计算几何 圆

```
struct Circle
{
    vec p;
    double r;
    Circle(){}
    Circle(vec p,double r):p(p),r(r){}
    vec vec(double rad){return vec(p.x+r*cos(rad),p.y+r*sin(rad));}
};
//判断一直线与圆是否相交并求交点
int L_C(Line l,Circle c,vector<vec>&ve)
{
    vec p = c.p;double r = c.r,dis = Dis_P_L(p,l);
    if(dcmp(dis-r)>0)return 0;
    else if(!dcmp(dis-r))
    {
        ve.push_back(P_L(p,l));
        return 1;
    }
    vec h = P_L(p,l);Vec tmp = Nol(h-p);
    double len = sqrt(r*r-dis*dis);
    ve.push_back(h+tmp*len);
    ve.push_back(h-tmp*len);
    return 2;
}
//判断圆与圆相交并求交点
int C_C(Circle a,Circle b,vector<vecs>&ve)
{
    double dis = lth(a.p-b.p);
    if(!dcmp(dis))
    {
        if(!dcmp(a.r-b.r))return -1;
        return 0;
    }
    if(dcmp(a.r+b.r-dis)<0)return 0;
    double rad = ang(b.p-a.p);
    double dlt = acos((a.r*a.r+dis*dis-b.r*b.r)/(2*a.r*dis));
    vec p1 = a.vec(rad+dlt),p2 = a.vec(rad-dlt);
    if(p1==p2)
    {
        ve.push_back(p1);
        return 1;
    }else
    {
        ve.push_back(p1);
        ve.push_back(p2);
        return 2;
    }
}
```

```
//过顶点做圆的切线，返回切线条数，在ve中是切线方向向量
int T_P_C(Circle c,vec p,vector<vec>&ve)
{
    vec v = c.p-p;
    double dis = lth(v);
    if(!dcmp(dis-c.r))
    {
        ve.push_back(Vec(-v.y,v.x));
        return 1;
    }else if(dcmp(dis-c.r)<0)return 0;
    double rad = asin(c.r/dis);
    ve.push_back(rot(v,rad));
    ve.push_back(rot(v,-rad));
    return 2;
}

//两圆公切线，返回公切线条数，共圆返回-1
//pa,pb分别是两个圆上的切点
int T_C_C(Circle a,Circle b,vector<vec>&pa,vector<vec>&pb)
{
    if(a.r>b.r)swap(a,b),swap(pa,pb);
    double dis = lth(a.p-b.p),rd = fabs(a.r-b.r),rs = a.r+b.r;
    if(dcmp(dis-rd)<0)return 0;
    if(!dcmp(dis)&&!dcmp(a.r-b.r))return -1;
    double rad = ang(b.p-a.p);
    if(!dcmp(dis-rd))
    {
        pa.push_back(a.vec(rad)),pb.push_back(b.vec(rad));
        return 1;
    }
    double drad = acos(rd/dis);
    pa.push_back(a.vec(rad+drad)),pb.push_back(b.vec(rad+drad));
    pa.push_back(a.vec(rad-drad)),pb.push_back(b.vec(rad-drad));
    if(!dcmp(dis-rs))
    {
        pa.push_back(a.vec(rad)),pb.push_back(b.vec(-rad));
        return 3;
    }else if(dcmp(dis-rs)>0)
    {
        drad = acos(rs/dis);
        pa.push_back(a.vec(rad+drad)),pb.push_back(b.vec(rad-drad));
        pa.push_back(a.vec(rad-drad)),pb.push_back(b.vec(rad+drad));
        return 4;
    }else return 2;
}
```