A. Вектор

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Даны два ненулевых вектора. Требуется вычислить:

* Длину первого и второго вектора (два числа)
* Вектор, образованный сложением данных двух векторов
* Скалярное и векторное произведения данных векторов
* Площадь треугольника, построенного из этих векторов

В общем, обычная векторная жизнь.

Формат ввода

В двух строках входного файла заданы по четыре целых числа, не превосходящих по модулю *10 000*, — координаты начала и конца первого вектора, затем второго.

Формат вывода

В каждой строке выходного файла — ответ на соответствующий пункт задачи с точностью не менее *10-6*.

Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 1 2 6  1 1 7 8 | 5.830952 9.219544  3.000000 12.000000  17.000000 -51.000000  25.500000 |

B. Две прямые

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Даны коэффициенты *A1*, *B1*, *C1* уравнения первой прямой и коэффициенты *A2*, *B2*, *C2* уравнения второй прямой. Требуется:

* Построить направляющие векторы для обеих прямых (вывести координаты)
* Найти точку пересечения двух прямых или вычислить расстояние между ними, если они параллельны

Формат ввода

В первой строке входного файла находятся три числа — коэффициенты нормального уравнения для первой прямой. Во второй строке — коэффициенты для второй прямой.

Все числа во входном файле по модулю не превосодят *10 000*.

Формат вывода

В каждой строке выходного файла выведите ответ на соответствующий пункт задачи с точностью до *10-6*.

Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 1 -1  1 -1 0 | 1.000000000 -1.000000000  -1.000000000 -1.000000000  0.500000000 0.500000000 |

# C. Точка и две точки

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Определите, принадлежит ли точка *C* заданной прямой, лучу и отрезку, образованными точками *A* и *B*.

Задумайтесь принадлежит ли вам ваша жизнь.

## Формат ввода

В первой строке входного файла даны два целых числа — координаты точки *C*. Во двух следующих строках в таком же формате заданы точки *A* и *B* (*A ≠ B*).

Все числа во входном файле по модулю не превосодят *10 000*.

## Формат вывода

В первой строке выведите «YES», если точка *C* принадлежит прямой *AB*, и «NO» в противном случае. Во второй и третьей строках аналогично выведите ответы для луча *AB* (*A* — начало луча) и отрезка *AB*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 6  3 7  5 8 | YES  NO  NO |

# D. Расстояния

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Даны три точки *A, B* и *C*. Необходимо подсчитать расстояния от точки *C* до прямой, луча и отрезка, образованного точками *A* и *B*.

## Формат ввода

В первой строке входного файла даны два целых числа — координаты точки *C*. Во двух следующих строках в таком же формате заданы точки *A* и *B* (*A ≠ B*).

Все числа во входном файле по модулю не превосодят *10 000*.

## Формат вывода

В первой строке выходного файла выведите одно вещественное число — расстояние от точки *C* до прямой. В следующих двух строках выведите соответственно расстояния до луча *AB* (*A* — начало луча) и до отрезка *AB*. Все числа выводить с точностью не менее *10-6*. Луч строится по направлению от точки *A* к точке *B*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3 0  1 1  2 1 | 1.000000000  1.000000000  1.414213562 |

# E. Два отрезка

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Необходимо проверить, пересекаются ли два отрезка.

## Формат ввода

В двух строках входного файла заданы по четыре целых числа, не превосходящих по модулю *10 000*, — координаты концов первого отрезка, затем второго.

## Формат вывода

В первой строке выходного файла выведите «YES», если отрезки имеют общие точки, и «NO» в противном случае.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5 1 2 6  1 1 7 8 | YES |

# F. Отрезки на дистанции

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Найдите расстояние между двумя отрезками.

## Формат ввода

В двух строках входного файла даны по четыре целых числа, не превосходящих по модулю *10 000*, — координаты концов сначала первого, затем второго отрезков.

## Формат вывода

В первой строке выходного файла выведите одно вещественное число — расстояние между отрезками с точностью не менее *10-6*.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 1 1 2 2  2 1 3 0 | 0.7071067812 |

# G. (Не)выпуклый

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Проверьте многоугольник на выпуклость. Задумайтесь над тем, что в 1863 году слово "впуклый" использовалось наравне с словом "выпуклый", но за следующие двести лет сокрушительно проиграло конкуренцию букве "ы".

## Формат ввода

В первой строке одно число *N* (*3 ≤ N ≤ 100000*). Далее в *N* строках по паре целых чисел — координаты очередной вершины простого многоугольника в порядке обхода по или против часовой стрелки.

