```cpp
#include "geometry.hpp"
#include "line.hpp"

class Circle
{
public:
  Point p;
  ld r;
  Circle() : p(Point(0, 0)), r(0) {}
  Circle(Point p, ld r) : p(p), r(r) {}
};

// c1, c2 の交点
std::vector<Point> is_cc(Circle c1, Circle c2)
{
  std::vector<Point> res;
  ld d = abs(c1.p - c2.p);
  ld rc = (d * d + c1.r * c1.r - c2.r * c2.r) / (2 * d);
  ld dfr = c1.r * c1.r - rc * rc;
  if (abs(dfr) < eps)
    dfr = 0.0;
  else if (dfr < 0.0)
    return res; // no intersection
  ld rs = sqrt(dfr);
  Point diff = (c2.p - c1.p) / d;
  res.emplace_back(c1.p + diff * Point(rc, rs));
  if (dfr != 0.0)
    res.emplace_back(c1.p + diff * Point(rc, -rs));
  return res;
}

std::vector<Point> is_lc(Circle c, Line l)
{
  std::vector<Point> res;
  ld d = dist_lp(l, c.p);
  if (d < c.r + eps)
  {
    ld len = (d > c.r) ? 0.0 : sqrt(c.r * c.r - d * d); //safety;
    Point nor = (l.a - l.b) / abs(l.a - l.b);
    res.emplace_back(proj(l, c.p) + len * nor);
    res.emplace_back(proj(l, c.p) - len * nor);
  }
  return res;
}

std::vector<Point> is_sc(Circle c, Line l)
{
  std::vector<Point> v = is_lc(c, l), res;
  for (Point p : v)
    if (isis_sp(l, p))
      res.emplace_back(p);
  return res;
}

// p から c への接線
std::vector<Line> tangent_cp(Circle c, Point p)
{
  std::vector<Line> ret;
  Point v = c.p - p;
  ld d = abs(v);
  ld l = sqrt(norm(v) - c.r * c.r);
  if (isnan(l))
  {
    return ret;
  }
  Point v1 = v * Point(l / d, c.r / d);
  Point v2 = v * Point(l / d, -c.r / d);
  ret.emplace_back(Line(p, p + v1));
  if (l < eps)
    return ret;
  ret.emplace_back(Line(p, p + v2));
  return ret;
}

// c1, c2 の共通接線
std::vector<Line> tangent_cc(Circle c1, Circle c2)
{
  std::vector<Line> ret;
  if (abs(c1.p - c2.p) - (c1.r + c2.r) > -eps)
  {
    Point center = (c1.p * c2.r + c2.p * c1.r) / (c1.r + c2.r);
    ret = tangent_cp(c1, center);
  }
  if (abs(c1.r - c2.r) > eps)
  {
    Point out = (-c1.p * c2.r + c2.p * c1.r) / (c1.r - c2.r);
    std::vector<Line> nret = tangent_cp(c1, out);
    ret.emplace(ret.end(), nret.begin(), nret.end());
  }
  else
  {
    Point v = c2.p - c1.p;
    v /= abs(v);
    Point q1 = c1.p + v * Point(0, 1) * c1.r;
    Point q2 = c1.p + v * Point(0, -1) * c1.r;
    ret.emplace_back(Line(q1, q1 + v));
    ret.emplace_back(Line(q2, q2 + v));
  }
  return ret;
}

```