**ГБОУ "Президентский ФМЛ № 239"**

**Поиск фигуры максимальной площади, ограниченной двумя острыми углами**

Годовой проект по информатике

Автор:

Ученица 10-7 класса

Кучина Мария

Санкт-Петербург

2021

**Постановка задачи**

На плоскости задано множество "острых углов". Найти такие два "острых угла", что фигура, находящаяся "внутри" обоих "острых углов" замкнута и имеет максимальную площадь. Опция: искать только такие фигуры, все точки которых находятся внутри отображаемой на экране системы координат. В этом случае крайне желательно "залить цветом пространство фигуры". В качестве ответа: выделить найденные два "острых угла", выделить контур фигуры, которая ограничивает точки "внутри" обоих "острых углов", желательно выделить внутреннее пространство фигуры ("залить цветом").

Имея множество острых углов, нужно найти такие, которые образуют своими лучами четырёхугольники. Для этого мы должны рассчитать координаты точек, являющихся пересечением лучей углов. После нужно посчитать площади всех четырёхугольников, образующихся при пересечении лучей углов. Затем сравним все значения и выберем максимальной из них. Это будет четырёхугольник с максимальной площадью. Его нужно залить цветом, выделить границы и выделить углы, в которых заключена фигура.

**Уточнение исходных и входных данных**

**Исходные данные:**

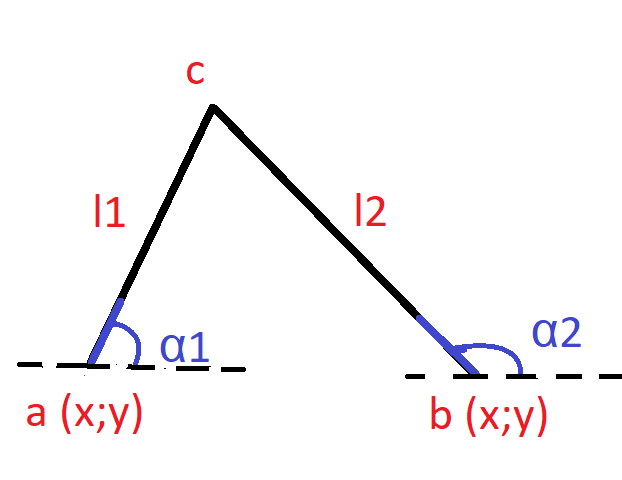
Входными данными является множество острых углов. Ввод данных реализован тремя способами ­– по одному, с указанием необходимых величин, загрузкой из файла angles.txt и генерацией заданного количества случайных углов.

**Выходные данные:**

Внутренними выходными данными являются переменные a1Sol и a2Sol, определяющие номера углов в массиве angles, являющиеся решением задачи и четыре переменные типа Vector2, определяющие вершины четырёхугольника с максимальной площадью. На экран выводятся углы, выделенные жирной синей линией и четырёхугольник с максимальной площадью, закрашенный красным цветом.

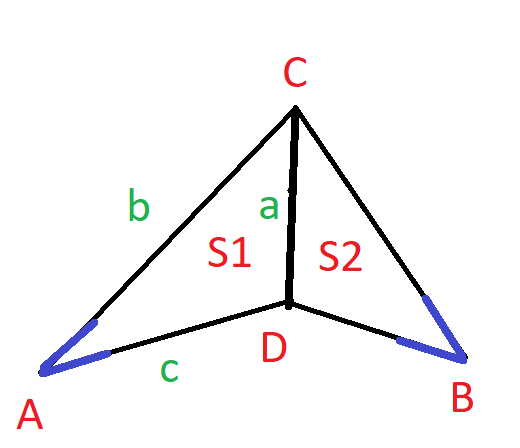
**Математическая модель**

Для решения задачи необходимо перебрать все пары углов, определить образует ли пересечение их лучей четырёхугольник и выбрать четырёхугольник с максимальной площадью. Для нахождения точки пересечения лучей углов необходимо решить систему уравнений

****

Решением являются длины отрезков l1, l2 и координаты точки пересечения c.x и c.y:

Чтобы углы образовали четырёхугольник, таких точек пересечения должно быть две или четыре. Для таких пар углов определяется площадь четырёхугольника как сумма площадей двух треугольников S1 и S2.

