Lab 09
Shadow

Shadow

- Modify the source file shadow.cpp so that it can run without extra header files.
- Modify the scene:
  1. Place 4 different light sources in the scene and only 1 of them will be enable each time (the other three will be hidden)
     - The shadow will also change according to the position of the light source.
  2. Draw the Jet in red
  3. Draw the Jet shadow in dark red color
- Keyboard Control
  - Rotate the plane along the center: (left-right & up-down)
  - Reset the plane
  - Select light sources: switch among 4 different light sources (key 1, 2, 3, 4)

Super bible v4. example code : shadow

Submit your code (shadow.cpp only)
, a demo video, and a pdf document

- Jet
- Shadow of Jet

```
// Save the matrix state and do the rotations
glPushMatrix();

// Draw jet at new orientation, put light in correct position
// before rotating the jet
glEnable(GL_LIGHTING);
glLightfv(GL_LIGHT0,GL_POSITION,lightPos);
glRotatef(xRot, 1.0f, 0.0f, 0.0f);
glRotatef(yRot, 0.0f, 1.0f, 0.0f);

DrawJet(0);

// Restore original matrix state
glPopMatrix();


// Get ready to draw the shadow and the ground
// First disable lighting and save the projection state
glDisable(GL_DEPTH_TEST);
glDisable(GL_LIGHTING);
glPushMatrix();

// Multiply by shadow projection matrix
glMultMatrixf((GLfloat *)shadowMat);

// Now rotate the jet around in the new flattend space
glRotatef(xRot, 1.0f, 0.0f, 0.0f);
glRotatef(yRot, 0.0f, 1.0f, 0.0f);

// Pass true to indicate drawing shadow
DrawJet(1);

// Restore the projection to normal
glPopMatrix();
```

glPushMatrix
glPopMatrix

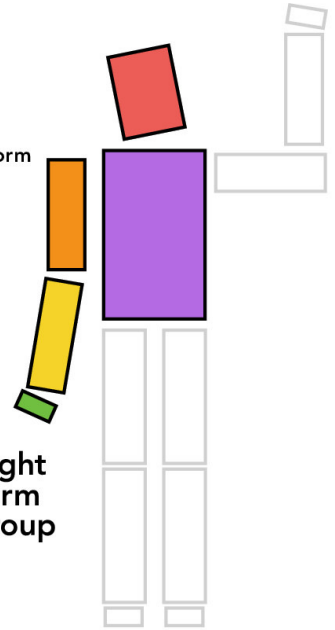
Skeleton - Hierarchical Representation

```
translate(0, 10);
    drawTorso();
        pushmatrix();  // push a copy of transform onto stack
            translate(0, 5);  // right-multiply onto current transform
            rotate(headRotation);  // right-multiply onto current transform
            drawHead();
        popmatrix();  // pop current transform off stack
        pushmatrix();  ----------------------------------------
            translate(-2, 3);
            rotate(rightShoulderRotation);
            drawUpperArm();
            pushmatrix();  ----------------------
                translate(0, -3);
                rotate(elbowRotation);
                drawLowerArm();
                pushmatrix();  ---------
                    translate(0, -3);
                    rotate(wristRotation);
                    drawHand();
                popmatrix();  ---------
            popmatrix();  ----------------------
        popmatrix(); ----------------------------------------
        ....
```

right hand

right lower arm group

right arm group



```
// Creae a projection to "squish" an object into the plane.
// Use m3dGetPlaneEquationf(planeEq, point1, point2, point3);
// to get a plane equation.
void m3dMakePlanarShadowMatrix(M3DMatrix44f proj, const M3DVector4f planeEq, const M3DVector3f vLightPos)
  {
  // These just make the code below easier to read. They will be
  // removed by the optimizer.
  float a = planeEq[0];
  float b = planeEq[1];
  float c = planeEq[2];
  float d = planeEq[3];

  float dx = -vLightPos[0];
  float dy = -vLightPos[1];
  float dz = -vLightPos[2];

  // Now build the projection matrix
  proj[0] = b * dy + c * dz;
  proj[1] = -a * dy;
  proj[2] = -a * dz;
  proj[3] = 0.0;

  proj[4] = -b * dx;
  proj[5] = a * dx + c * dz;
  proj[6] = -b * dz;
  proj[7] = 0.0;

  proj[8] = -c * dx;
  proj[9] = -c * dy;
  proj[10] = a * dx + b * dy;
  proj[11] = 0.0;

  proj[12] = -d * dx;
  proj[13] = -d * dy;
  proj[14] = -d * dz;
  proj[15] = a * dx + b * dy + c * dz;
  // Shadow matrix ready
  }
```