```cpp
/****** Vector geometry ******/
#include <bits/stdc++.h>
using namespace std;

struct Pt {
    double x, y,z;
    Pt(double a = 0, double b = 0,double c = 0):
        x(a), y(b), z(c) {}

    Pt operator-(const Pt &a) const {
        Pt ret;
        ret.x = x - a.x;
        ret.y = y - a.y;
        ret.z=z-a.z;
        return ret;
    }
};
double dist(Pt a, Pt b) {
    return hypot(a.x - b.x, a.y - b.y);
}
double length(Pt a) {
    return hypot(a.x, a.y);
}
double dot(Pt a, Pt b) {
    return a.x * b.x + a.y * b.y;
}
Pt cross2(Pt b, Pt c) {
    return Pt(b.y * c.z - c.y * b.z, b.z * c.x - c.z * b.x, b.x *  c.y);
}
double cross(Pt o, Pt a, Pt b) {
    return (a.x-o.x)*(b.y-o.y) - (a.y-o.y)*(b.x-o.x);
}
double angle(Pt a, Pt b) {
    return acos(dot(a, b) / length(a) / length(b));
}
Pt rotateRadian(Pt a, double radian) {
    double x, y;
    x = a.x * cos(radian) - a.y * sin(radian);
    y = a.x * sin(radian) + a.y * cos(radian);
    return Pt(x, y);
}
Pt getIntersection(Pt p, Pt l1, Pt q, Pt l2) {
    double a1, a2, b1, b2, c1, c2;
    double dx, dy, d;
    a1 = l1.y, b1 = -l1.x, c1 = a1 * p.x + b1 * p.y;
    a2 = l2.y, b2 = -l2.x, c2 = a2 * q.x + b2 * q.y;
    d = a1 * b2 - a2 * b1;
    dx = b2 * c1 - b1 * c2;
    dy = a1 * c2 - a2 * c1;
    return Pt(dx / d, dy / d);
}
Pt solve(Pt A, Pt B, Pt C) {
    double radABC = angle(A - B, C - B);//getting angle ABC
    double radACB = angle(A - C, B - C);//getting angle ACB

    Pt vB = rotateRadian(C - B, radABC /3);//getting the line BD for angle CBD
    Pt vC = rotateRadian(B - C, - radACB /3);//getting the line CD for angle BCD

    return getIntersection(B, vB, C, vC);
}
```

```
int main() {
    int a,b,c,d,e,i,j,T;
    scanf("%d",&T);
    for(i=1;i<=T;i++) {

        Pt A, B, C, D, E, F;
        scanf("%lf %lf", &A.x, &A.y);
        scanf("%lf %lf", &B.x, &B.y);
        scanf("%lf %lf", &C.x, &C.y);
        D = solve(A, B, C);
        E = solve(B, C, A);
        F = solve(C, A, B);
        printf("%lf %lf %lf %lf %lf %lf\n", D.x,
D.y, E.x, E.y, F.x, F.y);
    }
    return 0;
}
```