```
public class Sphere {                                               //declares public class named
sphere


    private double r;                                               // instance variable of type R
represents the radius of the sphere
    public Vector cs;                                               //cs and color are two public
instance variables of type vector. They represent the centre point and color of sphere.
    public Vector color;


    public Sphere(double radius, Vector cs, Vector color) {// Contructor method for the sphere class
        this.r = radius;                                            // It takes in the parameters
"radius", "cs" and "color".
        this.cs = cs;                                               // Method initializes the variable
intances "r", "cs" and "color" with the corresponding parameters in the constructor.
        this.color = color;
    }


    public double getRadius() {                             //A getter for the method "r" instance
variable.
        return r;                                                   //It returns the radius of the sphere
    }


    public void setRadius(int radius) {                     //A setter method for "r" instance variable.
        this.r = radius;                                    //It sets the radius of there sphere to the
value passed as a parameter.
    }


    public Vector getCs() {                                         //A getter method for the "cs"
instance variable which returns the centre point of the sphere.
        return cs;
    }


    public void setCs(Vector cs) {                          //Setter method for the "cs" instance
variable.
```

```
        this.cs = cs;                                           //Sets the centre point of the sphere to the
"vector" passed as a parameter.

    }


    public Vector getColor() {                                  //A getter method for the instance "color",
returning the color of the sphere.

        return color;

    }


    public void setColor(Vector color) {                        //A setter method for the "color" instance
variable.

        this.color = color;                                     //Sets the color of there sphere to the
"Vector" passed as a parameter.




    public Vector intersection(int i, int j, int h, int w) {    //The method called "intersection" takes in
the integer parameters "i", "j", "h" and "w".

                                                                                  //It returns a vector object.


        Vector o = new Vector();                                //Below code initializes several local
variables used in the intersection method.

                                                                                  //"o" is a "Vector" object
which reprresents the origin of the ray.

        Vector d = new Vector(0, 0, 1);                         // "d" is Every ray in same direction, 0 for x
and y. 1 for z.

        Vector cs = this.cs;                                    //cs is centre of sphere

        double r = this.r;

        Vector p = new Vector();                                //"p" is point of intersection.

        double t;                                                         // t is variable we are solving for

        double a, b, c;

        Vector v;

        Vector light = new Vector(200, 400, -220);

        o.x = i - 250;                                                    // "o.x"...Want middle pixel center
of image
```

```
        o.y = j - 250;

        o.z = -200;                                     //Z coordinae into screen, thus want
negative z axis.

        v = o.sub(cs);

        a = d.dot(d);                                        // a,b,c is the ray sphere
intersection equation

        b = 2 * v.dot(d);                               //multiply by scalar for "b"

        c = v.dot(v) - r * r;                           //ray sphere intersection equation

        double disc = b * b - 4 * a * c;

        Vector col = new Vector(0, 0, 0);

        if (disc >= 0) {                                // if the "disc" is greater than or equal to
intersection, multiply.

            t = (-b - Math.sqrt(disc)) / (2 * a); //solve equation for t, calculate disriminant

            p = o.add(d.mul(t));                        //Find intersection for light source.Direction
of the ray = intersection point.

            Vector Lv = light.sub(p);              //diffuse shading, vector from point to light source
              //Closest intersection is negative as smallest value. How far along ray intersection is.

            Lv.normalise();                             //normalise

            Vector n = p.sub(cs);                       //Surface normal

            n.normalise();

            double dp = Lv.dot(n);                      //Dot product between those 2 things
above

            if (dp >= 0) {

                col = color.mul(dp);

            }

        }


        return col;

    }
```