In this exercise you need to write a Python function computing the intersection between a ray and a sphere with a cylindrical hole in a 3D scene.

The ray is defined by its origin O and direction D, and it is represented as R(t) = O + tD, where t is a scalar indicating the distance of a point from the ray origin. The function takes two parameters: 'O' for ray origin and 'D' for ray direction. They are both tuples of 3 floats in the form of (x, y, z). The function returns a float 't'. If an intersection exists, then 't' is the distance from the ray origin to the closest intersection. If no intersections exist, then 't' is -1 indicating the lack of intersections.

The object is a solid sphere centered at (0, 0, 0) with radius 1, with a portion of it removed. The removed portion is the intersection between the original sphere and a cylinder centered at (0, 0, 0) parallel to the x axis with radius 0.7 and infinite width (from x = -inf to x = inf).

You may assume that the ray always starts outside the object, and points where the ray grazes the object at one single point are considered as intersections.

Some output images from test cases are shown here.

