Student's ***NetID***______________Student's Name____________________Grader's Name______________________
(netID == 3 letters, 3 digits: e.g. JET861 Please write clearly; make it easy to read)

EECS 351-1 Grading Sheet: Project C Winter 2018

J. Tumblin 3/1/2018

________**10% All file-naming correct + clear illustrated PDF report** with name, netID, title, goals, how to get help, user-guide, ≥4 results pictures, and sketch of your program's scene-graph (transform tree)

________**5% Ground-Plane Grid:** Project shows horizontal 'floor' of repeated shapes or lines that extend nearly endlessly to all distant horizons, and thus let us easily assess changes to camera position and aiming direction. In the world coordinate system where +z is 'up', the ground plane at z=0 spans x,y coords that appear horizontal on-screen.

________**10% ≥3 Solid, Separate, Jointed, Continually Flexing Shapes:** Properly lit & shaded 3D shapes at different ground-plane locations, with continually-changing joint angles. Wireframe is *not* acceptable!

________**5% Large, Slowly-spinning Sphere** lets us visually confirm all lighting and shading methods: sphere is easily viewable from any direction and easily lit from any direction, and rotation reveals faceted/smooth effect of Gouraud/Phong shading.

________**5% Single-Viewport Display fills entire browser window of any shape.** Re-sizing the browser window always keeps it filled with an undistorted image from a perspective camera with 35-degree vertical field-of-view; no shape distortions, no blank areas allowed except a fixed-height region to hold HTML buttons, text, edit boxes, etc.; no browser 'slider bars' needed to see complete webpage.

________**10% Smoothly adjustable 3D View Control:** User interaction for unrestricted viewpoint control: be able to aim camera in any direction without changing position: be able to move forward/backward in the gaze direction, and 'strafe' sideways left/right from any 3D position; (HINT: 'glass cylinder' method).

________**10% Obviously different-looking Materials for each separate object**
HINT: use materials parameters from those listed in `materials_Ayerdi.js` (Week08 starter code)

________**5% One 'headlight' light-source, co-located at camera eyepoint, that users can switch on/off**
(when correct, the specular highlights stay in the middle of any shiny sphere as camera moves)

________**10% A 2nd light source at user-adjustable 3D world-space position, that users can switch on/off, and with separate, user-adjustable R,G,B values for ambient, diffuse, and specular light amounts.** Light must NOT move when camera moves; moving light should cause moving reflections.

________**10% Interactive switching between all available lighting/shading methods (at least two)** without stopping or disrupting the program or its on-screen display.

________**20% Four lighting/shading methods:** select between Gouraud Shading or Phong Shading for Phong Lighting or Blinn-Phong Lighting; more methods welcome. Gouraud shading gives crudely-shaped highlights: Phong shading yields rounded highlights that can be smaller than triangles. Blinn-Phong lighting and Phong lighting yield slightly different specular highlights.

=========EXTRA CREDIT==========

________2% extra credit: 3 or more user-selected distance dependencies (ATT) for your light sources:
(must include choice between NONE, 1/dist, and 1/dist^2, with dist calc'd at each vertex)

________2% extra credit: geometric shape distortions in shaders, not reproducible by matrix transforms (e.g. twist, sinusoidal waviness, etc) implemented in Vertex Shader.

________2% extra credit: Advanced shading: implement Cook-Torrance or others such as 'toon' shaders that are not a sub-set of Phong or Blinn-Phong methods (see Lengyel book, search online),

________2% extra credit: Simple Texture Maps (Chap 5-like; emissive only is OK)

________2% extra credit per feature (circle each item completed): 3rd movable light-source; User-switched materials for just one object (>10 visually distinct mat'l choices; no effect on other objects) Advanced Texture Maps: render-to-texture (a 'mirror', etc); bump-maps; Lengyel-like: Phong-texture: use texture RGB value as specular, as bump map, displacement, etc)

====================**TOTAL POINTS/100** (24% of final grade)