**

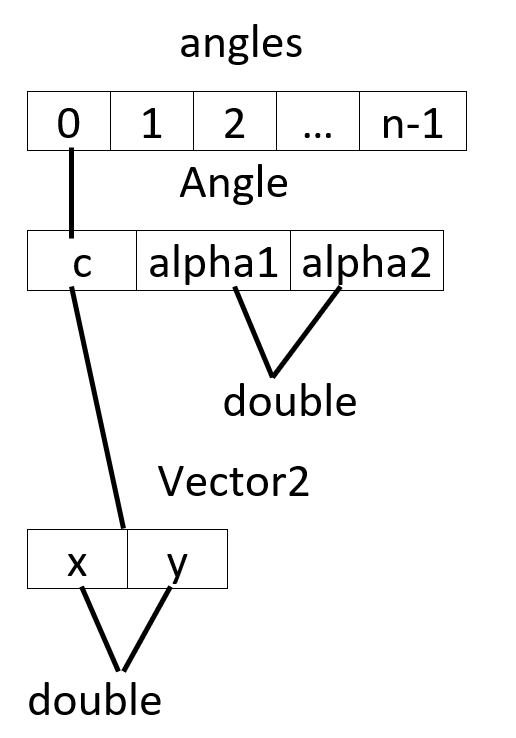
Площади треугольников находим по формуле Герона.

Где p – полупериметр треугольника.

Длины сторон треугольника находим по теореме Пифагора.

**Анализ используемой структуры данных**

Углы задаются в динамическом массиве angles объектами класса Angle. Класс Angle содержит координаты центра угла, заданные объектом «с» класса Vector2, и два угла-направления лучей, заданные переменными alpha1, alpha2 типа double.



Для решения потребуются следующие переменные:

i, j – переменные цикла;

c1, c2, c3, c4, cMax, cMin, для хранения 4 точек пересечения, самой дальней и самой ближней из них;

l1, l2, l3, l4, lMin, lMax - расстояния до точек пересечения, минимальное и максимальное из них;

beam11, beam12, beam 21, beam22 - счетчики точек на лучах;

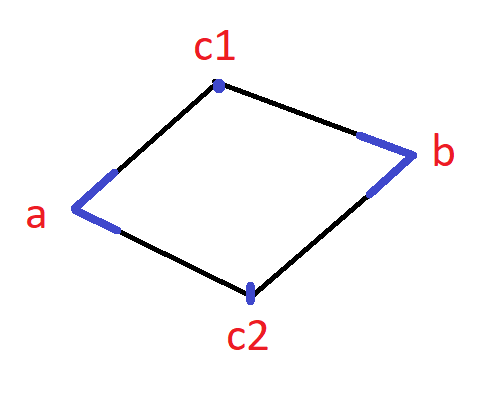
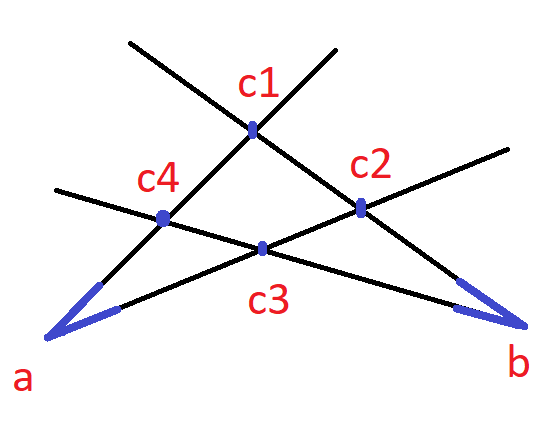
s, sMax – вычисленная и максимальная площадь;

a1sol, a2sol - номера углов-решений;

