Please check that you have 3 problems that are spanned across 5 pages in total (including this cover page).

The memory limits for the three problems are all the same, 512MB.

The 41st Annual ACM

International Collegiate Programming Contest

Asia Regional – Daejeon

Nationwide Internet Competition

Practice Problem A

Division

Time Limit: 1 Second

Given two integers a and b, write a program to print out a/b.

Input

Your program is to read from standard input. The first line contains two positive integers a and b, where $1 \leq a \leq 10{,}000$ and $1 \leq b \leq 10{,}000$.

Output

Your program is to write to standard output. Print exactly one line that contains a real number a/b rounded to two decimal places. Your output should contain two digits after decimal point.

The following shows sample input and output for three test cases.

Sample Input 1	Output for the Sample Input 1
`20 15`	`1.33`

Sample Input 2	Output for the Sample Input 2
`10 2`	`5.00`

Sample Input 3	Output for the Sample Input 3
`1049 10000`	`0.10`

The 41st Annual ACM
International Collegiate Programming Contest
Asia Regional – Daejeon
Nationwide Internet Competition

Practice Problem A
나눗셈
Time Limit: 1 Second

주어진 두 정수 a와 b에 대하여 a/b를 출력하는 프로그램을 작성하시오.

입력
입력은 표준 입력을 사용한다. 첫째 줄에 두 정수 a와 b가 주어지는데, 여기서 $1 \leq a \leq 10{,}000$이고 $1 \leq b \leq 10{,}000$이다.

출력
출력은 표준 출력을 사용한다. 한 줄에 실수 a/b를 소수점 셋째 자리에서 반올림하여 소수점 둘째 자리까지 출력한다.

다음은 세 개의 테스트 데이터에 대한 입력과 출력의 예이다.

입력 예제 1	입력 예제 1에 대한 출력
`20 15`	`1.33`

입력 예제 2	입력 예제 2에 대한 출력
`10 2`	`5.00`

입력 예제 3	입력 예제 3에 대한 출력
`1049 10000`	`0.10`

The 41st Annual ACM

International Collegiate Programming Contest

Asia Regional – Daejeon

Nationwide Internet Competition

Practice Problem B

3-Primes Problem

Time Limit: 1 Second

In number theory, the 3-primes problem states that:

> *Every odd number greater than 5 can be expressed as a sum of exactly three primes. (A prime may be used more than once in the same sum.)*

Some examples of the problem are:

$$7 = 2 + 2 + 3$$
$$11 = 2 + 2 + 7$$
$$25 = 7 + 7 + 11$$

In 1939, Russian mathematician I. M. Vinogradov showed that any *sufficiently large* odd integer can be expressible as a sum of three primes. Later it is shown that sufficiently large in Vinogradov's proof meant numbers greater than $3^{3^{15}} \approx 10^{7000000}$. The best known improved bound for the figure is approximately $e^{3100} \approx 2 \times 10^{1346}$. This number is too large to admit checking all smaller numbers by computer.

Given a positive odd integer greater than 5, write a program to test whether or not the integer can be represented as a sum of exactly three primes, where the primes may not be distinct.

Input

Your program is to read from standard input. The first line contains a positive integer $K (7 \leq K < 1{,}000)$.

Output

Your program is to write to standard output. Print exactly one line that contains three primes, in nondecreasing order, if the input number K can be represented as a sum of exactly three primes, otherwise print `0`(zero). If there is more than one case for three primes, print any case of them.

The following shows sample input and output for three test cases.

Sample Input 1	**Output for the Sample Input 1**
`7`	`2 2 3`

Sample Input 2	**Output for the Sample Input 2**
`11`	`2 2 7`

Sample Input 3	**Output for the Sample Input 3**
`25`	`5 7 13`

The 41st Annual ACM
International Collegiate Programming Contest
Asia Regional – Daejeon
Nationwide Internet Competition

Practice Problem B
3-소수 문제
Time Limit: 1 Second

정수론에서, 3-소수 문제는 다음과 같다.

> 5 보다 큰 모든 홀수는 세 소수(prime number) 합으로 표현할 수 있는가? (같은 소수를 한번 이상 사용할 수 있다.)

다음 예는, 홀수를 세 소수의 합으로 표현하는 것을 보여준다:

$$7 = 2 + 2 + 3$$
$$11 = 2 + 2 + 7$$
$$25 = 7 + 7 + 11$$

1939 년, 러시아 수학자 I. M. Vinogradov 는 충분히 큰 모든 홀수는 세개의 소수(prime number)의 합으로 표현할 수 있음을 보였다. Vinogradov 의 증명에서, 충분히 크다는 것은 $3^{3^{15}} \approx 10^{7000000}$보다 큰 수를 의미한다. 지금까지 알려진 가장 좋은 하계는 대략 $e^{3100} \approx 2 \times 10^{1346}$ 정도이다. 이는 컴퓨터로 작은 숫자들을 검사하여 확인하기에는 너무 큰 수이다.

