＜本日の課題＞

課題 5-1　3 次元図形の回転アニメーション（１）

次に示す WebGL のサンプルコード **sample05-1** を実行し，立方体が *y*軸を中心に回転しながら表示されることを確認しなさい。

sample05-1(抜粋)

```
function draw() {
  gl = GL.create();

  let angle = 0;

  gl.onupdate = function(s) {   // アニメーション処理で使用するイベントハンドラ
                                   （フレームごとに実行される関数を指定）
    angle += 50 * s;     //フレームごとに経過した時間を引数 s として受け取り angle を更新
  }

  ～ 中略 ～

  gl.ondraw = function() {
    gl.clearColor(0.0, 0.0, 0.0, 1.0);
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    gl.pushMatrix();
      gl.rotate(angle, 0.0, 1.0, 0.0);   // onupdate で更新した angle を反映する

      gl.pushMatrix();
        drawAxis();      //座標軸の描画関数
      gl.popMatrix();

      gl.pushMatrix();
        gl.color(1.0, 1.0, 0.0, 1.0);
        drawCube();      //立方体の描画関数
      gl.popMatrix();

    gl.popMatrix();

  };

  gl.fullscreen(); // 表示画面全体を指定
  gl.animate();    // アニメーション表示のためのループ処理（onupdate→ondraw をループ）
  gl.enable(gl.DEPTH_TEST);
}
```

課題 5-2　3 次元図形の回転アニメーション（２）

課題 5-1 の立方体は，Y 軸周りに回転している。これを X 軸周りに回転するように変更しなさい。

○課題 5-3　3 次元図形の回転アニメーション（３）

課題 5-1 の立方体はワイヤフレームモデルである。これをサーフェスモデルとして表示されるようにしなさい。ただし，立方体の各面の色はすべて異なるものとする。

課題 5-4　キーによる図形の操作（１）

次に示す WebGL のサンプルコード **sample05-2** を実行し，回転している図形をキー操作により拡大縮小できることを確認しなさい。

sample05-2(抜粋)

```
// obj[][0]: 図形の色，obj[][1]:図形の頂点座標
obj = [[ [0.0,1.0,0.0], [[0.0,0.0,0.0],[1.0,0.0,0.0],[1.0,1.0,0.0]] ],
       [ [0.0,1.0,0.0], [[1.0,1.0,0.0],[0.0,1.0,0.0],[0.0,0.0,0.0]] ],
       [ [1.0,0.0,0.0], [[0.0,0.0,0.0],[0.0,0.0,1.0],[0.0,1.0,1.0]] ],
       [ [1.0,0.0,0.0], [[0.0,1.0,1.0],[0.0,1.0,0.0],[0.0,0.0,0.0]] ],
         -1];

function draw() {
  const gl = GL.create();
  let angleX = 30.0, angleY = 45.0;   // X 軸周り，Y 軸周りの回転角度
  let zoom = 1.0;  // 拡大縮小用の係数

  gl.onupdate = function(s) {
    angleY += 50 * s;                      // 図形を Y 軸周りに回転
    if (GL.keys.Z)        zoom += 0.01;  // Z キーで拡大
    else if (GL.keys.X)   zoom -= 0.01;  // X キーで縮小
  };

  gl.ondraw = function() {
    gl.clearColor(1.0, 1.0, 1.0, 1.0);
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    gl.loadIdentity();
    gl.translate(0.0, 0.0, -5.0);
    gl.rotate(angleX, 1.0, 0.0, 0.0);   // X 軸周りの回転
    gl.rotate(angleY, 0.0, 1.0, 0.0);   // Y 軸周りの回転
    gl.scale(zoom, zoom, zoom);         // スケール係数の適用

    // 図形の描画
    let i = 0, t, c, vs;
    gl.begin(gl.TRIANGLES);
      while (obj[i] != -1) {
        t = obj[i];
        c = t[0]; gl.color(c[0],c[1],c[2]);
        vs = t[1];
        for (j = 0; j < 3; j++)
          gl.vertex(vs[j][0],vs[j][1],vs[j][2]);
        i++;
      }
    gl.end();
  };

  gl.fullscreen();
  gl.animate();                    // アニメーションループを開始
  gl.enable(gl.DEPTH_TEST);        // 深度テストを有効にする
}
```

補足）obj は 3D オブジェクトを定義する配列

第 1 要素は 色 (RGB) を表す 3 つの値 [R, G, B],

第 2 要素は 三角形の頂点を表す 3 つの座標 [[x1, y1, z1], [x2, y2, z2], [x3, y3, z3]]