Координаты всех точек целые, по модулю не превосходят *107*.

## Формат вывода

Одна строка «YES», если приведённый многоугольник является выпуклым, и «NO» в противном случае.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  0 0  0 1  1 0 | YES |

# H. Внутри или нет?

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Проверьте, лежит ли точка внутри многоугольника.

## Формат ввода

В первой строке вводятся три целых числа – *N* (https://contest.yandex.ru/testsys/tex/render/MyBcbGVxc2xhbnQgTiBcbGVxc2xhbnQgMTAwMDAw.png) и координаты точки. Далее в *N* строках задается по паре целых чисел – координаты очередной вершины простого многоугольника в порядке обхода по или против часовой стрелки.

## Формат вывода

Выведите одну строку: «YES», если заданная точка содержится в приведённом многоугольнике или на его границе, и «NO» в противном случае.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3 2 3  1 1  10 2  2 8 | YES |

# I. Площадь

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Ежегодно в Бостоне совершается более 300 ограблений банков. Большинство этих профессионалов живут в районе с площадью в всего S квадратных миль под названием Чарльзтаун. Найдите эту площадь.

## Формат ввода

В первой строке одно число *N* (*3 ≤ N ≤ 100 000*). Далее в *N* строках по паре чисел — координаты очередной вершины простого многоугольника в порядке обхода по или против часовой стрелки.

Координаты целые по модулю не превосходят *10 000*.

## Формат вывода

Одно число — величина площади приведённого многоугольника с абсолютной точностью.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 3  1 0  0 1  1 1 | 0.500000000000000 |

# J\*. Выпуклая оболочка

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Дано *N* точек на плоскости. Нужно построить их выпуклую оболочку, используя алгоритм Грэхема.

Гарантируется, что выпуклая оболочка не вырождена.

## Формат ввода

На первой строке число *N* *(3 ≤ N ≤ 105)*. Следующие *N* строк содержат пары целых чисел *x* и *y* *(-109 ≤ x, y ≤ 109)* — точки.

Будьте аккуратны! Точки произвольны. Бывают совпадающие, бывают лежащие на одной прямой в большом количестве.

## Формат вывода

В первой строке выведите *K* — число вершин выпуклой оболочки. Следующие *K* строк должны содержать координаты вершин в порядке обхода. Никакие три подряд идущие точки не должны лежать на одной прямой.

Кроме того, в последней строке выведите площадь получившейся выпуклой оболочки. Площадь необходимо вывести c точностью 1 знак после запятой.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  0 0  2 0  0 2  1 1  2 2 | 4  0 0  0 2  2 2  2 0  4.0 |

## Примечания

Оболочку нужно вывести в порядке обхода по часовой стрелке, начиная с самой левой (в случае нескольких таких, то с самой левой самой нижней) точки. Иначе ответ считается неверным.

Набрать здесьОтправить файл

# K\*. Выпуклая оболочка. Возвращение легенды

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод |
| Вывод | стандартный вывод |

Дано *N* точек на плоскости. Нужно построить их выпуклую оболочку. Всё то же самое, но по-другому. В этот раз реализуйте алгоритм Джарвиса.

Гарантируется, что выпуклая оболочка не вырождена.

## Формат ввода

На первой строке число *N* *(3 ≤ N ≤ 105)*. Следующие *N* строк содержат пары целых чисел *x* и *y* *(-109 ≤ x, y ≤ 109)* — точки.

Будьте аккуратны! Точки произвольны. Бывают совпадающие, бывают лежащие на одной прямой в большом количестве.

## Формат вывода

В первой строке выведите *K* — число вершин выпуклой оболочки. Следующие *K* строк должны содержать координаты вершин в порядке обхода. Никакие три подряд идущие точки не должны лежать на одной прямой.

Кроме того, в последней строке выведите площадь получившейся выпуклой оболочки. Площадь необходимо вывести c точностью 1 знак после запятой.

## Пример

| **Ввод**  Скопировать ввод | **Вывод**  Скопировать вывод |
| --- | --- |
| 5  0 0  2 0  0 2  1 1  2 2 | 4  0 0  0 2  2 2  2 0  4.0 |

## Примечания

Оболочку нужно вывести в порядке обхода по часовой стрелке, начиная с самой левой (в случае нескольких таких, то с самой левой самой нижней) точки. Иначе ответ считается неверным.