5 보다 큰 홀수가 주어질 때, 이 수가 세 소수의 합으로 표현할 수 있는지를 판별하는 프로그램을 작성하시오. 같은 소수를 여러번 사용할 수 있다.

입력
입력은 표준 입력을 사용한다. 입력으로 첫째 줄에 양의 정수 $K (7 \leq K < 1{,}000)$가 주어진다.

출력
출력은 표준 출력을 사용한다. 입력되는 수 K가 세 소수의 합으로 표현할 수 있으면 이들 세 소수를 감소하지 않는(nondecreasing) 순서대로 출력한다. 세 소수의 합으로 표현할 수 없으면 0(zero)을 출력한다. 조건을 만족하는 세 소수가 여럿 있으면, 그 중 임의로 하나를 출력한다.

다음은 세 개의 테스트 데이터에 대한 입력과 출력의 예이다.

입력 예제 1	**입력 예제 1 에 대한 출력**
7	2 2 3

입력 예제 2	**입력 예제 2 에 대한 출력**
11	2 2 7

입력 예제 3	**입력 예제 3 에 대한 출력**
25	5 7 13

Practice Problem C
Kernel
Time Limit: 1 Second

A farmer Will has a plan to purchase a robot management system for his farm. The system operates multiple autonomous robots in the farm to collect the useful information. To control the robots, the system has a server, called *beacon*, at a fixed position in the farm. At the end of day, the robots should be returned to the beacon, and checked their status for the next day. For this, the beacon keeps sending a special signal to the robots. The robots immediately know from the signal where the beacon is, and try to reach the beacon by moving continuously toward the beacon.

More precisely, the farm is modeled as a simple rectilinear polygon P such that horizontal and vertical edges appear alternatingly along its boundary, its vertices are all distinct, and two edges have no intersections except at their end vertices. A robot as well as a beacon is represented as a point *in* P, that is, on the boundary of P or in the interior of P. A robot p now moves in P greedily to minimize its Euclidean distance to the beacon b as follows: p moves along the ray from p to b until it reaches b or hits the boundary of P. If it hits the boundary, then it may either still reduce its Euclidean distance to b by sliding on the boundary, or it cannot reduce the distance even to any direction. A point p in P is *attracted* by b in P (equivalently, b *attracts* p) if it eventually reaches to b in P by strictly decreasing its Euclidean distance to b.

Let us look at the Figure 1 below. In the left figure, a point p can move along the path marked with thick solid segments, and finally reach to the beacon b. In the right figure, it can reach to x, but it cannot move anymore because there is no direction from x to reduce its Euclidean distance to b.

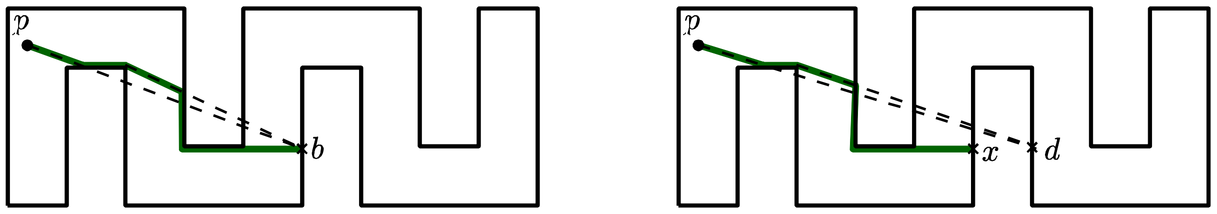

Figure 1. The cases where a beacon $\boldsymbol{b}$ can or cannot attract a point $\boldsymbol{p}$.

A *kernel* of P is the set of points in P that can attract all points in P. If a beacon is placed at any position of the kernel of P, then all points in P can be attracted by the beacon, so only one beacon is sufficient to call all the robots. But the kernel of some polygon may not exist.

Given a simple rectilinear polygon P of n vertices, write a program to decide whether its kernel exists or not.

Input
Your program is to read from standard input. The first line contains an integer, n ($4 \leq n \leq 10{,}000$), where n is the number of vertices in a simple rectilinear polygon P. The following n lines give the coordinates of the vertices in counterclockwise direction. Each vertex is represented by two numbers separated by a single space, which are the x-coordinate and the y-coordinate of the vertex, respectively. Each coordinate is given as an integer between $-1{,}000{,}000{,}000$ and $1{,}000{,}000{,}000$, inclusively. Note that all the vertices are distinct.

Output

Your program is to write to standard output. Print exactly one line that contains `YES` if the polygon given in the test case has a kernel, `NO` otherwise.

The following shows sample input and output for two test cases.

Sample Input 1	Output for the Sample Input 1

8
0 0
3 0
3 2
2 2
2 1
1 1
1 2
0 2
	YES

Sample Input 2	Output for the Sample Input 2

12
0 0
5 0
5 3
3 3
3 2
4 2
4 1
1 1
1 2
2 2
2 3
0 3
	NO