812 Largest Triangle Area

2018年5月5日 19:26

You have a list of points in the plane. Return the area of the largest triangle that can be formed by any 3 of the points.

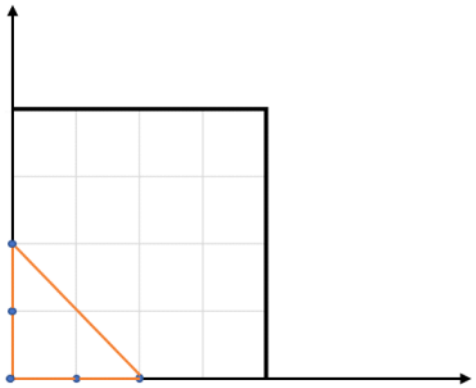
Example:

Input: points = [[0,0],[0,1],[1,0],[0,2],[2,0]]

Output: 2

Explanation:

The five points are show in the figure below. The red triangle is the largest.



Notes:

- 3 <= points.length <= 50.
- No points will be duplicated.
- -50 <= points[i][j] <= 50.
- Answers within 10^-6 of the true value will be accepted as correct.

来自 <https://leetcode.com/problems/largest-triangle-area/description/>

给定包含多个点的集合，从其中取三个点组成三角形，返回能组成的最大三角形的面积。

示例:

输入: points = [[0,0],[0,1],[1,0],[0,2],[2,0]]

输出: 2

解释:

这五个点如下图所示。组成的橙色三角形是最大的，面积为2。

注意:

- 3 <= points.length <= 50.
- 不存在重复的点。
- -50 <= points[i][j] <= 50.
- 结果误差值在 10^-6 以内都认为是正确答案。

来自 <https://leetcode-cn.com/problems/largest-triangle-area/description/>

Solution for Python3:

```python
# 暴力枚举
class Solution1:
    def largestTriangleArea(self, points):
        """
        :type points: List[List[int]]
        :rtype: float
        """
        def area(p,q,r):
            return 0.5 * abs(p[0]*q[1] + q[0]*r[1] + r[0]*p[1] - p[1]*q[0] - q[1]*r[0] - r[1]*p[0])
        return max(area(*triangle) for triangle in itertools.combinations(points, 3))


class Solution2:
    def largestTriangleArea(self, points):
            return max(0.5 * abs(i[0] * j[1] + j[0] * k[1] + k[0] * i[1]- j[0] *
```

```
        i[1] - k[0] * j[1] - i[0] * k[1])
                    for i, j, k in itertools.combinations(points, 3))
```

Solution for C++:

```
    class Solution1 {
1   public:
        double largestTriangleArea(vector<vector<int>>& points) {
2           int N = points.size();
            double ans = 0;
3           for (int i = 0; i < N; i++)
                for (int j = i+1; j < N; j++)
4                   for (int k = j+1; k < N; k++)
                        ans = max(ans, area(points[i], points[j], points[k]));
5           return ans;
        }
6       double area(vector<int> p, vector<int> q, vector<int> r) {
            return 0.5 * abs(p[0]*q[1] + q[0]*r[1] + r[0]*p[1] - p[1]*q[0] - q[1]*r[0] -
7   r[1]*p[0]);
        }
8   };

9   class Solution2 {
    public:
10      double largestTriangleArea(vector<vector<int>>& points) {
            double res = 0;
11          for (auto &i : points)
                for (auto &j : points)
12                  for (auto &k : points)
                        res = max(res, 0.5 * abs(i[0] * j[1] + j[0] * k[1] + k[0] * i[1]-
13  j[0] * i[1] - k[0] * j[1] - i[0] * k[1]));
            return res;
14      }
    };
15

16

17

18

19

20

21

22

23

24

25

26

27
```