

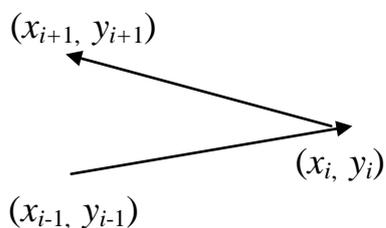
Задача 1 «Штраф за левые повороты». В городе X водителям запрещено выполнять левые повороты. За каждый такой поворот водитель должен уплатить штраф в размере M рублей. Для слежки за водителями в городе установлена компьютерная система, фиксирующая координаты автомобиля в начале движения, в конце движения и во время поворота.

Исходные данные: N – количество зафиксированных координат автомобиля, (x_i, y_i) – координаты автомобиля в процессе движения, $i=1,2, \dots, N$, где (x_1, y_1) – точка начала движения, (x_N, y_N) – последняя точка маршрута автомобиля.

Требуется по заданной последовательности координат движения вычислить сумму штрафа водителя.

Траекторию движения автомобиля можно представить в виде ломаной, состоящей из направленных отрезков из точек (x_i, y_i) в точки (x_{i+1}, y_{i+1}) , $i=1,2,\dots,N-1$. Поворот считается левым, если направление текущего отрезка пути a_{i+1} меняется относительно предыдущего отрезка a_i в левую сторону, т.е. против часовой стрелки.

Таким образом, решение задачи сводится к вычислению количества пар участков пути a_i и a_{i+1} , для которых выполняется условие $[a_i \times a_{i+1}] > 0$. Координаты векторов a_i и a_{i+1} вычисляются через координаты точек (x_i, y_i) : $a_i = (x_i - x_{i-1}, y_i - y_{i-1})$, $a_{i+1} = (x_{i+1} - x_i, y_{i+1} - y_i)$, следовательно, $[a_i \times a_{i+1}] = (x_i - x_{i-1})(y_{i+1} - y_i) - (y_i - y_{i-1})(x_{i+1} - x_i)$, $i=2, \dots, N-1$.



Задача 2 «Здесь будет город-сад». Жители одного дома города X решили высадить у себя во дворе несколько деревьев. Так как жильцы не смогли договориться, как должны быть расположены посадки, то каждый посадил дерево в том месте двора, где ему захотелось. После проведения посадок полученный сад решили обнести забором. Но пока доски не привезли, деревья обвязали одной длинной веревкой.

Исходная информация: N – количество деревьев в саду, (x_i, y_i) – координаты деревьев, $i=1,2, \dots, N$. Так как были высажены молодые саженцы, то их толщиной можно пренебречь.

Требуется определить, к каким из посаженных деревьев надо привязать веревку так, чтобы все деревья оказались внутри обнесенной зоны, а длина веревки была минимальная.

Эта задача сводится к определению для заданного множества точек на плоскости выпуклой оболочки, то есть выпуклого многоугольника с вершинами в некоторых точках из заданного множества, охватывающего все его точки.

Будем строить выпуклую оболочку в порядке обхода участка по часовой стрелке. Найдем самую левую точку $M_0 = (x_0, y_0)$, $x_0 = \min\{x_i\}$. Если таких точек несколько, то возьмем самую нижнюю из них. Эта точка наверняка принадлежит искомой выпуклой оболочке. Зададим первоначальный вектор a_0 с началом в точке (x_0, y_0) , параллельный оси Oy .

Следующей точкой оболочки будет такая точка M_1 , чтобы вектор a_1 с началом в точке M_0 и концом в точке M_1 образовывал с первоначальным вектором a_0 минимальный угол. Если таких точек несколько, то выбирается точка, расстояние до которой максимально.

Далее процесс продолжаем, то есть ищем точку M_2 с минимальным углом между вектором a_1 и вектором a_2 с началом в точке M_1 и концом в точке M_2 , затем точку M_3 и т.д. Процесс прекращаем, когда дойдем до первой выбранной точки или количество точек в оболочке станет равно N .

Для определения угла между векторами используется скалярное произведение. Причем сам угол можно не вычислять, так как минимальному углу соответствует максимальный косинус угла.

Задача 3 «Заяц». Недалеко от города X находится зоосад. Здешний житель, заяц, хаотично прыгая, оставил след в виде замкнутой самопересекающейся ломаной, охватывающей территорию его владения. Найти площадь минимального по площади выпуклого многоугольника, описанного вокруг этой территории.

В данной задаче необходимо не только найти выпуклую оболочку множества точек, но и вычислить площадь выпуклого многоугольника с заданным набором вершин.

Исходные данные: N – количество вершин выпуклого многоугольника, (x_i, y_i) – координаты вершин, $i=1, 2, \dots, N$.

Требуется определить площадь выпуклого N -угольника.