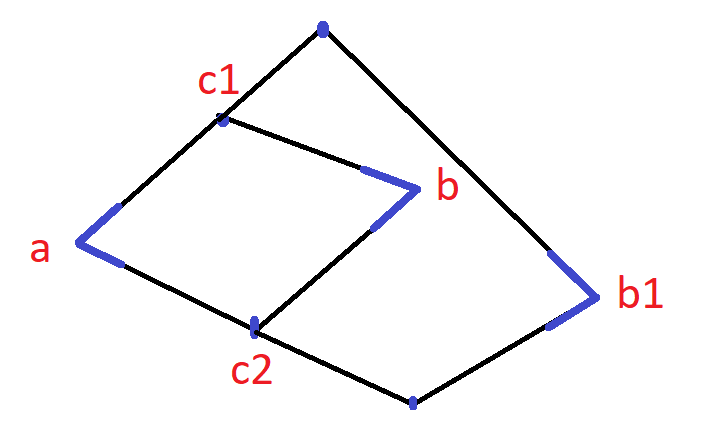
maxA, maxB, maxC, maxD - вершины четырехугольника.

**Описание алгоритма**

Имеется 2 варианта образования четырёхугольника парой углов:



1. Для решения требуется перебрать все пары углов с помощью вложенного цикла, при этом считается количество точек пересечения
2. Вычисляются длины между углами через общую точку пересечения (a-c1-b, a-c4-b…)
