```
bool parallel(line m, line n) {
    return abs(det(m.a, m.b, n.a, n.b)) < EPS;
}

bool equivalent(line m, line n) {
    return abs(det(m.a, m.b, n.a, n.b)) < EPS
        && abs(det(m.a, m.c, n.a, n.c)) < EPS
        && abs(det(m.b, m.c, n.b, n.c)) < EPS;
}
```

Check if two segments (by points) intersect:

```
struct pt {
    long long x, y;

    pt() {}
    pt(long long _x, long long _y) : x(_x), y(_y) {}

    pt operator-(const pt& p) const { return pt(x - p.x, y - p.y); }
    long long cross(const pt& p) const { return x * p.y - y * p.x; }
    long long cross(const pt& a, const pt& b) const { return (a - *this).cross(b - *this);}
};

int sgn(const long long& x) { return x ≥ 0 ? x ? 1 : 0 : -1; }

bool inter1(long long a, long long b, long long c, long long d) {
    if (a > b)
        swap(a, b);
    if (c > d)
        swap(c, d);
    return max(a, c) ⩽ min(b, d);
}

//use this
bool check_inter(const pt& a, const pt& b, const pt& c, const pt& d) {
    if (c.cross(a, d) == 0 && c.cross(b, d) == 0)
        return inter1(a.x, b.x, c.x, d.x) && inter1(a.y, b.y, c.y, d.y);
    return sgn(a.cross(b, c)) ≠ sgn(a.cross(b, d)) &&
           sgn(c.cross(d, a)) ≠ sgn(c.cross(d, b));
}
```