Intersection

Language: Python

To create synthesis images, potential intersection points between light rays and scene objects (here cylinders, spheres and cones) must be computed. This is exactly what we had to do in this project.

To do so, weed had to write a 3 dimensional equation of the considered surface, and inject into it the equation of the straight line representing the light ray.

As output you'll get a quadratic equation, with 0, 1, 2 or an infinite number of solutions that will give you the intersection points coordinates.

Usage:

```
USAGE
    ./104intersection opt xp yp zp xv yv zv p

DESCRIPTION
    opt             surface option: 1 for a sphere, 2 for a cylinder, 3 for a cone
    (xp, yp, zp)    coordinates of a point by which the light ray passes through
    (xv, yv, zv)    coordinates of a vector parallel to the light ray
    p               parameter: radius of the sphere, radius of the cylinder, or
                    angle formed by the cone and the Z-axis
```

Example:

```
marcpister@Marcs-MBP delivery % ./104intersection 1 4 0 3 0 0 -4 4
Sphere of radius 4
Line passing through the point (4, 0, 3) and parallel to the vector (0, 0, -4)
1 intersection point:
(4.000, 0.000, 0.000)
```

```
marcpister@Marcs-MBP delivery % ./104intersection 2 0 0 2 1 1 0 1
Cylinder of radius 1
Line passing through the point (0, 0, 2) and parallel to the vector (1, 1, 0)
2 intersection points:
(0.707, 0.707, 2.000)
(-0.707, -0.707, 2.000)
```

```
marcpister@Marcs-MBP delivery % ./104intersection 3 -1 -1 -1 1 1 5 30
Cone with a 30 degree angle
Line passing through the point (-1, -1, -1) and parallel to the vector (1, 1, 5)
2 intersection points:
(-1.568, -1.568, -3.842)
(-0.537, -0.537, 1.315)
```