D 寄蒜几盒 II

21101048 尹李

题意：

已知线段 P_1Q_1 和 P_2Q_2 的坐标，判断两线段**是否相交**，不相交情况下**是否平行**。

- $P_1 \neq Q_1, P_2 \neq Q_2$，线段非退化
- 坐标在 10^8 内，乘积需用 `long long`

思路：

1. 判断相交

- 两条线段相交有以下两种情况
  - 每条线段都跨越了包含另一条线段的直线
  - 一条线段的某个端点落在另一条线段上
- 第一种情况，用叉积判断向量旋转方向，即可判断一个点在线段所在直线的哪一边
- 第二种情况，用叉积可以判断一个点是否在线段所在直线上，再比较坐标判断是否在线段上

代码

```
long long direction(Point p1, Point p2, Point p3) {
    long long x0 = p2.x - p1.x, y0 = p2.y - p1.y;
    long long x1 = p3.x - p1.x, y1 = p3.y - p1.y;
    return x1 * y0 - x0 * y1;
}
bool onSegement(Point p1, Point p2, Point p3) {
    if (min(p1.x, p2.x) ≤ p3.x && p3.x ≤ max(p1.x, p2.x) &&
        min(p1.y, p2.y) ≤ p3.y && p3.y ≤ max(p1.y, p2.y)) {
        return true;
    }
    return false;
}
```

```
bool checkIntersect(Point p1, Point q1, Point p2, Point q2) {
```

```
    long long d1 = direction(p2, q2, p1);
    long long d2 = direction(p2, q2, q1);
    long long d3 = direction(p1, q1, p2);
    long long d4 = direction(p1, q1, q2);
    if (((d1 > 0 && d2 < 0) || (d1 < 0 && d2 > 0)) &&
        ((d3 > 0 && d4 < 0) || (d3 < 0 && d4 > 0))) {
        return true;
    } else if ((d1 == 0 && onSegement(p2, q2, p1)) ||
               (d2 == 0 && onSegement(p2, q2, q1)) ||
               (d3 == 0 && onSegement(p1, q1, p2)) ||
               (d4 == 0 && onSegement(p1, q1, q2))) {
        return true;
    } else {
        return false;
    }
}
```

思路：

2．判断平行

- 两个线段的向量叉乘，值为 0 则平行

代码

```
bool checkParallel(Point p1, Point q1, Point p2, Point q2) {
    Vec v1 = Vec(p1, q1), v2 = Vec(p2, q2);
    long long cross = v1.x * v2.y - v2.x * v1.y;
    if (cross == 0) {
        return true;
    }
    return false;
}
```