3. Если углы имеют 4 точки пересечения, то из длин из пункта 2 выбираются максимальная и минимальная
4. Вычисляются площади каждого четырёхугольника, как сумма площадей треугольников
5. В цикле выбирается максимальная площадь и углы её образующие



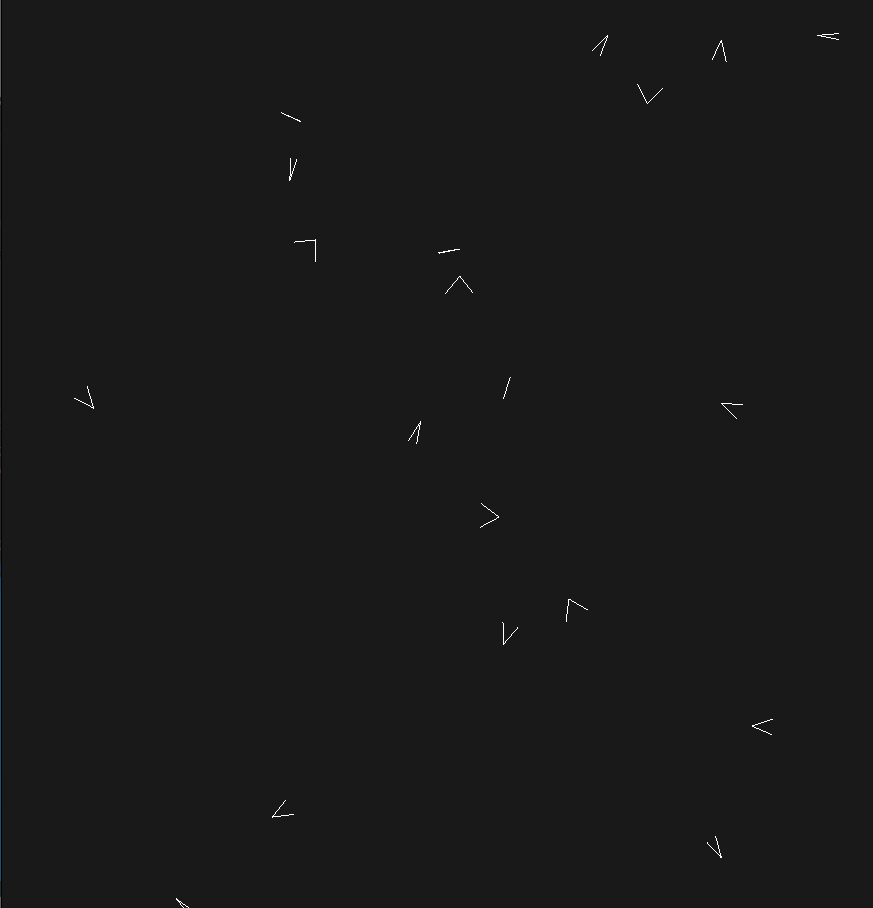
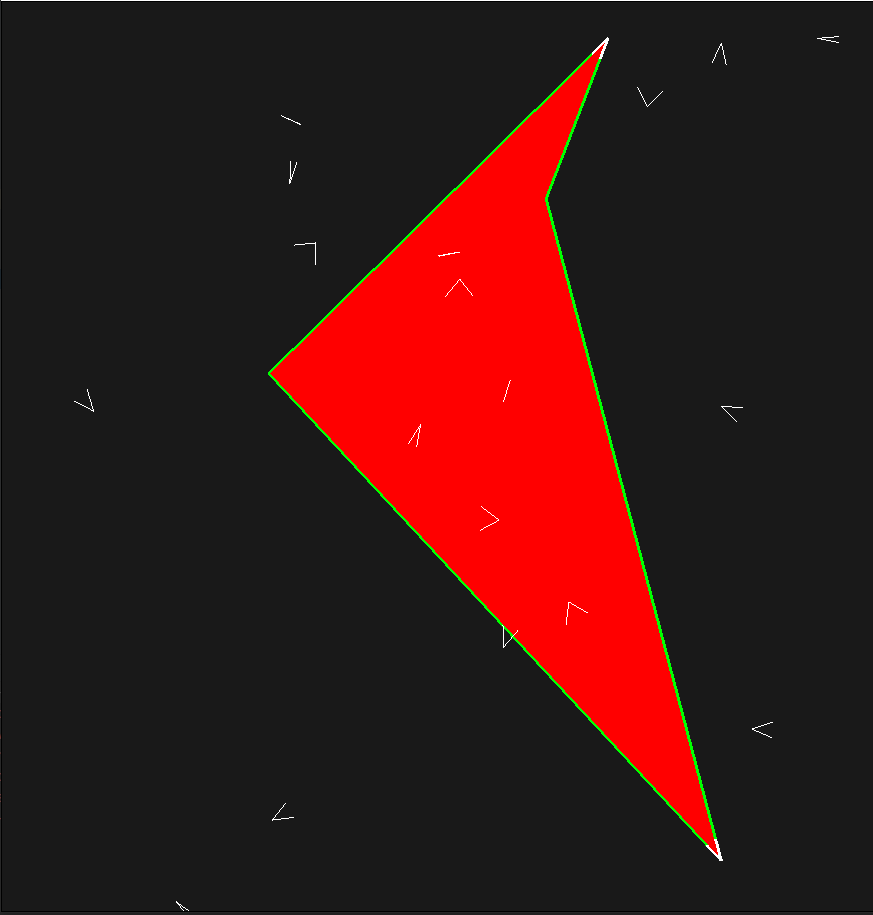
1. После перебора всех пар углов, на экран выводятся найденные углы и выделенный четырёхугольник с максимальной площадью.



