Exercises

1. In all the following problems, create test-suites of at least 20000 test cases, and confirm that your implementations work in all the cases.
2. Write a function that receives two line segments and returns **true** or **false**, indicating if the input segments intersect (or not).
3. Given a line L, specified by one point $L_1 \in P$ and one direction $d = (d_x, d_y, d_z)$, compute the distance from a point $P = (P_x, P_y, P_z)$ to L.
4. Given a triangle T (given by its vertices) and an input point P, decide if $P \in T$.
5. Given a non-crossed polygonal region P, possibly with interior empty regions, and given 2 points A and B, decide if it is possible to go from A to B without crossing any border of P.
6. Given a set S of segments, compute the maximum number of segments in S than can be intersected by one single line. What is the complexity of your algorithm?

Optionals

Before proceeding with the following exercises, it is recommended to install *meshlab*, and to learn the basics about the *OFF format for representing meshes.*

- Meshlab: `https://www.meshlab.net/`
- OFF Format:
  - `https://www.cs.princeton.edu/courses/archive/spr08/cos426/assn2/formats.html`
  - `https://en.wikipedia.org/wiki/OFF_(file_format)`

7. Compute a mesh of a 3d sphere centered at $0, 0, 5$ and with radious 1, approximated with cuadrilaterals.
8. Project the mesh on the plane xy
9. Project the mesh, but avoid projecting *non-visible* faces (back-faces culling)
10. Add color to the faces: Implement Flat-shaded and renderings of a *blue sphere.*
11. Add different color to the faces, such that there are *meridional regions* with different colors.
12. Make it possible to *rotate* the sphere in different directions, while visualizing it.