Points 3D Operation:

```cpp
struct point3d {
    double x, y, z;
    point3d() {}
    point3d(double x, double y, double z): x(x), y(y), z(z) {}
    point3d& operator+=(const point3d &t) {
        x += t.x;
        y += t.y;
        z += t.z;
        return *this;
    }
    point3d& operator-=(const point3d &t) {
        x -= t.x;
        y -= t.y;
        z -= t.z;
        return *this;
    }
    point3d& operator*=(double t) {
        x *= t;
        y *= t;
        z *= t;
        return *this;
    }
    point3d& operator/=(double t) {
        x /= t;
        y /= t;
        z /= t;
        return *this;
    }
    point3d operator+(const point3d &t) const {
        return point3d(*this) += t;
    }
    point3d operator-(const point3d &t) const {
        return point3d(*this) -= t;
    }
    point3d operator*(double t) const {
        return point3d(*this) *= t;
    }
    point3d operator/(double t) const {
        return point3d(*this) /= t;
    }
};
point3d operator*(double a, point3d b) {
    return b * a;
}

//dot & cross product
double dot(point3d a, point3d b) {
    return a.x * b.x + a.y * b.y + a.z * b.z;
}

point3d cross(point3d a, point3d b) {
    return point3d(a.y * b.z - a.z * b.y,
                   a.z * b.x - a.x * b.z,
                   a.x * b.y - a.y * b.x);
}
```