最後の [-1] は，配列の終端を示す

課題 5-5　マウスによる図形の操作

次に示す WebGL のサンプルコード **sample05-3** を実行し，（z キーを押しているとき）マウスにより図形を回転操作できることを確認しなさい。

sample05-3(抜粋)

```
obj = [[[0.0,1.0,0.0],[[0.0,0.0,0.0],[1.0,0.0,0.0],[1.0,1.0,0.0]]],
       [[0.0,1.0,0.0],[[1.0,1.0,0.0],[0.0,1.0,0.0],[0.0,0.0,0.0]]],
       [[1.0,0.0,0.0],[[0.0,0.0,0.0],[0.0,0.0,1.0],[0.0,1.0,1.0]]],
       [[1.0,0.0,0.0],[[0.0,1.0,1.0],[0.0,1.0,0.0],[0.0,0.0,0.0]]],
        -1];

function draw() {
  const gl = GL.create();
  let angleX = 30.0, angleY = 45.0;

  gl.onmousemove = function(e) {          // マウスの移動イベントを処理
    if ( e.dragging && GL.keys.Z ) {      // Z キーを押しながらドラッグで True
      angleX += e.deltaY;                 // マウスの縦移動量は X 軸回転角度に
      angleY += e.deltaX;                 // マウスの横移動量は Y 軸回転角度に
      gl.ondraw();                        // 再描画
    }
  };

  gl.ondraw = function() {
    gl.clearColor(1.0, 1.0, 1.0, 1.0);
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    gl.loadIdentity();
    gl.translate(0.0, 0.0, -5.0);
    gl.rotate(angleX, 1.0, 0.0, 0.0);   // X 軸周りの回転
    gl.rotate(angleY, 0.0, 1.0, 0.0);   // Y 軸周りの回転

    let i = 0, t, c, vs;
    gl.begin(gl.TRIANGLES);
      while (obj[i] != -1) {
        t = obj[i];
        c = t[0]; gl.color(c[0],c[1],c[2]);
        vs = t[1];
        for (j = 0; j < 3; j++)
          gl.vertex(vs[j][0],vs[j][1],vs[j][2]);
        i++;
      }
    gl.end();
  };

  gl.fullscreen();
  gl.enable(gl.DEPTH_TEST);          // 深度テストを有効にする
}
```

補足）gl.onmousemove　マウスの移動イベントを処理する関数

e.dragging : マウスがドラッグ（ボタンを押したまま移動）しているとき True

e.deltaX : マウスの移動量（横），e.deltaY : マウスの移動量（縦）

GL.keys.Z : Z キーが押されていれば True

○課題 5-6　キーによる図形の操作（２）

Sample05-2 を参考にして，矢印キーの操作でも図形を回転操作できるようにしなさい。

Hint：矢印キー入力は次のフラグにより取得できます。

GL.keys.UP：　　上矢印キーが押されている場合に true
GL.keys.DOWN：　下矢印キーが押されている場合に true
GL.keys.LEFT：　左矢印キーが押されている場合に true
GL.keys.RIGHT：　右矢印キーが押されている場合に true

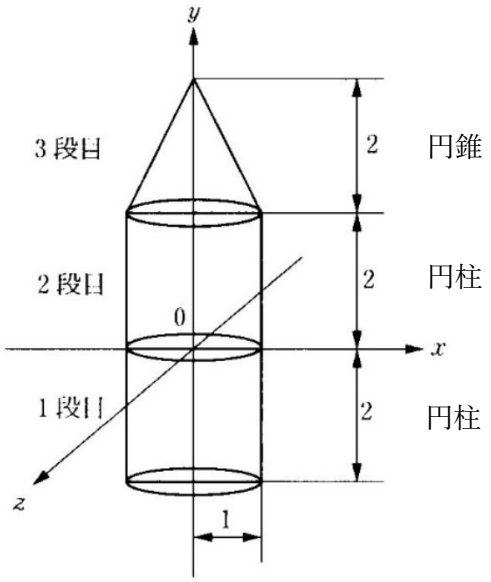
課題 5-7　深度テスト

Sample05-2,05-3 の末端にある「**gl.enable(gl.DEPTH_TEST);**」を削除すると描画にどのような影響があるか確認しなさい。また，なぜそのような描画結果となるのかを考えなさい。

○課題 5-8　ロケットの描画とアニメーション

次の指示に従いロケットを描画しなさい。なお，背景およびロケットの色は視認できる範囲で任意とする。

1）次の設計図に示すロケットを描画しなさい。
2）ロケットが y 軸周りに常に回転するようにしなさい。
3）１段目に羽を追加しなさい。ここで，羽の大きさや取り付け位置は任意とする。
4）D キーを押すと，１段目が離れるようにしなさい。



以上