Visibility

* Do not draw what is invisible.
  + Clipping, taking away lines against planes.
    - Find the intersection of a line and plane and take away the points that are not together.
  + When entire polygon is outside view volume, can be culled.
    - Really useful when triangles are group together.
  + Backface Culling
    - Remove backfacing polygons.
      * We can do this by using the z-component of the normal, only if the projection is orthographic.
    - We can do this just by seeing if the eye vector is above or below the plane.
      * Find normal and calculate plane equation. Then we check the plane with the eye position, and if the value is negative, it’s behind the eye.
* Visibility algorithms
  + Image-space algorithms
    - Operate on display primitives
    - Visibility resolved to the precision of the display
  + Object-space algorithms
    - BSP: binary space partitions.
  + Z-buffer: depth buffering, solution to the vision problem, checks to see if depth is different, and if less than removes it.
* Memory intensive (not a big deal now)
* Hardware implementation common
* Handles polygon interpenetration
* Jaggies!
  + A-buffer: accumulation buffer, each surface has data of RGB, alpha, area coverage.
  + BSP Tree: ordering polygons in a way where polygons in front re placed on front side of tree, and those behind are on hebhind prat.
  + Depth Sorting algorithm
    - Sort polygons by z
    - Resolve ambiguities where z-extents overlap
  + Sometimes ambiguities come up, we solve by doing the following:
    - If fail, exchange order of the surfaces and repeat
    - If fails, polygons can be intersected to split one of polygons.
    - Worst case, algorithm an generate O(n^2) new polygons.