```java
/**
 * A triangle in 3D, with three Vec vertices and a reflectivity (a Color).
 *<p>
 * For CSU540 Computer Graphics class, Spring 2005
 * CCIS, Northeastern University
 *<p>
 * Includes linear transform using a Mat
 *<br>
 * Method names ending with "Same" alter the given triangle.
 * Others return a copy.
 * @author Bob Futrelle
 * @version 22 January 2005
 */


public class Triangle  {

    /**
     * The only (non-static) field, a -element array of Vec vertices.
     */
    public Vec[] tri;

    /**
    * Tests and prints results of all methods (all are static)
     */
     public static void main(String[] args) {
     Vec v0 = new Vec(0.0,0.0,0.0);
     Vec v1 = new Vec(1.0,0.0,0.0);
     Vec v2 = new Vec(0.0,1.0,0.0);
     Triangle t1 = new Triangle(v0, v1, v2);
     System.out.println("Triangle vertices are:" + "t1");
          Mat trans = Mat.transMat(1.0, 2.0, 3.0);
          System.out.println("Translation matrix: " + trans);
     Triangle t2 = Triangle.transformTriangle(trans, t1);
     System.out.println("Translated triangle vertices are:" + t2);
    }

    /**
     * Lists the three vertices
     */
    public String toString(){
    return "\n" +
        tri[0].vec + "\n" +
        tri[1].vec + "\n" +
        tri[2].vec + "\n";
    }


    /**
     * Creates Triangle with the 3-element Vec array.
     */
    public Triangle() {
    tri  = new Vec[3];
    for(int trindex = 0; trindex < 3; trindex++)
        tri[trindex] = new Vec();
      }

    /**
     * Creates Triangle with the three given vertices
     */
    public Triangle(Vec v0, Vec v1, Vec v2) {
    Vec[] tri  = new Vec[3];
    tri[0] = v0;
    tri[1] = v1;
    tri[2] = v2;
      }
```

```
    /**
     * Produces a copy of the triangle after a linear transform is applied
     * @param mat The linear transform.
     * @param t1  The triangle to be transformed.
     * @return A new triangle which has the transformed vertices.
     */
    public static Triangle transformTriangle(Mat mat, Triangle t1){
    Triangle triReturn = new Triangle();
    for(int vert = 0; vert < 3; vert++)
        triReturn.tri[vert] = Mat.matrixXvector(mat, t1.tri[vert]);
    return triReturn;
    }

} // class Triangle
```