```
// Star War de Filipe Martins
#include <bits/stdc++.h>
#include <iostream>
#include <algorithm>
#include <cmath>
using namespace std;

#define fr(a,b,c) for( int a = b ; a < c ; ++a )
#define rep(a,b) fr(a,0,b)
#define db(x) cout << #x " == " << x << endl
#define dbg db
#define _ << ", " <<

#define EPS 1e-7
int comp(double x, double y) {
    if( fabs(x-y) < EPS ) return 0;
    return x < y ? -1 : 1;
}

struct P{
    double x,y,z;
    P() {}
    P(double x, double y, double z): x(x), y(y), z(z) {}

    P operator+(P b) { return P(x+b.x, y+b.y, z+b.z); }
    P operator-() { return P(-x,-y,-z); }
    P operator-(P b) { return *this+-b; }
    double operator*(P b){ return x*b.x + y*b.y + z*b.z; }
    P operator*(double k){ return P(x*k, y*k, z*k); }
    P operator%(P b){ return P(y*b.z - z*b.y, z*b.x - x*b.z, x*b.y - y*b.x); } // cross product
    P operator/(P b){ return b*(*this*b/(b*b)); } // projection of this onto b
    double operator!() { return sqrt(*this**this); } // length
} p[4], q[4];

// Distance from point c to segment [a, b]
double distSP(P a, P b, P c) {
    P pp = a + (c-a)/(b-a);
    if( !comp(!(a-pp) + !(pp-b), !(a-b)) ) return !( c-pp );
    return min(!(a-c), !(b-c));
}

// Distance from segment [a, b] to segment [c, d]
double distSS(P a, P b, P c, P d) {
    P ba = b-a;
    P cd = c-d;
    P ca = c-a;
    P w = ba%cd;
    double dd = w*w;
    if( !comp(dd,0) ) { // both segments are parallel
        return min(min(distSP(a,b,c), distSP(a,b,d)), min(distSP(c,d,a), distSP(c,d,b)));
    }
    double x = ((ca%cd)*w)/dd;
    double y = ((ba%ca)*w)/dd;
    double z = ((ba%cd)*ca)/dd;
    if( x >= 0 && x <= 1 && y >= 0 && y <= 1 ) return !(w*z);
    return min(min(distSP(a,b,c), distSP(a,b,d)), min(distSP(c,d,a), distSP(c,d,b)));
}

// Distance from point d to triangle [a, b, c]
double distPP(P a, P b, P c, P d) {
```

```
    P ba = b-a;
    P ca = c-a;
    P da = d-a;
    P w = ba%ca;
    P q = d-da/w;
    double x = (b-a)%(q-a) * w, y = (c-b)%(q-b) * w, z = (a-c)%(q-c) * w;
    if( x <= 0 && y <= 0 && z <= 0 || x >= 0 && y >= 0 && z >= 0 ) return !(da/w);
    return min( min(distSP(a,b,d), distSP(b,c,d)), distSP(c,a,d));
}

int read() {
    fr(i,0,4) scanf("%lf%lf%lf", &p[i].x, &p[i].y, &p[i].z);
    fr(i,0,4) scanf("%lf%lf%lf", &q[i].x, &q[i].y, &q[i].z);

    double dist = 1./0. ;
    fr(i,0,4) fr(j,i+1,4) fr(k,j+1,4) fr(l,0,4) {
        double d = distPP(p[i], p[j], p[k], q[l]);
        if( d < dist ) dist = d;
        d = distPP(q[i], q[j], q[k], p[l]);
        if( d < dist ) dist = d;
    }
    fr(i,0,4) fr(j,i+1,4) fr(k,0,4) fr(l,k+1,4) {
        double d = distSS(p[i], p[j], q[k], q[l]);
        if( d < dist ) dist = d;
    }

    printf("%.2lf\n", dist);

    return 1;
}

void process() {

}

int main() {
    int t = 1;
    scanf("%d", &t);
    while( t-- && read() ) process();
    return 0;
}
```