

```

61 Points 3D Operation:
62 struct point3d {
63     double x, y, z;
64     point3d() {}
65     point3d(double x, double y, double z): x(x), y(y), z(z) {}
66     point3d& operator+=(const point3d &t) {
67         x += t.x;
68         y += t.y;
69         z += t.z;
70         return *this;
71     }
72     point3d& operator--(const point3d &t) {
73         x -= t.x;
74         y -= t.y;
75         z -= t.z;
76         return *this;
77     }
78     point3d& operator*=(double t) {
79         x *= t;
80         y *= t;
81         z *= t;
82         return *this;
83     }
84     point3d& operator/=(double t) {
85         x /= t;
86         y /= t;
87         z /= t;
88         return *this;
89     }
90     point3d operator+(const point3d &t) const {
91         return point3d(*this) += t;
92     }
93     point3d operator-(const point3d &t) const {
94         return point3d(*this) -= t;
95     }
96     point3d operator*(double t) const {
97         return point3d(*this) *= t;
98     }
99     point3d operator/(double t) const {
100         return point3d(*this) /= t;
101     }
102 };
103 point3d operator*(double a, point3d b) {
104     return b * a;
105 }
106
107 //dot & cross product
108 double dot(point3d a, point3d b) {
109     return a.x * b.x + a.y * b.y + a.z * b.z;
110 }
111
112 point3d cross(point3d a, point3d b) {
113     return point3d(a.y * b.z - a.z * b.y,
114                   a.z * b.x - a.x * b.z,
115                   a.x * b.y - a.y * b.x);
116 }

```