$\times$ is used to denote vector cross product, $\bullet$ is used to denote vector dot product, and $\cdot$ is used to denote vector-scalar multiplication.

The course notes give us the derivative of an edge e_x of a rigid body x having an angular velocity of ω_x:

$$\frac{d}{dt}e_x = \omega_x \times e_x \tag{1}$$

Using this, we can find the derivative of $e_a \times e_b$ using the product rule (`https://en.wikipedia.org/wiki/Cross_product#Differentiation`):

$$\frac{d}{dt}(e_a \times e_b) = \frac{de_a}{dt} \times e_b + e_a \times \frac{de_b}{dt} = (\omega_a \times e_a) \times e_b + e_a \times (\omega_b \times e_b) \tag{2}$$

The derivative of the norm of a vector is given by (`https://math.stackexchange.com/questions/291318/derivative-of-the-2-norm-of-a-multivariate-function`):

$$\frac{d}{dt}||n|| = \frac{1}{||n||}(n \bullet \frac{d}{dt}n) \tag{3}$$

With which we find the derivative of $||e_a \times e_b||$:

$$\frac{d}{dt}||e_a \times e_b|| = \frac{1}{||e_a \times e_b||}(e_a \times e_b \bullet \frac{d}{dt}(e_a \times e_b)) \tag{4}$$

With the quotient rule and Equations 2 and 4, we can derive $\hat{n}$:

$$\begin{aligned}
\frac{d\hat{n}}{dt} = \frac{d}{dt}\frac{e_a \times e_b}{||e_a \times e_b||} &= \frac{||e_a \times e_b|| \cdot \frac{d}{dt}(e_a \times e_b) - (e_a \times e_b) \cdot \frac{d}{dt}||e_a \times e_b||}{||e_a \times e_b||^2} \\
&= \frac{||e_a \times e_b|| \cdot \frac{d}{dt}(e_a \times e_b) - (e_a \times e_b) \cdot (\frac{1}{||e_a \times e_b||}(e_a \times e_b \bullet \frac{d}{dt}(e_a \times e_b)))}{||e_a \times e_b||^2} \\
&= \frac{\frac{d}{dt}(e_a \times e_b) - \frac{e_a \times e_b}{||e_a \times e_b||} \cdot (\frac{e_a \times e_b}{||e_a \times e_b||} \bullet \frac{d}{dt}(e_a \times e_b))}{||e_a \times e_b||}
\end{aligned} \tag{5}$$

Finally, we can now state the correct code listing. The `*` symbols in the `n1` and `z` calculations should have been `^` indicating cross products. The `n` variable on the last line should have been `n1`, the normalized cross product.

```
triple computeNdot(Contact *c)
{
    if(c->vf)
    {
        return c->b->omega ^ c->n;
    }
    else
    {
        triple eadot = c->a->omega ^ ea,
               ebdot = c->b->omega ^ eb,
               n1 = ea ^ eb,
               z = eadot ^ eb + ea ^ ebdot;
        double l = length(n1);
        n1 = n1 / l;
        return (z - ((z * n1) * n1)) / l;
    }
}
```