1. Consider a 3d holonomic robot; A dense set of obstacles are predefined at various locations in **3D**. At a given time, robot can see the obstacles only which are at **less than x** meters (x is arbitraty. Choose one number) from it. Given a starting and end point, plan a velocity control function which is defined w.r.t time between start and end points. Find the possibility of collisions and modify the control signals with minimal cost of deviation from the ideal trajectory to avoid collisions and reach end point. Plot the robot path

   Run the robot dynamics and the rest of computation in separate files, using multiprocessing (with control signal and robot position variables etc., sharing between them).

   Prog Lang: **Python**

   t = time

   Control signal: **f(t) = [$f_1(t)$; $f_2(t)$; $f_3(t)$] which is a function of time € R^3**

   Robot dynamics**: $\dot{p}$ = f(t) € R^3 , where $p = [x; y; z]$, $\dot{p} = derivative\ of\ p$ i.e., p(t) = p(t-1)+f(t)*dt , dt=sampling time**

   Obstacles: [[1,2,0],[2,3,1],…….] (for example, only)

2. Taken a point cloud (without color) file, read and , using computer vision & linear algebra extract the largest plane (or all the planes) and 3d plot it and calculate it's plane surface area. (without using pcl-python).
   Prog Lang: **Python**