```java
package hevs.fragil.patapon.physics;

import com.badlogic.gdx.math.Polygon;
import com.badlogic.gdx.math.Vector2;

import ch.hevs.gdx2d.components.bitmaps.BitmapImage;
import ch.hevs.gdx2d.lib.GdxGraphics;

public class Spear extends Projectile{
    // for every arrow
    private static BitmapImage img;
    //TODO put this boolean into the projectile super class
    private boolean flipped;
    private static float[] spearVertices = { -1, 0, -1, 90, 0, 100, 1, 90, 1, 0 };

    public Spear(Vector2 startPos, int startAngle, int distance, int collisionGroup, int damage) {
        super(startPos,startAngle,collisionGroup,distance,damage,getSpearVertices(startAngle,(distance<0)),"arrow");
        this.flipped = (distance<0);
    }

    public Vector2 getSpike() {
        Vector2 temp = getBodyWorldCenter();
        double angle = getBodyAngle() + startAngle;
        temp.add((float) (Math.cos(angle) * 28), (float) (Math.sin(angle) * 28));
        return temp;
    }

    private static Vector2[] getSpearVertices(int angle, boolean flipped) {
        Polygon poly = new Polygon(spearVertices);
        poly.setOrigin(0, 40);
        if(flipped)
            poly.rotate(270-angle);
        else
```

```java
            poly.rotate(angle - 90);
        return verticesToVector2(poly.getTransformedVertices());
    }

    @Override
    public void draw(GdxGraphics g) {
        float angleDegrees;
        if(flipped)
            angleDegrees = getBodyAngleDeg() - startAngle + 180 ;
        else
            angleDegrees = getBodyAngleDeg() + startAngle;
//        // better penetration depending of the impact angle
//        int distance = (int) (5 * Math.cos(angleRadians));
//        Vector2 offset = new Vector2 (
//                (float) Math.cos(angleRadians) * distance,
//                (float) Math.sin(angleRadians) * distance
//        );

        Vector2 pos = getBodyWorldCenter();
//        pos = pos.add(offset);
        g.drawAlphaPicture(pos.x, pos.y, angleDegrees, .4f, life, img);
    }

    public static void setImgPath(String url) {
        img = new BitmapImage(url);
    }

    @Override
    public void step(float dt) {
        Vector2 v = getBodyLinearVelocity();
        float angle = getBodyAngle();
        double vNorm = Math.sqrt(v.x * v.x + v.y * v.y) * getBodyMass();

        // process lift force relative to the angle and the velocity
        float lift = (float) (-Math.cos(angle) * vNorm  * 120 * dt);
        if(v.x < 0)
            lift = -lift;
        // apply air damping
        applyBodyTorque(lift, true);

        // if this arrow is stuck, it start degrading itself
        if (stuck)
            this.life = Math.max(0, life - 0.005f);
    }
}
```