**Листинг программы**

public void solve() {  
 double sMax = 0; // обнуляем максимальную площадь  
 // перебираем пары углов  
 for (int i=0;i<angles.size();i++) {  
 for (int j=i+1;j<angles.size();j++){  
 Angle a = angles.get(i);//выбираем углы для работы в цикле  
 Angle b = angles.get(j);  
 if (a != b) { // если углы не совпадают  
 // Инициализация переменных для пары углов  
 Vector2 c1, c2, c3, c4, cMax, cMin; // 4 точки пересечения, самая дальняя и самая ближняя из них  
 double l1=0, l2=0, l3=0, l4=0, lMin=100, lMax=0; // расстояния до точек пересечения  
 int beam11 = 0; int beam12 = 0; int beam21 = 0; int beam22 = 0; // счетчики точек на каждом луче  
 c1 = cross (a.c, b.c, a.alpha1, b.alpha1); // поиск точек пересечения  
 cMin = c1; cMax = c1;  
 if (c1.x!=-100) { // если найдено пересечение  
 l1 = getlength(c1, a.c)+getlength(c1,b.c); // расстояние между углами через точку пересечения  
 lMin = l1; // определяем самое большое и маленькое   
 lMax = l1; // расстояние  
 beam11 ++; // изменяем счетчики точек на лучах  
 beam21 ++;  
 }  
 c2 = cross(a.c, b.c, a.alpha1, b.alpha2); // поиск точек пересечения  
 if (c2.x!=-100) { // если найдено пересечение  
 l2 = getlength(c2, a.c)+getlength(c2,b.c); // расстояние между углами через точку пересечения  
 if(l2>lMax) {lMax=l2; cMax = c2; } //определяем самое большое   
 if(l2<lMin) {lMin=l2; cMin = c2; } // и маленькое расстояние  
 beam11 ++; // изменяем счетчики точек на лучах  
 beam22 ++;  
 }  
 c3 = cross(a.c, b.c, a.alpha2, b.alpha1); // поиск точек пересечения  
 if (c3.x!=-100) { // если найдено пересечение  
 l3 = getlength(c3, a.c)+getlength(c3,b.c); // расстояние между углами через точку пересечения  
 if(l3>lMax) {lMax=l3; cMax = c3; } // определяем самое большое   
 if(l3<lMin) {lMin=l3; cMin = c3; } // и маленькое расстояние  
 beam12++; // изменяем счетчики точек на лучах  
 beam21++;  
 }  
 c4 = cross(a.c, b.c, a.alpha2, b.alpha2);  
 if (c4.x!=-100) { // если найдено пересечение  
 l4 = getlength(c4, a.c)+getlength(c4,b.c); // расстояние между углами через точку пересечения  
 if(l4>lMax) { cMax = c4; } // определяем самое большое и   
 if(l4<lMin) { cMin = c4; } // маленькое расстояние  
 beam12++; // изменяем счетчики точек на лучах  
 beam22++;  
 }  
 // если пересекаются - вычисление площади, если она больше предыдущего максимума - запоминаем точки пересечения и площадь  
 if ((beam11 > 0) && (beam12 > 0) && (beam21 > 0) && (beam22 > 0) && (beam11+beam12+beam21+beam22)%4==0) {  
 double s = getSqare(a.c, b.c, cMax) + getSqare(a.c, b.c, cMin);  
 if (s > sMax) {  
 sMax = s; // запоминаем максимальную площадь  
 a1sol = i; // запоминаем номер первого угла-решения  
 a2sol = j; // запоминаем номер второго угла-решения  
 maxA = a.c; // запоминаем вершины четырехугольника   
 maxB = b.c;  
 maxC = cMax;  
 maxD = cMin;  
 }  
 }  
 }  
 }  
 }  
}

**Пример работы программы**

 Входные данные Выходные данные

**Анализ правильности решения**

Решение проверено на различных наборах углов – как сгенерированных случайным образом, так и введенных по одному.

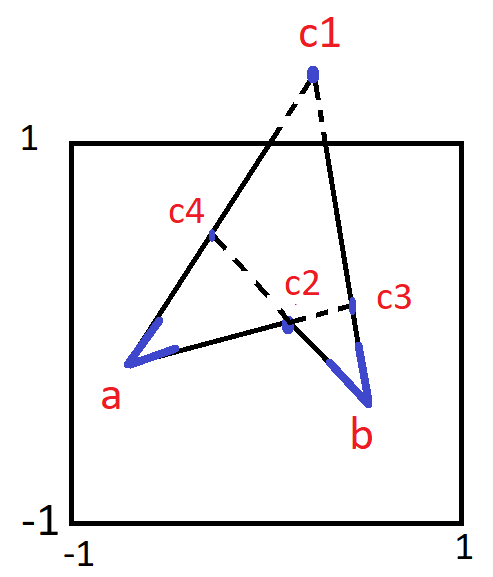
При проверке анализировалось:

- нахождение решения в заданной области;

- отсутствие пар углов, образующих четырёхугольник большей площади;

- правильность построения четырёхугольника.

В некоторых комбинациях программа неправильно выбирала пары углов. Ошибка возникала в том случае, если одна из точек пересечения лучей углов образовывалась за пределами допустимой области. Такие точки не учитывались алгоритмом и получалось 3 точки пересечения.



Для исправления ошибки в программу были добавлены переменные, считающие количество точек пересечения на каждом из лучей. После проверки того, что количество точек пересечения в допустимой области кратно 2, ошибка пропала.

В результате программа правильно выделяет максимальную площадь между двумя острыми углами для любого количества углов в массиве.