def get\_sum#создаем метод суммы  
return 0.0 unless self.respond\_to? :points [#если](http://vk.com/feed?section=search&q=%23%D0%B5%D1%81%D0%BB%D0%B8)текущий метод не отвечает points, то мы возвращам 0(нет ребер)  
pts = self.points#возвращает точки выпуклой оболочки  
p pts  
s = 0.0#сумма углов 0  
pts.size.times do |i|#н раз проходим по всем ребрам нашей выпуклой оболочки  
p1,p2 = pts[i-1], pts[i]#находим вершины ребра  
m = cross\_point $points[0], $points[1], p1, p2# находим точку пересечения ребра и прямой  
if m#если точка пересечения существует (ребра и прямой)  
cp = (m == p1) ? p2 : p1#выбераем точку на ребре не совпадающей с точкой пересечения  
cl = (m == $points[0]) ? $points[1] : $points[0]#выберам точку на прямой не совпадающей с точкой пересечения  
s += angle(cp, m, cl).abs unless cp == cl#вычисляем угл между ребром и премой и добавляем его в сумму  
end  
end  
s#возвраем сумму всех углов (пробежались в цикле по всем ребрам)  
end  
  
# находит уравнение прямой, заданной двумя точками  
def equation\_from\_segment(p1, p2)  
# Ax + By + C = 0  
x1, y1 = p1.x, p1.y  
x2, y2 = p2.x, p2.y  
  
a = y1 - y2  
b = x2 - x1  
c = x1 \* y2 - x2 \* y1  
[a, b, c]  
end  
  
# точка пересечения линии AB и отрезка PQ  
def cross\_point(a, b, p, q)  
a1, b1, c1 = equation\_from\_segment(a, b)  
a2, b2, c2 = equation\_from\_segment(p, q)  
  
d = (a1\*b2-a2\*b1)  
# Если прямые не перескаются - завершаем работу, отрезки тоже не будут пересекаться  
return false if d == 0  
  
# Находим координаты пересечения прямых  
x = -(c1\*b2-c2\*b1).to\_f / d  
y = -(a1\*c2-a2\*c1).to\_f / d  
  
# Точка пересечения прямых должна принадлежать отрезку PQ  
m = R2Point.new(x, y)  
return false unless m.inside?(p, q)  
m  
end   
  
# ищем угол В  
def angle a, b, c  
# координаты векторов  
ba\_x = a.x - b.x  
ba\_y = a.y - b.y  
bc\_x = c.x - b.x  
bc\_y = c.y - b.y  
  
dot\_product = ba\_x\*bc\_x + ba\_y\*bc\_y # скалярное произведение  
module\_ba = ba\_x\*\*2 + ba\_y\*\*2  
module\_bc = bc\_x\*\*2 + bc\_y\*\*2  
cos = dot\_product / Math.sqrt(module\_ba \* module\_bc)  
cos \*= -1 if cos < 0  
return Math.acos(cos)  
end