```cpp
#include "geometry.hpp"

class Line
{
public:
    Point a, b;
    Line() : a(Point(0, 0)), b(Point(0, 0)) {}
    Line(Point a, Point b) : a(a), b(b) {}
};

ld dot(Line l, Line m)
{
    return dot((l.a - l.b), (m.a - m.b));
}

// l:line, m:line が交点を持つか
bool isis_ll(Line l, Line m)
{
    return !eq(cross(l.b - l.a, m.b - m.a), 0);
}

// l:line, s:segment
bool isis_ls(Line l, Line s)
{
    return isis_ll(l, s) &&
           (cross(l.b - l.a, s.a - l.a) * cross(l.b - l.a, s.b - l.a) < eps);
}

// s:segment, t:segment
bool isis_ss(Line s, Line t)
{
    return ccw(s.a, s.b, t.a) * ccw(s.a, s.b, t.b) <= 0 &&
           ccw(t.a, t.b, s.a) * ccw(t.a, t.b, s.b) <= 0;
}

// p が l:line 上に存在するか
bool isis_lp(Line l, Point p)
{
    return (abs(cross(l.b - p, l.a - p)) < eps);
}

bool isis_sp(Line s, Point p)
{
    return (abs(s.a - p) + abs(s.b - p) - abs(s.b - s.a) < eps);
}

// p から l に下ろした足との交点
Point proj(Line l, Point p)
{
    ld t = dot(p - l.a, l.a - l.b) / norm(l.a - l.b);
    return l.a + t * (l.a - l.b);
}

// l:line, t:line の交点
Point is_ll(Line l, Line m)
{
    Point lv = l.b - l.a, mv = m.b - m.a;
    assert(cross(lv, mv) != 0);
    return l.a + lv * cross(mv, m.a - l.a) / cross(mv, lv);
}

// p, l:line の距離
ld dist_lp(Line l, Point p)
{
    return abs(p - proj(l, p));
}

ld dist_ll(Line l, Line m)
{
    return isis_ll(l, m) ? 0 : dist_lp(l, m.a);
}

ld dist_ls(Line l, Line s)
{
    return isis_ls(l, s) ? 0 : std::min(dist_lp(l, s.a), dist_lp(l, s.b));
}

ld dist_sp(Line s, Point p)
{
    Point r = proj(s, p);
    return isis_sp(s, r) ? abs(r - p) : std::min(abs(s.a - p), abs(s.b - p));
}

ld dist_ss(Line s, Line t)
{
    if (isis_ss(s, t))
        return 0;
    return std::min({dist_sp(s, t.a), dist_sp(s, t.b), dist_sp(t, s.a), dist_sp(t, s.b)});
}

// a, b の垂直二等分線. a -> b を90度反時計回り回転
Line bisector(Point a, Point b)
{
    Point mid = (a + b) * Point(0.5, 0);
    return Line(mid, mid + (b - a) * Point(0, pi / 2));
```

```
}

// 直線 l, m のなす角を求める
ld degree_ll(Line l, Line m)
{
    ld cos_shita = dot(l, m) / (abs(l.b - l.a) * abs(m.b - m.a));
    if (cos_shita < -1.0)
        cos_shita = -1.0;
    if (cos_shita > 1.0)
        cos_shita = 1.0;
    ld shita = acos(cos_shita);
    // shita = sita * 180.0 / PI;
    return shita;
}

```