二维平面直线上最多的点个数

W.J.Z

2019.4.10

1 问题描述

Given n points on a 2D plane, find the maximum number of points that lie on the same straight line.

```
* Definition for a point.
* struct Point {
*     int x;
*     int y;
*     Point() : x(0), y(0) {}
*     Point(int a, int b) : x(a), y(b) {}
* };
```

2 问题分析

2.1 问题解的所有情况

第 i 个点与其他点有以下三种情况：

1. 重合：判断条件 $point[i].x == point[j].x \ \&\& \ point[i].y == point[j].y$
2. 无斜率 (垂直)：判断条件 $point[i].x == point[j].x$
3. 有斜率：判断条件 $k = y/x$

3 编程思路

针对重合情况，令变量 cp 记录与第 i 个点重合的点数; 针对无斜率情况，令 vp 变量记录与第 i 个点垂直的点数；针对有斜率情况，由于斜率有很多，所有使用 map<double,int> 代表当斜率为 k 时的点数。

```
int maxPoints(vector<Point> &points) {
    int size = points.size();
    if(size<=2 )
```

```
            return size;
        int vp,cp,cmax,res=0;

        for(int i=0;i<size;i++)
        {
            vp=0;cp=0;cmax = 0;
            map<double,int> grad;
            for(int j=i+1;j<size;j++)
            {
                double x = points[i].x - points[j].x;
                double y = points[i].y - points[j].y;
                if(x==0 && y==0)  //重合点
                    cp++;
                else if(x==0)     //竖直方向
                    vp++;
                else                  //斜率
                {
                    double k = y/x;
                    if(grad.find(k)==grad.end())
                        grad.insert(pair<double,int>(k,1));
                    else
                        grad[k]++;
                }
            }
            map<double,int>::iterator it = grad.begin();
            cmax = it->second;
            while(it!=grad.end())
            {
                cmax = max(cmax,it->second);
                it++;
            }
            res = max(max(cmax,vp)+cp+1,res);
        }
        return res;
    }
```