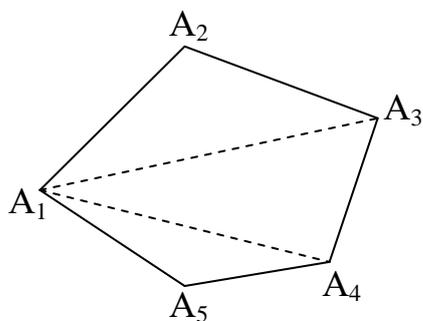
Площадь N -угольника может быть вычислена как сумма площадей треугольников, из которых N -угольник составлен. Для нахождения площади треугольника используем векторное произведение. Длина векторного произведения векторов, как известно, равна удвоенной площади треугольника, построенного на этих векторах. Пусть вершины треугольника расположены в точках $A=(x_1, y_1)$, $B=(x_2, y_2)$, $C=(x_3, y_3)$. Совместим начало координат с первой точкой. Векторное произведение равно

$$[AB \times AC] = \begin{vmatrix} i & j & k \\ x_2 - x_1 & y_2 - y_1 & 0 \\ x_3 - x_1 & y_3 - y_1 & 0 \end{vmatrix},$$

следовательно, площадь треугольника равна

$$S_{ABC} = 1/2 ((x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)).$$

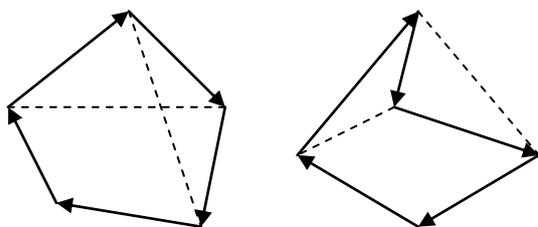
Значение величины S_{ABC} может быть как положительным, так и отрицательным числом, так как оно зависит от взаимной ориентации векторов AB и AC , поэтому говорят, что площадь ориентированная.



Для нахождения площади N -угольника последний требуется разбить на треугольники и найти сумму ориентированных площадей этих треугольников. Разбиение N -угольника на треугольники можно провести так: зафиксировать одну из вершин N -угольника, например, первую $A_1=(x_1, y_1)$ и рассматривать все треугольники $A_1A_iA_{i+1}$, $i=2, 3, \dots, N-1$.

Заметим, что аналогичным способом можно находить площадь произвольного многоугольника.

Использование свойства ориентации площади треугольника, вычисленной по



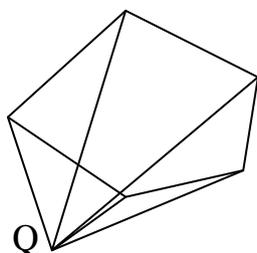
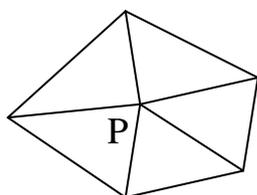
векторному произведению, позволяет определить, является ли заданный многоугольник выпуклым. Для выпуклого многоугольника все треугольники, образованные тройками соседних вершин в порядке их обхода, имеют одну ориентацию. Поэтому проверка многоугольника на выпуклость может быть проведена с помощью последовательного сравнения знаков векторных произведений для всех пар соседних сторон многоугольника.

Задача 4 «Тигр в загоне». *Недалеко от города X находится заповедник, в котором обитают уссурийские тигры. Работники заповедника очень переживают, когда тигр покидает охраняемую зону. Программа охраны уссурийских тигров предусматривает снабжение каждого тигра ошейником с радиомаяком. Сигнал от тигриного радиомаяка поступает в центр охраны и позволяет определить местоположения тигра. Территория заповедника представляет собой произвольный многоугольник.*

Исходные данные: N – количество вершин многоугольника, задающего заповедник, (x_i, y_i) – координаты его вершин, $i=1,2, \dots, N$. (X, Y) – координаты точки, в которой находится тигр.

Требуется определить, находится ли тигр на территории заповедника, или надо срочно снаряжать спасательную экспедицию.

Очень часто при решении задач геометрического содержания требуется проверить, лежит ли заданная точка внутри или вне многоугольника. Таким способом можно решить, например, задачу о Бармаглоте, проверяя каждую точку Бармаглота относительно одеяла-многоугольника. Есть много способов проверки принадлежности точки многоугольнику, однако мы приведем здесь один из них, основанный на использовании произведения векторов.



Идея метода заключается в том, чтобы определить сумму углов, под которыми стороны многоугольника видны из проверяемой точки. Если точка лежит внутри многоугольника, то суммарный угол равен 2π (точка P на рисунке), если же точка лежит вне

многоугольника, то сумма углов не равна 2π (точка Q).

Таким образом, для решения надо перебрать в цикле последовательно все вершины многоугольника и найти сумму углов между векторами PA_i и PA_{i+1} , $i=1,2, \dots, N$. Не забудьте добавить угол между векторами PA_N и PA_1 . Для определения величины угла между векторами нам потребуется формула скалярного произведения.

Так как стороны многоугольника должны рассматриваться последовательно, в порядке обхода вершин, то при нахождении суммарного угла следует учитывать взаимное расположение векторов. Угол, под которым сторона видна из исследуемой точки, может быть как положительным, так и отрицательным. Для определения знака угла воспользуемся векторным произведением. Знак векторного произведения и определит знак конкретного угла в общей сумме.